

IBM Tivoli Netcool/System Service Monitors  
Version 4.0.1  
for AIX, HP-UX, Linux, Solaris, and Windows

## *Reference Guide*



**Note**

Before using this information and the product it supports, read the information in “Notices and trademarks” on page 623.

This edition applies to version 4.0.1 of Netcool/SSM (product numbers 5724-P39, 5724-P40, 5724-P41, 5724-P43) and to all subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright IBM Corporation 1996, 2013.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

# Contents

<b>Figures . . . . .</b>	<b>ix</b>
--------------------------	-----------

<b>Tables . . . . .</b>	<b>xi</b>
-------------------------	-----------

<b>About this guide . . . . .</b>	<b>xvii</b>
-----------------------------------	-------------

Who should read this guide . . . . .	xvii
Publications . . . . .	xvii
Documentation library . . . . .	xvii
Standards . . . . .	xvii
Accessing terminology online . . . . .	xviii
Accessing publications online . . . . .	xviii
Ordering publications . . . . .	xviii
Tivoli technical training . . . . .	xix
Support information . . . . .	xix
Conventions used in this guide . . . . .	xix
Typeface conventions . . . . .	xix
SNMP naming conventions . . . . .	xx
MIB object names . . . . .	xx
Operating system considerations . . . . .	xx

<b>Chapter 1. Introduction . . . . .</b>	<b>1</b>
--	----------

Netcool/SSM subagents and MIB modules . . . . .	1
Service Level Agreement (SLA) . . . . .	1
Traceroute . . . . .	1
Application Flow (Appflow) . . . . .	1
Service Discovery . . . . .	2
Programmability . . . . .	2
Host Resources . . . . .	2
System Resources . . . . .	2
Process . . . . .	2
NT Performance Monitoring . . . . .	2
Security . . . . .	2
Netcool/ASM . . . . .	2
Netcool/ASM for Oracle . . . . .	3
Netcool/ASM for Microsoft SQL . . . . .	3
Netcool/ASM for Microsoft Exchange . . . . .	3
Netcool/ASM for Microsoft IIS . . . . .	3
Netcool/ASM for Apache . . . . .	3
Netcool/ASM for IBM WebSphere . . . . .	3
Netcool/ASM for WebLogic . . . . .	3
Netcool/ASM for Microsoft Active Directory . . . . .	3
Netcool/ASM for IBM Lotus Notes/Domino Server . . . . .	4
Netcool/ASM for Sybase ASE . . . . .	4
Netcool/ASM for Microsoft Cluster Service Control . . . . .	4
Understanding MIB modules . . . . .	4

<b>Chapter 2. Agent . . . . .</b>	<b>5</b>
-----------------------------------	----------

Component files . . . . .	5
Guidelines . . . . .	5
Data control . . . . .	6
Configuration commands . . . . .	7
Agent configuration table . . . . .	8

Agent configuration variables . . . . .	8
Data control table . . . . .	8
Notification destinations . . . . .	9
Examples . . . . .	10
MIB module . . . . .	11
MIB objects . . . . .	12
Notification types . . . . .	18

<b>Chapter 3. Application flow . . . . .</b>	<b>21</b>
--	-----------

Component files . . . . .	21
Guidelines . . . . .	21
Client/server filtering . . . . .	21
Protocol filtering . . . . .	22
Event-controlled activation and deactivation . . . . .	22
SNMPv1 compatibility . . . . .	22
Configuration commands . . . . .	22
Control table . . . . .	23
Alarm table . . . . .	23
Examples . . . . .	25
MIB module . . . . .	26
MIB tables . . . . .	27
Notification types . . . . .	35

<b>Chapter 4. Application usage . . . . .</b>	<b>37</b>
---	-----------

Component files . . . . .	37
Guidelines . . . . .	37
Inivars . . . . .	37
Configuration commands . . . . .	38
Control table . . . . .	38
Examples . . . . .	39
MIB module . . . . .	39
MIB tables . . . . .	40

<b>Chapter 5. Arithmetic . . . . .</b>	<b>41</b>
--	-----------

Component files . . . . .	41
Guidelines . . . . .	41
General monitoring procedure . . . . .	41
Defining control expressions . . . . .	42
Generating events . . . . .	52
Configuration commands . . . . .	53
Control table . . . . .	53
Examples . . . . .	54
MIB module . . . . .	57
MIB tables . . . . .	58

<b>Chapter 6. Availability . . . . .</b>	<b>63</b>
--	-----------

Component files . . . . .	63
Guidelines . . . . .	63
Availability method . . . . .	63
Polling and history . . . . .	64
Event activation and deactivation . . . . .	64
Event generation . . . . .	64
Configuration commands . . . . .	65
Control table . . . . .	65
Examples . . . . .	66

MIB Module . . . . .	67
MIB tables . . . . .	68
Notification types . . . . .	70

## **Chapter 7. Crontab . . . . . 73**

Component files . . . . .	73
Guidelines . . . . .	73
Creating schedules . . . . .	73
Event times . . . . .	74
Start time . . . . .	75
Configuration commands . . . . .	75
Control table . . . . .	75
Examples . . . . .	76
MIB module . . . . .	78
MIB objects . . . . .	78
Notification types . . . . .	80

## **Chapter 8. Desktop . . . . . 83**

Component files . . . . .	83
Using Desk Tool . . . . .	83
Starting Desk Tool . . . . .	83
Sending help desk queries . . . . .	84
Changing the personal profile . . . . .	84
MIB module . . . . .	85
MIB objects . . . . .	86
Notification types . . . . .	88

## **Chapter 9. File monitoring . . . . . 91**

Component files . . . . .	91
Guidelines . . . . .	91
Monitoring procedure . . . . .	91
Selecting files and directories . . . . .	92
Selecting an attribute . . . . .	92
filemon inivars . . . . .	93
Configuration commands . . . . .	94
Control table . . . . .	94
Examples . . . . .	96
MIB module . . . . .	97
MIB tables . . . . .	98
Notification types . . . . .	101

## **Chapter 10. File transfer . . . . . 103**

Component files . . . . .	103
Guidelines . . . . .	103
Inivars . . . . .	103
Examples . . . . .	104
MIB module . . . . .	105
MIB tables . . . . .	105

## **Chapter 11. Generic alarm . . . . . 109**

Component files . . . . .	109
Guidelines . . . . .	109
Alarm modes . . . . .	110
Monitoring multiple instances . . . . .	114
Configuring the data sent in notifications . . . . .	115
SNMPv1 compatibility . . . . .	116
Configuration commands . . . . .	116
Control table . . . . .	116
Variable bindings . . . . .	118
Examples . . . . .	119

MIB module . . . . .	121
MIB tables . . . . .	121
Notification types . . . . .	126

## **Chapter 12. Host resources . . . . . 129**

Component files . . . . .	129
Guidelines . . . . .	129
Remote mount points . . . . .	129
Host Resources object support . . . . .	129
hostres inivars . . . . .	131
Device mapping (Linux only) . . . . .	133
MIB module . . . . .	133
MIB objects . . . . .	134

## **Chapter 13. IP configuration . . . . . 137**

Overview . . . . .	137
Guidelines . . . . .	137
MIB module . . . . .	137
MIB objects . . . . .	138
Notification types . . . . .	140

## **Chapter 14. LogMonX . . . . . 141**

Component files . . . . .	141
Guidelines . . . . .	141
Specifying the monitored log file . . . . .	142
Log file entry delimiters . . . . .	143
Filter expressions . . . . .	143
Log file entry size limitations . . . . .	144
Storing state information . . . . .	145
Configuration commands . . . . .	145
Control table . . . . .	146
Examples . . . . .	147
MIB module . . . . .	149
MIB objects . . . . .	150
Notification types . . . . .	153

## **Chapter 15. NT performance monitor 157**

Overview . . . . .	157
Guidelines . . . . .	157
Monitoring performance metrics . . . . .	157
Identifying performance metrics in Windows	
perfmom . . . . .	158
Data table indexing . . . . .	159
SNMPv1 compatibility . . . . .	159
Configuration commands . . . . .	159
Control table . . . . .	159
Counter table . . . . .	160
Examples . . . . .	160
MIB module . . . . .	162
MIB objects . . . . .	162

## **Chapter 16. NT services . . . . . 167**

Component files . . . . .	167
Guidelines . . . . .	167
Inivars . . . . .	167
Configuration commands . . . . .	168
Control table . . . . .	168
Examples . . . . .	168
MIB module . . . . .	169
MIB objects . . . . .	169

## **Chapter 17. Process . . . . . 175**

Component files . . . . .	175
Guidelines . . . . .	175
General procedure. . . . .	175
Allocating storage in the process exception table	176
Specifying actions . . . . .	176
Using regular expressions to identify processes	177
Configuration commands . . . . .	177
Control table . . . . .	178
Examples. . . . .	180
MIB module. . . . .	182
MIB tables . . . . .	183
Notification types . . . . .	190

## **Chapter 18. Process handle . . . . . 193**

Component files . . . . .	193
Guidelines . . . . .	193
Configuration commands . . . . .	194
Control table . . . . .	194
Examples. . . . .	194
MIB module. . . . .	196
Control table . . . . .	196
Data table . . . . .	197
Port table. . . . .	198

## **Chapter 19. Programmable . . . . . 199**

Component files . . . . .	199
General procedure. . . . .	199
Program status . . . . .	200
Specifying programs . . . . .	201
Program execution and I/O modes . . . . .	202
Event-triggered execution . . . . .	203
Event generation . . . . .	203
Configuration commands . . . . .	204
Control table . . . . .	204
In table . . . . .	205
Examples. . . . .	206
MIB module. . . . .	207
MIB tables . . . . .	208
Notification types . . . . .	212

## **Chapter 20. Scratchpad . . . . . 215**

Component files . . . . .	215
Guidelines . . . . .	215
Allocating storage. . . . .	215
Storing values . . . . .	216
Retrieving values . . . . .	216
Deleting values. . . . .	216
MIB module. . . . .	217
MIB objects . . . . .	218

## **Chapter 21. Service level agreement 219**

Component files . . . . .	219
Guidelines . . . . .	219
Client/server filtering . . . . .	219
Protocol filtering . . . . .	220
Event-controlled activation and deactivation . . . . .	220
Event generation . . . . .	220
SNMPv1 compatibility . . . . .	220
Configuration commands . . . . .	220

Control table . . . . .	221
Examples. . . . .	222
MIB module. . . . .	223
MIB tables . . . . .	223
Notification types . . . . .	228

## **Chapter 22. Solaris kstat facility . . . 231**

Component files . . . . .	231
Guidelines . . . . .	231
General monitoring procedure. . . . .	231
Threshold monitoring and event generation . . . . .	234
SNMPv1 compatibility . . . . .	234
Configuration commands . . . . .	234
Control table . . . . .	234
Field table . . . . .	235
Examples. . . . .	235
MIB module. . . . .	237
Control table . . . . .	237
Field table . . . . .	238
Data table . . . . .	239

## **Chapter 23. Recently installed software . . . . . 241**

Component files . . . . .	241
Guidelines . . . . .	241
MIB Module. . . . .	241
MIB objects . . . . .	242

## **Chapter 24. Storage . . . . . 245**

Component files . . . . .	245
Guidelines . . . . .	245
MIB module. . . . .	245
MIB tables . . . . .	246

## **Chapter 25. Service discovery . . . . . 249**

Component files . . . . .	249
Guidelines . . . . .	249
Discovering services . . . . .	249
Inivars . . . . .	250
Configuration commands . . . . .	250
Control table . . . . .	251
Examples. . . . .	252
MIB module. . . . .	252
MIB tables . . . . .	253

## **Chapter 26. Server security . . . . . 257**

Component files . . . . .	257
Attacks . . . . .	257
Guidelines . . . . .	258
Configuration commands . . . . .	258
Control table . . . . .	259
Parameter table. . . . .	259
Examples. . . . .	260
MIB module. . . . .	261
MIB tables . . . . .	262
Notification types . . . . .	265

## **Chapter 27. System resources . . . . . 267**

Component files . . . . .	267
---------------------------	-----

Guidelines . . . . .	267
Inivars . . . . .	267
Monitoring log files . . . . .	268
System resources object support . . . . .	270
Configuration commands . . . . .	273
Control table . . . . .	273
Examples . . . . .	274
MIB module . . . . .	275
MIB structure . . . . .	276
Notification types . . . . .	282

## **Chapter 28. Timer . . . . . 285**

Component files . . . . .	285
Guidelines . . . . .	285
Configuration commands . . . . .	286
Control table . . . . .	286
Examples . . . . .	288
MIB module . . . . .	288
MIB tables . . . . .	289
Notification types . . . . .	292

## **Chapter 29. Traceroute . . . . . 295**

Component files . . . . .	295
Guidelines . . . . .	295
Trace methods and firewalls . . . . .	296
UDP and TCP port numbers . . . . .	296
Packet TTL and timeout values . . . . .	297
Event activation and deactivation . . . . .	297
Creating trace destinations dynamically . . . . .	297
Configuration commands . . . . .	298
Control table . . . . .	298
Destination table . . . . .	299
Alarm table . . . . .	300
Examples . . . . .	301
MIB module . . . . .	302
MIB tables . . . . .	302
Notification types . . . . .	309

## **Chapter 30. Transaction . . . . . 311**

Component files . . . . .	311
Guidelines . . . . .	311
Inivars . . . . .	312
Application and transaction enumeration . . . . .	312
Time-outs . . . . .	313
Transaction logging . . . . .	313
Minimizing packet loss . . . . .	313
Monitoring clients and servers on the same host . . . . .	314
Monitoring Microsoft Exchange transactions . . . . .	314
SNMPv1 compatibility . . . . .	314
Configuration commands . . . . .	314
Control table . . . . .	314
Examples . . . . .	315
MIB module . . . . .	316
MIB tables . . . . .	317

## **Chapter 31. Netcool/ASM for Microsoft Exchange . . . . . 323**

Component files . . . . .	323
Guidelines . . . . .	323
Target server names . . . . .	324

Microsoft Exchange Server 5.5 configuration . . . . .	324
Inivars . . . . .	324
Events . . . . .	325
MIB module . . . . .	326
MIB objects . . . . .	326

## **Chapter 32. Netcool/ASM for Microsoft IIS . . . . . 337**

Component files . . . . .	337
Guidelines . . . . .	337
Events . . . . .	337
MIB module . . . . .	338
MIB structure . . . . .	339

## **Chapter 33. Netcool/ASM for Oracle 345**

Overview . . . . .	345
Deployment . . . . .	345
Guidelines . . . . .	346
General monitoring procedure . . . . .	346
Locating the Oracle home directory . . . . .	347
Inivars . . . . .	347
Specifying the Oracle database user . . . . .	347
Controlling queries . . . . .	348
Monitoring performance thresholds and generating events . . . . .	349
Monitoring Oracle RAC . . . . .	349
Configuration commands . . . . .	349
Instance table . . . . .	349
Query control table . . . . .	350
Examples . . . . .	351
MIB module . . . . .	352
MIB structure . . . . .	353
Configuration group . . . . .	353
Statistics group . . . . .	360
Efficiency group . . . . .	369

## **Chapter 34. Netcool/ASM for Microsoft SQL Server . . . . . 377**

Component files . . . . .	377
Guidelines . . . . .	377
General monitoring procedure . . . . .	377
Events . . . . .	378
Server connections . . . . .	379
Server aliases . . . . .	379
Configuration commands . . . . .	379
Server table . . . . .	380
MIB module . . . . .	380
MIB tables . . . . .	381

## **Chapter 35. Netcool/ASM for Apache 397**

Overview . . . . .	397
Guidelines . . . . .	397
Enabling the Apache status module . . . . .	397
Monitoring Apache Web servers in HTTPS mode . . . . .	398
Inivars . . . . .	399
MIB module . . . . .	399
MIB objects . . . . .	400

## **Chapter 36. Netcool/ASM for WebSphere . . . . . 403**

Component files . . . . .	403
Guidelines . . . . .	403
Integrating with WebSphere Application Server	
V7.0 and later . . . . .	404
Configuring the subagent . . . . .	404
Monitoring performance data . . . . .	405
Configuration commands . . . . .	406
Control table . . . . .	406
Counter table . . . . .	406
Scalars. . . . .	407
Examples. . . . .	407
MIB module. . . . .	408
Scalar objects . . . . .	409
Collection table. . . . .	410
Control table . . . . .	410
Counter table . . . . .	410
Data table . . . . .	411

## **Chapter 37. Netcool/ASM for WebLogic . . . . . 413**

Component files . . . . .	413
Guidelines . . . . .	413
Subagent configuration . . . . .	413
Monitoring performance data . . . . .	414
SNMPv1 compatibility . . . . .	415
Configuration commands . . . . .	415
Control table . . . . .	415
Scalars. . . . .	416
Examples. . . . .	416
MIB module. . . . .	417
MIB objects . . . . .	418

## **Chapter 38. Netcool/ASM for Active Directory . . . . . 421**

Component files . . . . .	421
Guidelines . . . . .	421
Monitoring Active Directory connectivity . . . . .	422
Monitoring Active Directory objects . . . . .	424
Inivars . . . . .	425
SNMPv1 compatibility . . . . .	426
Configuration commands . . . . .	426
Scalars. . . . .	426
Connection control table. . . . .	426
Monitor control table. . . . .	427
Examples. . . . .	428
MIB module. . . . .	429
MIB tables . . . . .	430

## **Chapter 39. Netcool/ASM for IBM Lotus Notes/Domino Server . . . . . 445**

Component files . . . . .	445
Guidelines . . . . .	445
Configuring the subagent . . . . .	445
General monitoring procedure. . . . .	446
Signalling changes made to the notes.ini file . . . . .	446
Configuration commands . . . . .	447
Control table . . . . .	447

Server table . . . . .	447
Scalars. . . . .	448
Examples. . . . .	448
MIB module. . . . .	449
MIB objects and tables . . . . .	450

## **Chapter 40. Netcool/ASM for Sybase ASE . . . . . 455**

Component files . . . . .	455
Guidelines . . . . .	455
Deployment . . . . .	456
Configuring Sybase . . . . .	456
Locating the Sybase installation . . . . .	457
Loading the subagent . . . . .	458
Encrypting passwords . . . . .	458
Threshold monitoring and event generation . . . . .	458
Monitoring MDA performance statistics . . . . .	459
General procedure. . . . .	459
Creating SQL queries. . . . .	460
General procedure. . . . .	460
Constructing queries . . . . .	461
Query results . . . . .	462
Threshold monitoring and event generation . . . . .	463
Executing stored procedures . . . . .	466
Extracting sp_sysmon report data . . . . .	466
Interaction between sp_sysmon and MDA tables . . . . .	467
General procedure. . . . .	467
Identifying performance metrics . . . . .	468
Generated metrics . . . . .	469
Threshold monitoring and event generation . . . . .	471
Configuration commands . . . . .	472
MDA group . . . . .	472
SQL query group . . . . .	473
Sysmon group . . . . .	474
MIB module. . . . .	475
MIB structure . . . . .	476
MDA group . . . . .	476
SQL query group . . . . .	493
Sysmon group . . . . .	495

## **Chapter 41. Netcool/ASM for MSCS 499**

Component files . . . . .	499
Guidelines . . . . .	499
Restarting groups and resources automatically . . . . .	500
Inivars . . . . .	500
Examples. . . . .	500
MIB module. . . . .	501
MIB objects . . . . .	502
Notification types . . . . .	507

## **Appendix A. MIB-2 group . . . . . 509**

system . . . . .	510
interfaces . . . . .	511
ifTable. . . . .	512
at . . . . .	514
ip group . . . . .	515
ipAddrTable. . . . .	517
ipRouteTable . . . . .	518
ipNetToMediaTable . . . . .	520
icmp . . . . .	521



tcp . . . . .	523
tcpConnTable . . . . .	524
udp group . . . . .	525
udpTable . . . . .	526
egpGroup . . . . .	526
egpNeighTable . . . . .	527
transmission group . . . . .	529
snmp group . . . . .	529

## Appendix B. RMON1 MIB group . . . 533

statistics group . . . . .	534
etherStatsTable . . . . .	534
tokenRingMLStatsTable . . . . .	536
tokenRingPStatsTable . . . . .	537
etherStats2Table . . . . .	539
tokenRingMLStats2Table . . . . .	540
tokenRingPStats2Table . . . . .	541
history group . . . . .	541
historyControlTable . . . . .	541
EtherHistoryTable . . . . .	543
tokenRingMLHistoryTable . . . . .	544
tokenRingPHistoryTable . . . . .	547
historyControl2Table . . . . .	549
alarm group . . . . .	549
alarmTable . . . . .	549
host group . . . . .	551
hostControlTable . . . . .	551
hostTable . . . . .	552
hostTimeTable . . . . .	553
hostControl2Table . . . . .	555
hostTopN group . . . . .	555
hostTopNControlTable . . . . .	555
hostTopNTable . . . . .	557
matrix group . . . . .	557
matrixControlTable . . . . .	558
matrixSDTable . . . . .	559
matrixDSTable . . . . .	560
matrixControl2Table . . . . .	561
filter group . . . . .	561
filterTable . . . . .	561
channelTable . . . . .	563
channel2Table . . . . .	564
filter2Table . . . . .	565
capture group . . . . .	565
bufferControlTable . . . . .	565
captureBufferTable . . . . .	567
event group . . . . .	568
eventTable . . . . .	568
logTable . . . . .	569
tokenRing group . . . . .	570
ringStationControlTable . . . . .	570
ringStationTable . . . . .	571
ringStationOrderTable . . . . .	573
ringStationConfigControlTable . . . . .	574
ringStationConfigTable . . . . .	575
sourceRoutingStatsTable . . . . .	576
ringStationControl2Table . . . . .	578
sourceRoutingStats2Table . . . . .	578

Configuration commands . . . . .	579
Buffer control table . . . . .	579
Ethernet statistics table . . . . .	580
History control table . . . . .	580
Filter table . . . . .	581
Host control table . . . . .	581
TopN control table . . . . .	581
Matrix control table . . . . .	582
Channel table . . . . .	582

## Appendix C. RMON2 MIB group . . . 585

protocolDir group . . . . .	586
protocolDist group . . . . .	588
protocolDistControlTable . . . . .	588
protocolDistStatsTable . . . . .	589
addressMap group . . . . .	589
addressMapControlTable . . . . .	590
addressMapTable . . . . .	591
nlHost group . . . . .	591
hlHostControlTable . . . . .	592
nlHostTable . . . . .	593
nlMatrix group . . . . .	594
hlMatrixControlTable . . . . .	594
nlMatrixSDTable . . . . .	595
nlMatrixDSTable . . . . .	596
nlMatrixTopNControlTable . . . . .	597
nlMatrixTopNTable . . . . .	599
alHost group . . . . .	600
alMatrix group . . . . .	601
alMatrixSDTable . . . . .	601
alMatrixDSTable . . . . .	602
alMatrixTopNControlTable . . . . .	602
alMatrixTopNTable . . . . .	604
usrHistory group . . . . .	605
usrHistoryControlTable . . . . .	605
usrHistoryObjectTable . . . . .	606
usrHistoryTable . . . . .	607
probeConfig group . . . . .	608
trapDestTable . . . . .	609
Configuration commands . . . . .	609
Address map control table . . . . .	610
Application-layer topN control table . . . . .	610
Higher-layer host control table . . . . .	610
Higher-layer matrix control table . . . . .	611
Network-layer Top-N matrix control table . . . . .	611
Protocol distribution control table . . . . .	612
Address map scalars . . . . .	612
User history control table . . . . .	612
User history object table . . . . .	613

## Appendix D. Regular expressions . . . 615

Regular expression syntax . . . . .	615
Regular expression examples . . . . .	617

## Notices and trademarks . . . . . 623

Trademarks . . . . .	625
----------------------	-----



## Figures

1. OID tree diagram of the agent MIB module	11
2. OID tree diagram for agent notification types	19
3. OID diagram of the appflow MIB module	26
4. OID tree diagram for appFlow notification types	35
5. OID tree diagram for appUsage	39
6. OID tree diagram of the arithmetic MIB module	57
7. OID tree diagram for arithmetic notification types	61
8. Availability event generation.	65
9. OID tree diagram of the availability MIB module	67
10. OID tree diagram for availability notification types	71
11. OID tree diagram of the crontab MIB module	78
12. OID tree diagram for crontab notification types	80
13. Helpdesk query window	84
14. Personal profile window	85
15. OID tree diagram of the desktop MIB module	86
16. OID tree diagram for desktop notification types	88
17. OID tree diagram for fileMon	97
18. OID tree diagram for fileMon notification types	102
19. OID tree diagram of the filetransfer MIB module	105
20. Hysteresis alarm mode	111
21. Rising Continuous alarm mode	111
22. Single Edge alarm mode	112
23. Single Edge Alarm mode (non-zero rising duration)	113
24. Falling Continuous alarm mode	113
25. Match alarm mode	114
26. OID tree diagram of the genAlarm MIB module	121
27. OID tree diagram for generic alarm notification types	126
28. OID tree diagram of the host MIB module	134
29. OID tree diagram of the ipconfig MIB module	138
30. OID tree diagram for ipconfig notification type	140
31. OID tree diagram of the logMonX MIB module	149
32. OID tree diagram for logMonX notification types	154
33. Identifying perfmon performance objects, counters and instances	158
34. OID tree diagram of the NT performance monitor MIB module	162
35. OID tree diagram of the ntServices MIB module	169
36. OID tree diagram of the process MIB module	183
37. OID tree diagram for psException notification type	190
38. OID tree diagram of the procHandle MIB module	196
39. Program status modes	200
40. Operation of \$INPARAM and \$OUTPARAM tokens	202
41. Program execution and I/O modes	203
42. OID tree diagram for programmable	208
43. OID tree diagram for programmable notification type	212
44. OID tree diagram of the scratch MIB module	217
45. OID tree diagram of the sla MIB module	223
46. OID tree diagram for slaViolation notification type	229
47. OID tree diagram for kstat MIB module	237
48. OID tree diagram of the srSoftware MIB module	242
49. OID tree diagram of the srStorage MIB module	246
50. OID tree diagram of the svcDisc MIB module	253
51. OID tree diagram of the svrSecurity MIB module	261
52. OID tree diagram for svrSecurityAlert notification type	266
53. OID tree diagram of the sysResources MIB module	276
54. OID tree diagram for system resources notification types	283
55. OID tree diagram of the timer MIB module	289
56. OID tree diagram for timerExpiry notification type	293
57. OID tree diagram of the traceroute MIB module	302
58. OID tree diagram for traceroute notification types	309
59. OID tree diagram of the transaction MIB module	317
60. OID tree diagram of the msexchange MIB module	326
61. OID Tree Diagram of the IIS MIB Module	339
62. OID Tree Diagram of the Netcool/ASM for Oracle MIB Module	352
63. OID tree diagram of the sqlsvr MIB module	381
64. OID tree diagram of the apache MIB module	399
65. OID tree diagram of the webSphere MIB module	409
66. OID Tree Diagram of the weblogic MIB Module	418
67. OID tree diagram of the activeDir MIB module	429
68. OID tree diagram for lotus MIB module	450
69. Excerpt from an sp_sysmon report	467
70. Relationship between report column headings and sybaseSysMonDataTable	470

71. OID tree diagram of the Netcool/ASM for Sybase MIB module . . . . .	476	112. Structure of MatrixSDTable . . . . .	559
72. OID tree diagram for mscs . . . . .	501	113. Structure of matrixDSTable . . . . .	560
73. OID tree diagram for mscs notification types . . . . .	507	114. Structure of MatrixControl2Table . . . . .	561
74. Location of the MIB-2 group . . . . .	509	115. Structure of filterTable . . . . .	562
75. Structure of system group . . . . .	510	116. Structure of channelTable . . . . .	563
76. Structure of interfaces group . . . . .	511	117. Structure of channel2Table . . . . .	564
77. Structure of IfTable . . . . .	512	118. Structure of filter2Table . . . . .	565
78. Structure of atTable . . . . .	514	119. Structure of bufferControlTable . . . . .	566
79. Structure of ip group . . . . .	515	120. Structure of captureBufferTable . . . . .	567
80. Structure of ipAddrTable . . . . .	517	121. Structure of eventTable . . . . .	568
81. Structure of ifRouteTable . . . . .	518	122. Structure of logTable . . . . .	569
82. Structure of ipNetToMediaTable . . . . .	520	123. Structure of ringStationControlTable . . . . .	570
83. Structure of the icmp group . . . . .	521	124. Structure of ringStationTable . . . . .	572
84. Structure of tcp group . . . . .	523	125. Structure of ringStationOrderTable . . . . .	574
85. Structure of tcpConnTable . . . . .	524	126. Structure of ringStationConfigControlTable . . . . .	574
86. Structure of the udp group . . . . .	525	127. Structure of ringStationConfigTable . . . . .	575
87. Structure of udpTable . . . . .	526	128. Structure of sourceRoutingStatsTable . . . . .	576
88. Structure of egpGroup . . . . .	527	129. Structure of ringStationControl2Table . . . . .	578
89. Structure of egpNeighTable . . . . .	528	130. Structure of sourceRoutingStats2Table . . . . .	579
90. Structure of transmission group . . . . .	529	131. RMON2 MIB group structure . . . . .	585
91. Structure of snmp group . . . . .	530	132. Structure of protocolDirTable . . . . .	586
92. RMON1 MIB group structure . . . . .	533	133. Structure of protocolDistTable . . . . .	588
93. Structure of etherStatsTable . . . . .	534	134. Structure of protocolDistStatsTable . . . . .	589
94. Structure of tokenRingMLStatsTable . . . . .	536	135. Structure of addressMap group . . . . .	589
95. Structure of tokenRingPStatsTable . . . . .	538	136. Structure of addressMapControlTable . . . . .	590
96. Structure of etherStats2Table . . . . .	540	137. Structure of addressMapTable . . . . .	591
97. Structure of tokenRingMLStats2Table . . . . .	540	138. Structure of nlHostControlTable . . . . .	592
98. Structure of tokenRingStats2Table . . . . .	541	139. Structure of nlHostTable . . . . .	593
99. Structure of historyControlTable . . . . .	542	140. Structure of hlMatrixControlTable . . . . .	594
100. Structure of etherHistoryTable . . . . .	543	141. Structure of nlMatrixSDTable . . . . .	596
101. Structure of tokenRingMLHistoryTable . . . . .	545	142. Structure of nlMatrixDSTable . . . . .	597
102. Structure of tokenRingPHistoryTable . . . . .	547	143. Structure of nlMatrixTopNControlTable . . . . .	598
103. Structure of historyControl2Table . . . . .	549	144. Structure of nlMatrixTopNTable . . . . .	599
104. Structure of alarmTable . . . . .	550	145. Structure of alHostTable . . . . .	600
105. Structure of hostControlTable . . . . .	551	146. Structure of alMatrixSDTable . . . . .	601
106. Structure of hostTable . . . . .	552	147. Structure of alMatrixDSTable . . . . .	602
107. Structure of hostTimeTable . . . . .	554	148. Structure of alMatrixTopNControlTable . . . . .	603
108. Structure of hostControl2Table . . . . .	555	149. Structure of alMatrixTopNTable . . . . .	604
109. Structure of hostTopNControlTable . . . . .	556	150. Structure of usrHistoryControlTable . . . . .	605
110. Structure of hostTopNTable . . . . .	557	151. Structure of usrHistoryObjectTable . . . . .	606
111. Structure of matrixControlTable . . . . .	558	152. Structure of usrHistoryTable . . . . .	607
		153. Structure of probeConfig group . . . . .	608
		154. Structure of trapDestTable . . . . .	609

# Tables

1. Common terms . . . . .	xx
2. Agent component files . . . . .	5
3. Operation of the length parameter . . . . .	7
4. Configuration command parameters - agentconfig . . . . .	8
5. Configuration command parameters - agentconfigvar . . . . .	8
6. Configuration command parameters - datactrl . . . . .	9
7. Configuration command parameters - agenttrapdest . . . . .	9
8. Agent build group (agentBuild) . . . . .	12
9. Agent identification group (agentId) . . . . .	12
10. Agent location group (agentLocation) . . . . .	13
11. Agent authorization group (agentAuth) . . . . .	13
12. Agent configuration group objects . . . . .	13
13. Agent control group objects . . . . .	14
14. Subagent data control table . . . . .	15
15. Subagent control table . . . . .	16
16. Trap destination table . . . . .	16
17. Agent configuration table . . . . .	17
18. Agent configuration variable table . . . . .	18
19. Agent inivar table . . . . .	18
20. Notification types for agent MIB . . . . .	19
21. Appflow component files . . . . .	21
22. Configuration command parameters - appflow . . . . .	23
23. Configuration command parameters - appflowalarm . . . . .	23
24. appFlow control table (appFlowControlTable) . . . . .	27
25. Application flow summary table (appflowSummaryTable) . . . . .	28
26. Application flow summary history table (appflowSummaryHistTable) . . . . .	29
27. Application flow connection table (appflowConnTable) . . . . .	30
28. Application flow alarm table (appflowAlarmTable) . . . . .	32
29. Notification types for appFlow MIB . . . . .	35
30. Appusage component files . . . . .	37
31. appusage subagent Inivars . . . . .	38
32. Configuration command parameters - appusage . . . . .	38
33. Application usage control table (appUsageControlTable) . . . . .	40
34. Application usage data table (appUsageDataTable) . . . . .	40
35. Arithmetic component files . . . . .	41
36. Control expression syntax . . . . .	47
37. Expression operators . . . . .	49
38. Built-in functions . . . . .	50
39. Configuration command parameters - arithmetic . . . . .	53
40. Arithmetic control table (arControlTable) . . . . .	58
41. Arithmetic evaluation table (arEvalTable) . . . . .	60
42. Availability component files . . . . .	63
43. Configuration command parameters - availability . . . . .	65
44. Availability control table (availabilityControlTable) . . . . .	68
45. Availability data table (availabilityDataTable) . . . . .	69
46. Availability history table (availabilityHistoryTable) . . . . .	70
47. Notification types for availability MIB . . . . .	71
48. crontab component files . . . . .	73
49. Date and time fields for crontabControlCronEventTime . . . . .	74
50. Date and time fields for crontabControlStartTime . . . . .	75
51. Configuration command parameters - crontab . . . . .	75
52. crontab control table (crontabControlTable) . . . . .	79
53. Notification types for crontab MIB . . . . .	81
54. desktop component files . . . . .	83
55. Desktop profile group objects . . . . .	86
56. Desktop help group objects . . . . .	87
57. Desktop input group objects . . . . .	87
58. Notification types for desktop MIB . . . . .	89
59. filemon component files . . . . .	91
60. Monitored file and directory attributes . . . . .	93
61. filemon subagent inivars . . . . .	93
62. Configuration command parameters - filemon . . . . .	94
63. Control table (fmControlTable) . . . . .	98
64. Data table (fmDataTable) . . . . .	100
65. filetransfer component files . . . . .	103
66. filetransfer subagent inivars . . . . .	103
67. File transfer table (filetransferTable) . . . . .	106
68. genalarm component files . . . . .	109
69. Configuration command parameters - genalarm . . . . .	116
70. Configuration command parameters - genalarmvb . . . . .	118
71. genAlarm control table (genAlarmControlTable) . . . . .	122
72. Data table (genAlarmDataTable) . . . . .	125
73. Variable binding table (genAlarmVarBindTable) . . . . .	125
74. genAlarm notification types . . . . .	126
75. hostres component files . . . . .	129
76. Host Resources supported objects . . . . .	130
77. Host Resources subagent inivars . . . . .	131
78. ipconfig component files . . . . .	137
79. Configuration group . . . . .	138
80. Interface summary group objects . . . . .	139
81. Interface table (srIfTable) . . . . .	139
82. logmonx component files . . . . .	141
83. Combinations for logMonXControlBeginDelimiter and logMonXControlEndDelimiter . . . . .	143
84. Configuration command parameters - logmonx . . . . .	146
85. Control table (logMonXControlTable) . . . . .	150

86.	Stats table (logMonXStatsTable)	152	132.	Configuration command parameters - kstatfield	235
87.	History table (logMonXHistoryTable)	153	133.	Control table (kstatControlTable)	237
88.	logMonX notification types	154	134.	Field table (kstatFieldTable)	238
89.	ntperfmon component files	157	135.	Data table (kstatDataTable)	239
90.	Configuration command parameters - perfmon	159	136.	srssoftware component files	241
91.	Configuration command parameters - perfmonctr	160	137.	Software summary group scalar objects	242
92.	ntPerfMon control table (ntPerfMonControlTable)	163	138.	Recently installed software table (srSoftwareRecentTable)	243
93.	ntPerfMon counter table (ntPerfMonCounterTable)	164	139.	srstorage component files	245
94.	ntPerfMon data table (ntPerfMonDataTable)	164	140.	Storage table (srStorageTable)	246
95.	NTServices component files	167	141.	Logical storage table (srStorageLogicalTable)	247
96.	NT services subagent inivars	167	142.	Augmented file systems table (srFsTable)	247
97.	Configuration command parameters - ntscm	168	143.	svcdisc component files	249
98.	NT services filter group (ntServiceFilter)	170	144.	svcdisc subagent inivars	250
99.	NT services control table (ntServiceControlTable)	170	145.	Configuration command parameters - svcdisc	251
100.	NT services table (ntServiceTable)	171	146.	Control table (svcDiscControlTable)	253
101.	process component files	175	147.	Service table (svcDiscSvcTable)	255
102.	Configuration command parameters - process	178	148.	Client table (svcDiscClientTable)	255
103.	TrapMask bit positions for available objects	179	149.	svrsecurity component files	257
104.	Process running table (psRunningTable)	184	150.	Configuration command parameters - svrsecurity	259
105.	Process monitor control table (psControlTable)	186	151.	Configuration command parameters - svrsecurityParam	259
106.	Process exception table (psExceptionTable)	190	152.	Server security control table (svrSecurityControlTable)	262
107.	psException notification type	191	153.	svrSecurityParamTable values for ping flood detection	263
108.	prochandle component files	193	154.	svrSecurityParamTable values for syn flood detection	264
109.	Configuration command parameters - prochandle	194	155.	svrSecurityParamTable values for port scan detection	264
110.	procHandle control table (phControlTable)	196	156.	svrSecurityParamTable values for login monitor detection	264
111.	procHandle data table (phDataTable)	197	157.	svrSecurityParamParameterA default values for log monitor detection on UNIX	265
112.	procHandle port table (phPortTable)	198	158.	Server security log table (svrSecurityLogTable)	265
113.	programmable component files	199	159.	sysres component files	267
114.	Command string tokens	201	160.	System resources subagent inivars	267
115.	Configuration command parameters - program	204	161.	System resources object support by platform	270
116.	Configuration command parameters - programin	206	162.	Configuration command parameters - logmon	273
117.	Control instance scalar objects	209	163.	System group (srSystem)	277
118.	Programmable control table (programmableControlTable)	209	164.	Processor table (srProcessorTable)	279
119.	Input table (programmableInTable)	211	165.	Disk table (srDiskTable)	279
120.	Out table (programmableOutTable)	212	166.	Control table (srLogMonControlTable)	280
121.	Error table (programmableErrorTable)	212	167.	Statistics table (srLogMonStatsTable)	281
122.	Notification types for programmable	213	168.	History table (srLogMonTable)	282
123.	Scratch component files	215	169.	User table (srUserLoginTable)	282
124.	sla component files	219	170.	Notification types for system resources	283
125.	Configuration command parameters - sla	221	171.	timer component files	285
126.	Service level agreement control table (slaControlTable)	224	172.	Configuration command parameters - timer	286
127.	Service level agreement data/history table objects	226	173.	Timer information group (timerInfo)	289
128.	kstat component files	231	174.	Timer control table (timerControlTable)	290
129.	kstat_io_t field names	233	175.	traceroute component files	295
130.	I/O formulas	233	176.	Configuration command parameters - tracert	298
131.	Configuration command parameters - kstat	234	177.	Configuration command parameters - tracedst	299

178. Configuration command parameters - tracertalarm . . . . .	300	217. Public folder table (msxPerfPublicFolderTable) . . . . .	335
179. traceroute control table (tracerouteControlTable) . . . . .	303	218. Netcool/ASM for Microsoft IIS component files . . . . .	337
180. traceroute destination table (tracerouteDestTable) . . . . .	305	219. Monitored IIS objects . . . . .	338
181. traceroute data table (tracerouteDataTable)	305	220. IIS global scalar objects . . . . .	339
182. traceroute history table (tracerouteHistoryTable) . . . . .	306	221. Site table (iisWebSiteTable) . . . . .	340
183. traceroute alarm table (tracerouteAlarmTable)	307	222. ASP scalars . . . . .	341
184. traceroute notification types. . . . .	309	223. FTP site table (iisFtpSiteTable) . . . . .	342
185. transaction component files . . . . .	311	224. SMTP table (iisSmtServerTable) . . . . .	343
186. transaction subagent inivars . . . . .	312	225. NNTP table (iisNntpServerTable) . . . . .	344
187. Typical transactionenumerationtable setup	312	226. Netcool/ASM for Oracle component files	345
188. Configuration command parameters - transaction . . . . .	314	227. oracle subagent inivars . . . . .	347
189. Transaction control table (transactionControlTable) . . . . .	318	228. Query control objects . . . . .	348
190. Transaction data table (transactionDataTable) . . . . .	319	229. Configuration command parameters - oracle	349
191. Transaction enumeration table (transactionEnumTable) . . . . .	320	230. Configuration command parameters - oraclectrl . . . . .	350
192. transaction MIB scalar objects . . . . .	321	231. Installation table (oraInstallationTable)	353
193. Netcool/ASM for Microsoft Exchange component files . . . . .	323	232. Instance table (oraInstanceTable) . . . . .	353
194. msxmon subagent inivars . . . . .	324	233. Instance information table (oraInstanceInfoTable). . . . .	355
195. Monitored MS Exchange objects . . . . .	325	234. Database file table (oraDBFileTable)	355
196. Exchange configuration group (msxConfExchange) . . . . .	327	235. Table space table (oraTableSpaceTable)	356
197. Component configuration group (msxConfComponentTable) . . . . .	327	236. Rollback table (oraRollbackTable) . . . . .	357
198. Information store configuration group (msxConfIS) . . . . .	327	237. Redo log table (oraRedoLogTable) . . . . .	357
199. Message transfer agent configuration group (msxConfMTA) . . . . .	328	238. Parameter table (oraParameterTable)	358
200. Directory services configuration group (msxConfDS) . . . . .	328	239. Database information table (oraDBInfoTable) . . . . .	359
201. Exchange server resource usage group (msxPerfExchange) . . . . .	329	240. Query control group objects (oraQueryControl) . . . . .	359
202. Exchange server component performance table (msxPerfComponentTable) . . . . .	329	241. SGA table (oraSGATable) . . . . .	360
203. Exchange server information store performance group (msxPerfIS) . . . . .	330	242. Buffer pool table (oraBufferPoolTable)	361
204. Public store performance group (msxPerfISPublic) . . . . .	330	243. SGA dictionary table (oraSGADictionaryTable) . . . . .	361
205. Private store/mailbox performance group (msxPerfISPrivate) . . . . .	330	244. Shared pool table (oraSGASharedPoolTable)	361
206. MTA performance group (msxPerfMTA)	331	245. Session table (oraSessionTable) . . . . .	362
207. Directory services performance group (msxPerfDS) . . . . .	332	246. Process table (oraProcessTable) . . . . .	363
208. Site replication service performance group (msxPerfSRS) . . . . .	332	247. Performance input/output table (oraPerformanceIODataFileTable) . . . . .	364
209. POP3 performance group (msxPerfPOP3)	333	248. Performance input/output summary table (oraPerformanceIODataFileSummaryTable) . . . . .	364
210. POP3 command table (msxPerfPOP3CmdTable) . . . . .	333	249. Performance input/output tablespace table (oraPerformanceIOTablespaceTable) . . . . .	364
211. SMTP performance group (msxPerfSMTP)	333	250. Statistics lock table (oraStatLockTable)	365
212. MMC performance group (msxPerfMMC)	334	251. Statistics latch table (oraStatLatchTable)	365
213. MMC performance group (msxPerfCCMC)	334	252. Statistics wait table (oraStatWaitTable)	366
214. IM performance group (msxPerfIM)	334	253. Statistics wait table (ora64StatWaitTable)	366
215. Address list group (msxPerfAL) . . . . .	335	254. Statistics application table (oraStatApplicationTable) . . . . .	366
216. Mailbox table (msxPerfMailboxTable)	335	255. Statistics library cache table (oraStatLibCacheTable) . . . . .	367
		256. Statistics queue cache table (oraStatQueueTable) . . . . .	368
		257. Dispatcher statistics table (oraStatDispatcherTable) . . . . .	368
		258. Parallel servers statistics table (oraStatPQTable) . . . . .	368
		259. Batch job statistics table (oraStatBatchJobTable) . . . . .	369
		260. Replication statistics table (oraStatReplicationTable) . . . . .	369



261.	Efficiency memory table (oraEffMemTable)	369	302.	Configuration Command Parameters - websphertext	406
262.	Efficiency storage table (oraEffStorageTable)	370	303.	Configuration command parameters - websphereglobal	407
263.	Efficiency application table (oraEffAppTable)	371	304.	webSphere scalar objects	409
264.	SQL control table (oraEffSQLControlTable)	372	305.	Collection table (wsCollectionTable)	410
265.	SQL data table (oraEffSQLDataTable)	373	306.	Control table (wsControlTable)	410
266.	Library cache efficiency table (oraEffLibCacheTable)	374	307.	Counter table (wsCounterTable)	410
267.	Queue efficiency table (oraEffQueueTable)	374	308.	Data table (wsDataTable)	411
268.	Dispatcher efficiency table (oraEffDispatcherTable)	374	309.	Netcool/ASM for WebLogic component files	413
269.	Wait efficiency table (oraEffWaitTable)	375	310.	wl_setup parameters	414
270.	Parallel query efficiency table (oraEffPQTable)	375	311.	Configuration command parameters - weblogic	415
271.	Netcool/ASM for Microsoft SQL Server component files	377	312.	Configuration command parameters - weblogicglobal	416
272.	Monitored SQL Server objects	378	313.	Bean type table (wlBeanTypeTable)	419
273.	Configuration command parameters - sqlsvr	380	314.	Bean attribute table (wlBeanAttrTable)	419
274.	Server table (sqlsvrServerTable)	381	315.	Bean instance table (wlBeanInstTable)	419
275.	Server description table (sqlsvrSvrDescTable)	382	316.	Control table (wlControlTable)	419
276.	Configuration table (sqlsvrSvrCfgTable)	383	317.	Data table (wlDataTable)	420
277.	Database description table (sqlsvrDbDescTable)	384	318.	Netcool/ASM for Active Directory component files	421
278.	Database configuration table (sqlsvrDbCfgTable)	386	319.	adConnectControlConnectTo object values	422
279.	SQL Server database backup table (bacsqsvrDbBackupTable)	386	320.	activedir subagent inivars	425
280.	SQL Server jobs table (sqlsvrJobsTable)	386	321.	Configuration command parameters - activedirglobal	426
281.	SQL Server jobs summary table (sqlsvrJobsSummaryTable)	387	322.	Configuration command parameters - activedirconn	426
282.	SQL Server statistics table (sqlsvrSvrStatsTable)	388	323.	Configuration command parameters - activedirmon	427
283.	SQL Server process table (sqlsvrSvrProcessTable)	389	324.	activeDir configuration scalars	430
284.	Internal process table (sqlsvrInternalProcTable)	389	325.	Active Directory domain controller configuration group scalar objects	430
285.	Database statistics table (sqlsvrDbStatsTable)	390	326.	File replica service group scalar objects	432
286.	Database files table (sqlsvrDbFilesTable)	390	327.	Address book group scalar objects	432
287.	Database performance table (sqlsvrDbPerfTable)	391	328.	Directory service group scalar objects	433
288.	General statistics table (sqlsvrGenStatsTable)	391	329.	Directory service read operations group scalar objects	434
289.	SQL statistics table (sqlsvrSQLStatsTable)	392	330.	Directory service write operations group scalar objects	434
290.	Locks table (sqlsvrLocksTable)	392	331.	Directory service search operations group scalar objects	435
291.	Latches Table (sqlsvrLatchesTable)	392	332.	Replication management group scalar objects	435
292.	Access methods table (sqlsvrAccessMethodsTable)	393	333.	Replication table row objects	436
293.	Buffer manager table (sqlsvrBufferMgrTable)	394	334.	Inbound DRA group scalar objects	436
294.	Cache manager table (sqlsvrCacheMgrTable)	394	335.	Outbound DRA group scalar objects	437
295.	Memory manager table (sqlsvrMemMgrTable)	395	336.	Distributed file system control table	438
296.	Netcool/ASM for Apache component files	397	337.	Distributed file system target table	439
297.	apache subagent inivars	399	338.	Connection control table	439
298.	Top-level scalars for Apache MIB	400	339.	Connection data table.	441
299.	Site table (apacheSiteTable)	401	340.	Monitor control table (adMonitorControlTable)	441
300.	Netcool/ASM for WebSphere component files	403	341.	Monitor data table (adMonitorDataTable)	442
301.	Configuration command parameters - websphere	406	342.	Netcool/ASM for IBM Lotus Domino component files	445
			343.	Domino API library location and filename by platform	446
			344.	Configuration command parameters - lotus	447

345. Configuration command parameters - lotusserver . . . . .	448	390. sybaseMDASysStatementTable row objects	492
346. Configuration command parameters - lotusglobal . . . . .	448	391. sybaseMDACachedProceduresTable row objects . . . . .	492
347. lotus MIB scalar objects . . . . .	451	392. sybaseMDASysSQLTextTable row objects	493
348. Server table (loServerTable) . . . . .	451	393. sybaseSQLQueryControlTable row objects	493
349. Facility table (loFacilityTable) . . . . .	451	394. sybaseSQLQueryInTable row objects	494
350. Control table (loControlTable) . . . . .	452	395. sybaseSQLQueryResultsTable row objects	494
351. Data table (loDataTable) . . . . .	453	396. sybaseSysMonHeaderTable row objects	495
352. Task table (loTaskTable) . . . . .	453	397. sybaseSysMonControlTable row objects	495
353. Netcool/ ASM for Sybase component files	455	398. sybaseSysMonMatchTable row objects	496
354. Indexing results for multiple columns	462	399. sybaseSysMonDataTable row objects	496
355. Configuration command properties - sybasemda . . . . .	473	400. mscs component files . . . . .	499
356. Configuration command properties - sybasesql . . . . .	473	401. MSCS subagent inivars . . . . .	500
357. Configuration command properties - sybasesqlin . . . . .	474	402. mscs scalar objects . . . . .	502
358. Configuration command properties - sysmon . . . . .	474	403. Node table (mscsNodeTable) . . . . .	503
359. Configuration command properties - sysmonheader . . . . .	475	404. Group table (mscsGroupTable) . . . . .	503
360. Configuration command properties - sysmonmatch . . . . .	475	405. Resource table (mscsResourceTable) . . . . .	504
361. sybaseMDAControlTable row objects	476	406. Network table (mscsNetworkTable) . . . . .	505
362. sybaseMDAStateTable row objects . . . . .	477	407. Network interface table (mscsNetworkInterfaceTable) . . . . .	506
363. sybaseMDAEngineTable row objects . . . . .	478	408. Notification types for mscs MIB . . . . .	508
364. sybaseMDADataCacheTable row objects	479	409. Description of system group objects	510
365. sybaseMDAProcedureCacheTable row objects . . . . .	479	410. Description of interfaces group objects	511
366. sybaseMDAOpenDatabasesTable row objects . . . . .	479	411. Description of IfTable objects . . . . .	512
367. sybaseMDASysWorkerThreadTable row objects . . . . .	480	412. Description of atTable objects . . . . .	514
368. sybaseMDANetworkIOTable row objects	480	413. Description of ip group objects . . . . .	515
369. sybaseMDAErrorLogTable row objects	481	414. Description of ipAddrTable Objects	518
370. sybaseMDALocksTable row objects . . . . .	481	415. Description of ifRouteTable Objects . . . . .	519
371. sybaseMDADeadLockTable row objects	482	416. Description of ipNetToMediaTable Objects	520
372. sybaseMDAWaitClassInfoTable row objects	483	417. Description of the icmp Group Objects	522
373. sybaseMDAWaitEventInfoTable row objects	483	418. Description of tcp group objects . . . . .	523
374. sybaseMDACachedObjectTable row objects	483	419. Description of tcpConnTable objects	525
375. sybaseMDACachePoolTable row objects	484	420. Description of udp Group Objects . . . . .	525
376. sybaseMDAOpenObjectActivityTable row objects . . . . .	484	421. Description of udpTable Objects . . . . .	526
377. sybaseMDAIOQueueTable row objects	485	422. Description of egpGroup objects . . . . .	527
378. sybaseMDADeviceIOTable row objects	485	423. Description of egpNeighTable objects	528
379. sybaseMDASysWaitsTable row objects	486	424. Description of snmp group objects . . . . .	530
380. sybaseMDAProcessTable row objects	486	425. Descriptions of etherStatsTable objects	534
381. sybaseMDAProcessLookupTable row objects	487	426. Descriptions of tokenRingMLStatsTable objects . . . . .	536
382. sybaseMDAProcessActivityTable row objects . . . . .	488	427. Descriptions of tokenRingPStatsTable objects	538
383. sybaseMDAProcessNetIOTable row objects	488	428. Descriptions of etherStats2Table objects	540
384. sybaseMDAProcessObjectTable row objects	489	429. Descriptions of tokenRingMLStats2Table objects . . . . .	540
385. sybaseMDAProcessWaitsTable row objects	489	430. Descriptions of tokenRingStats2Table objects	541
386. sybaseMDAProcessStatementTable row objects . . . . .	489	431. Descriptions of historyControlTable objects	542
387. sybaseMDAProcessSQLTextTable row objects . . . . .	490	432. Descriptions of etherHistoryTable objects	543
388. sybaseMDAProcessProceduresTable row objects . . . . .	491	433. Descriptions of tokenRingMLHistoryTable objects . . . . .	545
389. sybaseMDASysPlanTextTable row objects	491	434. Descriptions of tokenRingPHistoryTable objects . . . . .	547
		435. Descriptions of historyControl2Table objects	549
		436. Descriptions of alarmTable objects . . . . .	550
		437. Descriptions of hostControlTable objects	551
		438. Description of hostTable objects . . . . .	553
		439. Descriptions of hostTimeTable objects	554
		440. Descriptions of hostControl2Table objects	555
		441. Descriptions of hostTopNControlTable objects	556
		442. Descriptions of hostTopNTable objects	557
		443. Descriptions of matrixControlTable objects	558
		444. Structure of MatrixSDTable . . . . .	559



445.	Descriptions of matrixDSTable objects	560	476.	Descriptions of addressMapTable objects	591
446.	Descriptions of MatrixControl2Table objects	561	477.	Descriptions of hlHostControlTable objects	592
447.	Descriptions of filterTable objects . . . . .	562	478.	Descriptions of nlHostTable objects . . . . .	593
448.	Description of channelTable objects . . . . .	563	479.	Descriptions of nlMatrixControlTable objects	595
449.	Descriptions of channel2Table objects	564	480.	Descriptions of nlMatrixSDTable objects	596
450.	Descriptions of filter2Table objects . . . . .	565	481.	Descriptions of nlMatrixDSTable objects	597
451.	Descriptions of bufferControlTable objects	566	482.	Descriptions of nlMatrixTopNControlTable objects . . . . .	598
452.	Descriptions of captureBufferTable objects	567	483.	Descriptions of nlMatrixTopNTable Objects	599
453.	Descriptions of eventTable objects . . . . .	568	484.	Descriptions of alHostTable objects . . . . .	600
454.	Descriptions of logTable Objects . . . . .	569	485.	Descriptions of alMatrixSDTable objects	601
455.	Descriptions of ringStationControlTable objects . . . . .	570	486.	Descriptions of alMatrixDSTable objects	602
456.	Descriptions of ringStationTable objects	572	487.	Descriptions of nlMatrixTopNControlTable objects . . . . .	603
457.	Descriptions of ringStationOrderTable objects	574	488.	Descriptions of alMatrixTopNTable objects	604
458.	Descriptions of ringStationConfigControlTable objects . . . . .	575	489.	Descriptions of usrHistoryControlTable objects . . . . .	606
459.	Descriptions of ringStationConfigTable objects . . . . .	575	490.	Descriptions of usrHistoryObjectTable objects . . . . .	606
460.	Descriptions of sourceRoutingStatsTable objects . . . . .	576	491.	Descriptions of usrHistoryTable objects	607
461.	Descriptions of ringStationControl2Table objects . . . . .	578	492.	Descriptions of probeConfig objects	608
462.	Descriptions of sourceRoutingStats2Table objects . . . . .	579	493.	Descriptions of trapDestTable objects	609
463.	Configuration command parameters - capture . . . . .	579	494.	Configuration command parameters - addressmap . . . . .	610
464.	Configuration command parameters - etherstats . . . . .	580	495.	Configuration command parameters - almatrixtopn . . . . .	610
465.	Configuration command parameters - history . . . . .	580	496.	Configuration command parameters - hlhost	611
466.	Configuration command parameters - filter	581	497.	Configuration command parameters - hlmatrix . . . . .	611
467.	Configuration command parameters - hosts	581	498.	Configuration command parameters - nlmatrixtopn . . . . .	611
468.	Configuration command parameters - hosttopn . . . . .	582	499.	Configuration command parameters - protocoldist . . . . .	612
469.	Configuration command parameters - matrix . . . . .	582	500.	Configuration command parameters - addressmapglobal . . . . .	612
470.	Configuration command parameters - channel . . . . .	582	501.	Configuration command parameters - usrhistory . . . . .	612
471.	Descriptions of protocolDirTable Objects	586	502.	Configuration command parameters - usrhistoryvar . . . . .	613
472.	Descriptions of protocolDistTable objects	588	503.	Regular expression syntax . . . . .	615
473.	Descriptions of protocolDistStatsTable objects	589	504.	Regular expression examples . . . . .	617
474.	Descriptions of addressMap objects . . . . .	589			
475.	Descriptions of addressMapControlTable objects . . . . .	590			

---

## About this guide

This guide provides detailed reference information for Netcool/SSM subagents and the Netcool/ASM suite of monitors.

---

## Who should read this guide

This guide is intended for network administrators and engineers who install and use Netcool/SSM to monitor networks and hosts. It provides detailed, cross-platform information about the tools, functions, and capabilities of Netcool/SSM. Use this guide to assist you in designing and configuring your network management and monitoring environment.

To use Netcool/SSM effectively, and to understand the information in this guide, you should already be familiar with network technologies, network management practices, and the Simple Network Management Protocol (SNMP).

---

## Publications

This section lists publications in the Netcool/SSM library. It also describes how to access Tivoli publications online and how to order them.

### Documentation library

The following documents are available in the Netcool/SSM library:

- *Netcool/SSM Administration Guide*  
Provides information about installing and using Netcool/SSM.
- *Netcool/SSM Reference Guide*  
Provides detailed reference material covering the subagents and MIB modules included in Netcool/SSM.
- *Netcool/SSM Patch Installation Guide*  
Provides instructions on installing patches to Netcool/SSM.
- *Netcool/SSM Release Notes*  
Provides the latest information about Netcool/SSM.

### Standards

Netcool/SSM and Netcool/ASM MIB modules are based on the Internet standard Network Management Framework which consists of the following components:

- RFC 2578 which defines the SMI [SMIPv2], the mechanisms used for describing and naming objects for the purpose of management.
- RFC 1213 which defines MIB-2, the core set of managed objects for the Internet suite of protocols.
- RFC 1905 which defines the SNMP [SNMPv2], the protocol used for network access to managed objects.

Another important component is RFC 2021 (RMON2) the network monitoring Internet standard on which Netcool/SSM is based. RMON2 is a MIB that defines objects for managing remote network monitoring devices over TCP/IP networks.

## Accessing terminology online

The *Tivoli Software Glossary* includes definitions for many of the technical terms related to Tivoli software. The *Tivoli Software Glossary* is available at the following Tivoli software library Web site:

<http://publib.boulder.ibm.com/tividd/glossary/tivoliglossarymst.htm>

The IBM Terminology Web site consolidates the terminology from IBM product libraries in one convenient location. You can access the Terminology Web site at the following Web address:

<http://www.ibm.com/ibm/terminology>

## Accessing publications online

IBM posts publications for this and all other Tivoli products, as they become available and whenever they are updated, to the Tivoli software information center Web site. Access the Tivoli software information center by first going to the Tivoli software library at the following Web address:

<http://www.ibm.com/software/tivoli/library>

Scroll down and click the **Product manuals** link. In the Tivoli Product Documents Alphabetical listing window, click **M** to access all IBM Tivoli Monitoring product manuals.

**Note:** If you print PDF documents on other than letter-sized paper, set the option in the File -> Print window that allows Adobe Reader to print letter-sized pages on your paper.

## Ordering publications

You can order many Tivoli publications online at the following Web site:

<http://www.elink.ibm.link.ibm.com/public/applications/publications/cgibin/pbi.cgi>

You can also order by telephone by calling one of these numbers:

- In the United States: 800-879-2755
- In Canada: 800-426-4968

In other countries, contact your software account representative to order Tivoli publications. To locate the telephone number of your local representative, perform the following steps:

1. Go to <http://www.ibm.com/planetwide/>
2. Select the letter that your country starts with and click the name of your country. A list of numbers for your local representatives is displayed.

---

## Tivoli technical training

For information about Tivoli technical training, refer to the following IBM Tivoli Education Web site:

<http://www.ibm.com/software/tivoli/education>

---

## Support information

If you have a problem with your IBM software, you want to resolve it quickly. IBM provides a number of ways for you to obtain the support you need.

- Searching knowledge bases: You can search across a large collection of known problems and workarounds, Technotes, and other information.
  - Obtaining fixes: You can locate the latest fixes that are already available for your product.
  - Contacting IBM Software Support: If you still cannot solve your problem, and you need to work with someone from IBM, you can use a variety of ways to contact IBM Software Support.
- 

## Conventions used in this guide

This guide uses several conventions for operating system-dependent commands and paths, special terms, actions, and user interface controls.

### Typeface conventions

This guide uses the following typeface conventions:

#### **Bold**

- Lowercase commands and mixed case commands that are otherwise difficult to distinguish from surrounding text
- Interface controls (check boxes, push buttons, radio buttons, spin buttons, fields, folders, icons, list boxes, items inside list boxes, multicolumn lists, containers, menu choices, menu names, tabs, property sheets), labels (such as **Tip:**, and **Operating system considerations:**)
- Keywords and parameters in text

#### *Italic*

- Words defined in text
- Emphasis of words (words as words)
- New terms in text (except in a definition list)
- Variables and values you must provide

#### **Monospace**

- Examples and code examples
- File names, programming keywords, and other elements that are difficult to distinguish from surrounding text
- Message text and prompts addressed to the user
- Text that the user must type
- Values for arguments or command options

## SNMP naming conventions

Table 1 lists definitions for terms that are commonly used when working with the SNMP features provided in Netcool/SSM.

*Table 1. Common terms*

Term	Meaning
MIB	An acronym for Management Information Base. The MIB provides a structured set of data variables, or objects, that represent the resources to be managed. The MIB consists of a set of MIB modules that add information to it.
object	A data variable.
OID	An object identifier that uniquely identifies an object within the MIB.
subagent	A module that implements a part of the network management and monitoring functions provided by Netcool/SSM.
trap	An unsolicited message sent by the subagent to notify the management station of an event. Traps are also known as notifications.
tree	Shows the hierarchical structure between related objects.

## MIB object names

In this guide, MIB objects are referred to by an abbreviated object name or by the name of the function they provide. The object name usually has a prefix that specifies the MIB module in which the object appears, while the remainder of the object name usually describes its function.

For example, in the object name `sysObjectID`:

- The prefix is `sys`
- The abbreviated object name is `ObjectID`
- The function is object identifier

The function name has initial capitals in normal body type with a space between words, whereas the object name is shown as a single word in monospace font. Object names are taken directly from MIB module definition documents.

## Operating system considerations

All command line formats and examples are provided for both the standard UNIX shell and the Windows command-line. UNIX is case-sensitive. You must type commands in the case shown in the book.

---

## Chapter 1. Introduction

Netcool/SSM provides real-time monitoring for systems, applications, and networks. It is a key technology in maintaining overall performance and availability of business and application services. Netcool/SSM is based on the Simple Network Management Protocol (SNMP) and implements the MIB-II, RMON and RMON2 network monitoring standards.

Netcool/SSM is a rich data source for network management systems such as Netcool/OMNIbus that provide performance evaluation, reporting, event management, and fault resolution tools. You can configure Netcool/SSM to send event-related data to management systems in the form of SNMP notifications, or use the management system to poll Netcool/SSM for performance data using SNMP requests.

---

### Netcool/SSM subagents and MIB modules

The system service monitoring capabilities provided by Netcool/SSM are implemented by subagents and their MIB modules.

Subagents are modules that you load and run on the Netcool/SSM master agent. A subagent consists of a binary implementation of its associated MIB module. The MIB module defines an SNMP interface, control facilities and storage for the performance metrics monitored.

Netcool/SSM includes subagents for the following system and application monitoring tasks.

#### Service Level Agreement (SLA)

The sla subagent provides metrics for application performance (response time and availability). The MIB allows threshold values to be set (dynamically) for application performance metrics and can generate an SNMP trap. The SLA MIB measures various metrics about client, server and network response time and can be configured to monitor individual applications, clients or servers.

#### Traceroute

The traceroute subagent performs periodic traceroutes that show the path that an application takes from the client to the server. Data is stored for future baselining and for historical reporting of routes and hop delay information.

#### Application Flow (Appflow)

The appflow subagent measures information on application connections, called flows. The metrics gathered include connection start-time, setup-time, duration, teardown time, inter-packet arrival times, and retransmissions— information useful for application policy management, quality of service (by applications such as voice and video) monitoring and capacity planning.

## Service Discovery

The svcdisc subagent provides information on the relationship between clients, servers and the applications they are using. This supports application topology discovery as it can be used to discover the relationship between clients and servers for each different application.

## Programmability

The programmable subagent can invoke any program script to be executed manually or in response to specific events. Using this feature you can add customized functionality to the agent, allowing it to take action on specific events.

## Host Resources

Netcool/SSM can monitor all of the variables that are part of the Host Resource MIB (RFC1514). These include CPU, memory and disk usage and as well as installed software and its usage on the host system.

## System Resources

The sysres subagent extends the functionality of host resource management to include additional data about CPU, memory and disks. It allows you to monitor system and application log files and gather data on system and application events.

## Process

The process subagent provides facilities to monitor, start, stop and restart applications and processes running on a server or other network device. It can monitor multiple attributes of a running process, including status, memory usage, size and resident set size, (RSS), process time, threads, and disk and network I/O.

## NT Performance Monitoring

The ntperfmon subagent views data from performance tables on Windows hosts, which enables it to monitor tasks and processes. These elements include ACS/RSVP Service, browser, cache, distribution transaction, authentication, network, printer, disk, process, RAS, and system metrics.

## Security

The svrsecurity subagent provides host-based intrusion detection by looking for any indication of a security breach. It can detect and recognize excessive login attempts, denial-of-service attacks and port-scanning attempts.

---

## Netcool/ASM

To complement the Netcool/SSM management of network devices and systems, Netcool/Application Service Monitors™ (Netcool/ASM™) provide detailed monitoring for leading commercial applications and databases.

Each Netcool/ASM is a subagent that loads and runs just like any Netcool/SSM subagent. The Netcool/ASMs are installed with Netcool/SSM during the Netcool/SSM installation process.



## **Netcool/ASM for Oracle**

The Netcool/ASM for Oracle monitors various aspects of Oracle database performance such as database, table space, buffer pool, processes, sessions, and I/O.

## **Netcool/ASM for Microsoft SQL**

The Netcool/ASM for Microsoft SQL monitors performance metrics for the server, database configuration, internal processes, and locks as well as SQL statistics and available buffer, cache, and memory.

## **Netcool/ASM for Microsoft Exchange**

The Netcool/ASM for Microsoft Exchange monitors both the configuration and performance aspects of Exchange servers, providing metrics such as memory and CPU usage, page faults, SMTP queue length, POP3 queue length, and cache hit ratios. It can automatically generate alarms based on the value of particular performance metrics.

## **Netcool/ASM for Microsoft IIS**

Netcool/ASM for Microsoft IIS provides comprehensive support for monitoring performance and availability of Microsoft Internet Information Server/Services (IIS) servers. It addresses aspects such as Web, FTP, SMTP and NNTP performance, and contains metrics such as failed Web and FTP logins, pages not found, traffic monitors and FTP traffic.

## **Netcool/ASM for Apache**

The Netcool/ASM for Apache provides monitoring for aspects of the overall Apache server performance, including details about specific Web sites hosted by the server.

## **Netcool/ASM for IBM WebSphere**

The Netcool/ASM for IBM WebSphere contains a wide range of performance metrics for IBM WebSphere servers, derived from the WebSphere Performance Monitoring Infrastructure (PMI). It provides a flexible monitoring scheme, enabling you to monitor particular counters in a collection.

## **Netcool/ASM for WebLogic**

The Netcool/ASM for WebLogic provides performance metrics for BEA WebLogic servers. The performance metrics are available as raw bean attribute data, however you can derive composite metrics from this data using a range of built-in functions.

## **Netcool/ASM for Microsoft Active Directory**

The Netcool/ASM for Microsoft Active Directory provides facilities for monitoring the status and performance of Active Directory domain controllers, the connectivity of Active Directory entities and the contents of the Active Directory itself.

## Netcool/ASM for IBM Lotus Notes/Domino Server

The Netcool/ASM for IBM Lotus Notes/Domino Server delivers performance data about IBM Lotus Domino servers by polling metrics provided by the Lotus Domino API. It can monitor multiple Domino server instances running on one machine.

## Netcool/ASM for Sybase ASE

Netcool Application Service Monitor for Sybase Adaptive Server Enterprise (Netcool/ASM for Sybase) provides performance monitoring facilities for a range of Sybase ASE database servers.

## Netcool/ASM for Microsoft Cluster Service Control

Netcool/ASM for Microsoft Cluster Service Control (MSCS) monitors and controls MSCS server clusters. It monitors and controls the status of nodes, groups, resources, networks and network interfaces, and generate events in response to changes in status.

---

## Understanding MIB modules

Management information is viewed as a collection of managed objects, residing in a virtual information store, called the Management Information Base (MIB). Collections of related objects are defined in MIB modules. These modules are written using an adapted subset of OSI's Abstract Syntax Notation One, ASN.1 (1988).

A MIB module provides a structured set of data variables, or objects, that represent the resources to be managed. This data appears as a series of objects and tables that enable you to monitor network behavior and node data. Control rows are often used to set data monitoring parameters that control data collection and provide access to MIB functions. However, in some cases MIB scalar objects are used instead of tables for control.

MIBs are based on the Internet-standard Network Management Framework which consists of the following components:

- RFC 2578 which defines the SMI [SMIV2], the mechanisms used for describing and naming objects for the purpose of management.
- RFC 1213 which defines MIB-2, the core set of managed objects for the Internet suite of protocols.
- RFC 1905 which defines the SNMP [SNMPv2], the protocol used for network access to managed objects.

Another important component is RFC 2021 (RMON2) the network monitoring Internet-standard on which the agent is based. RMON2 is a MIB that defines objects for managing remote network monitoring devices over TCP/IP networks.

Implementation of a MIB is by one or more subagents. The subagent is a software module that performs network management functions requested by network management stations.

---

## Chapter 2. Agent

The agent MIB module provides administrative functions for the Netcool/SSM agent.

The MIB identifies, configures, and controls the Netcool/SSM master agent and its subagents. When Netcool/SSM starts, the agent MIB is called to control the overall operation of all other Netcool/SSM subagents.

The following subagents implement the agent MIB:

- agentconfig
- agentreg
- datactrl

---

### Component files

The agent MIB module and its associated subagents are comprised of a number of component files.

Table 2 lists the agent component files and their locations.

*Table 2. Agent component files*

File	Location	Description
agentconfig.dll (Windows) libagentconfig.so/.sl (UNIX)	bin	Binary implementation of the agentconfig subagent.
agentreg.dll (Windows) libagentreg.so/.sl (UNIX)	bin	Binary implementation of the agentreg subagent.
datactrl.dll (Windows) libdatactrl.so/.sl (UNIX)	bin	Binary implementation of the datactrl subagent.
agent-mib.mib	mibs	agent MIB definition document.
agentconfig.oid	config/oid	agentconfig subagent object identifier file.
agentreg.oid	config/oid	agentreg subagent object identifier file.
datactrl.oid	config/oid	datactrl subagent object identifier file.

---

### Guidelines

To use the features of the agent MIB module, ensure that you load the agentconfig, agentreg, and datacontrol subagents.

To load the subagents, execute the configuration commands:

```
subagent load agentreg
subagent load agentconfig
subagent load datactrl
```

**Tip:** In standard agent configurations, the agent configuration file, agent.cfg, already contains these commands.

## Data control

Data control enables you to toggle the monitoring state of a control row without affecting its configuration.

The data control table allows you to activate or deactivate control rows in any control table that provides a data control object (data control objects have names with the format *tableNameDataControl*).

The data control object enables data collection to be stopped without affecting any collected data. If the value of the data control object is *off(2)*, data collection pauses and data tables are 'frozen'. When the value is *on(1)*, normal data collection takes place.

To control data collection by control rows, the target control rows must be ready to start collecting data. The status objects of the target rows first must be *active(1)* for the data control objects to have any effect.

### Chaining data control

You can chain rows in the data control table together so that they trigger in sequence.

To chain data control rows, set the *dataControlNextEntry* object in each control row to the index value of the control row to be executed next.

### Regular expressions

You can switch the data control of multiple control rows by using regular expressions.

To switch the data control objects of more than one control row in the same table, set the *dataControlInstanceFilter* object to a regular expression that identifies all the data control objects that you wish to control.

For example, to specify the first three data control objects (that is, those objects with indexes 1,2 and 3) in the table *appflowControlTable*, set *dataControlObject* to *.1.3.6.1.4.1.1977.7.1.1.7.1* and set the *dataControlInstanceFilter* to *[1-3]{1}*.

To specify all data control objects in a table, use the regular expression *.\**.

**Note:** If you specify a regular expression in *dataControlInstanceFilter* you must still supply a valid OID in *dataControlObject*, however the particular instance of the data control object is not significant as it is replaced by the value of the regular expression.

**Note:** The subagent performs a string match on the regular expression specified in *dataControlInstanceFilter*. To match a substring, use the format *\*expression.\**, where *expression* denotes the regular expression for the substring.

## Specifying data control object instance filters

Data control instance filters identify particular instances of data control objects.

The `dataControlInstanceFilter` object specifies the particular instance of the data control object to be switched by a data control row using the following rules:

- When the value of `dataControlFilterMode` is `regex(1)` then `dataControlInstanceFilter` is interpreted as a regular expression and applied to the last sub-OID of the OID defined by `dataControlObject`.
- When the value of `dataControlFilterMode` is `varbindOID(2)` then `dataControlInstanceFilter` is interpreted as a control string indicating how to extract an OID from one of the varbinds associated with the events specified by `dataControlTurnOffEventIndex` and `dataControlTurnOnEventIndex`.

This control string has the following format:

*varbind, index, length*

- *varbind* indicates the varbind from which the OID is extracted.
- *index* specifies the sub-OID from which extraction commences.
- *length* determines how many sub-OIDs are extracted and in which direction they are extracted. Its operation is summarized in Table 3.

Table 3. Operation of the length parameter

Length	Sub-OIDs extracted
<i>+n</i>	<i>[index..index+n]</i>
<i>-n</i>	<i>[index-n..index]</i>
<i>+</i>	<i>[index..]</i> (that is, the entire OID string commencing from <i>index</i> )
<i>-</i>	<i>[0..index]</i> (that is, the entire OID string up to the sub-OID indicated by <i>index</i> )

## Extracting OIDs from variable bindings

1,1,+ extracts the entire OID of the first varbind.

2,2,2 extracts the second and third sub-OIDs of the second varbind.

3,7,-1 extracts the sixth and seventh sub-OIDs of the third varbind.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Agent configuration table

The agentconfig commands create rows in the agent configuration table.

The general syntax of these commands is:

```
agentconfig property=value
agentconfig create [property=value ...]
agentconfig reset
```

Table 4 lists the properties supported in these commands.

Table 4. Configuration command parameters - agentconfig

Property	Type	Description	Sets MIB object
control	enum	Controls execution of the configuration file:  active - Executes the configuration file. This is equivalent to executing the configuration command config execute <i>filename</i> .  inactive - Clears any objects set by this configuration file.	Control
file	string	The filename of the configuration file to be executed.	FileName

## Agent configuration variables

The agentconfigvar commands create rows in the agent configuration variable table and associate them with the next row created in the agent configuration table using the agentconfig command.

The general syntax of these commands is:

```
agentconfigvar property=value
agentconfigvar store [property=value ...]
agentconfigvar reset
```

Table 5 lists the properties supported in these commands.

Table 5. Configuration command parameters - agentconfigvar

Property	Type	Description	Sets MIB object
name	string	The name of the configuration variable.	Name
value	string	The value of the variable.	Value

## Data control table

The datactrl commands create rows in the data control table.

The general syntax of these commands is:

```
datactrl property=value
datactrl create [property=value ...]
datactrl reset
```

Table 6 on page 9 lists the properties supported in these commands.

Table 6. Configuration command parameters - *datactrl*

Property	Type	Description	Sets MIB object
delay	int	A time delay (in ticks) after which the next data control entry is triggered.	NextEntryDelay
filter	string	Specifies either a regular expression or a control string that selects the instance of the data control object operated on.	InstanceFilter
filtermode	enum	Selects the action taken when this event is generated. The allowed values are:  regex - Activates the configuration file  varbindOID - Clears any objects set by this configuration file	FilterMode
next	int	The value of the dataControlIndex object of the next data control table row to turn on or off.	NextEntry
object	OID	The OID of the data control object operated on.	Object
offevent	int	The RMON event index of the event that switches data control object off.	TurnOffEventIndex
onevent	int	Then RMON event index of the event that switches data control object on.	TurnOnEventIndex

## Notification destinations

The agentconfig commands create notification destinations.

The general syntax of these commands is:

```
agenttrapdest property=value
agenttrapdest create [property=value ...]
agenttrapdest reset
```

Table 7 lists the properties supported in these commands.

Table 7. Configuration command parameters - *agenttrapdest*

Property	Type	Description	Sets MIB object
address	string (dec)	The destination IP address and port number of the destination in network-byte order. That is, a decimal string of format a.b.c.d.e.f where a.b.c.d represents the IP address of the trap destination and .e.f represents the destination port encoded as .(port div 256).(port mod 256).	trapDestAddress agentTrapDestAddress
community	string	The destination community of the notification.	trapDestCommunity agentTrapDestCommunity



Table 7. Configuration command parameters - agenttrapdest (continued)

Property	Type	Description	Sets MIB object
transport	enum	Selects the transport protocol used in sending the notification. The allowed values are:  tcp - not supported. To send SNMPv2 Traps over TCP, use the SNMPv3 message processing subsystem.  udp - UDP	agentTrapDestTransport
version	enum	Selects the type of notification to be sent:  1 - SNMPv1 Trap  2 - SNMPv2 Trap	agentTrapDestVersion

## Examples

The following example demonstrates how to use the dataControlTable to control the operation of a monitor configured using the logmonx subagent.

### Monitoring log file monitors

Monitor the operation of the logmonx log file monitors. If any monitor generates an excessive number of matches (more than 100 per minute), disable it by switching off its data control object, logMonXControlDataControl.

The configuration commands for implementing this monitor are:

```
subagent load rmonc
subagent load genalarm
subagent load datacontrol

event reset
event type=snmp-trapevent description="logMonX monitor data control"event create
evtIdx=?

genalarm reset
genalarm var=$logMonXStatsMatches.*
genalarm type=delta
genalarm risethresh=100
genalarm interval=60
genalarm riseevent=$evtIdx
genalarmvb reset
genalarmvb oid=$logMonXStatsMatches.*
genalarmvb store
genalarm create

datactrl reset
datactrl object=$logMonXControlDataControl
datactrl filtermode=varbindOID
datactrl filter="11,12,1"
datactrl offevent=$evtIdx
datactrl create
```

**Tip:** The dataControlInstanceFilter filter string 11,12,1 extracts the twelfth sub-OID from the eleventh varbind (the logMonXStatsMatches object) contained in the genalarm event. This sub-OID corresponds to the row index of the logmonx

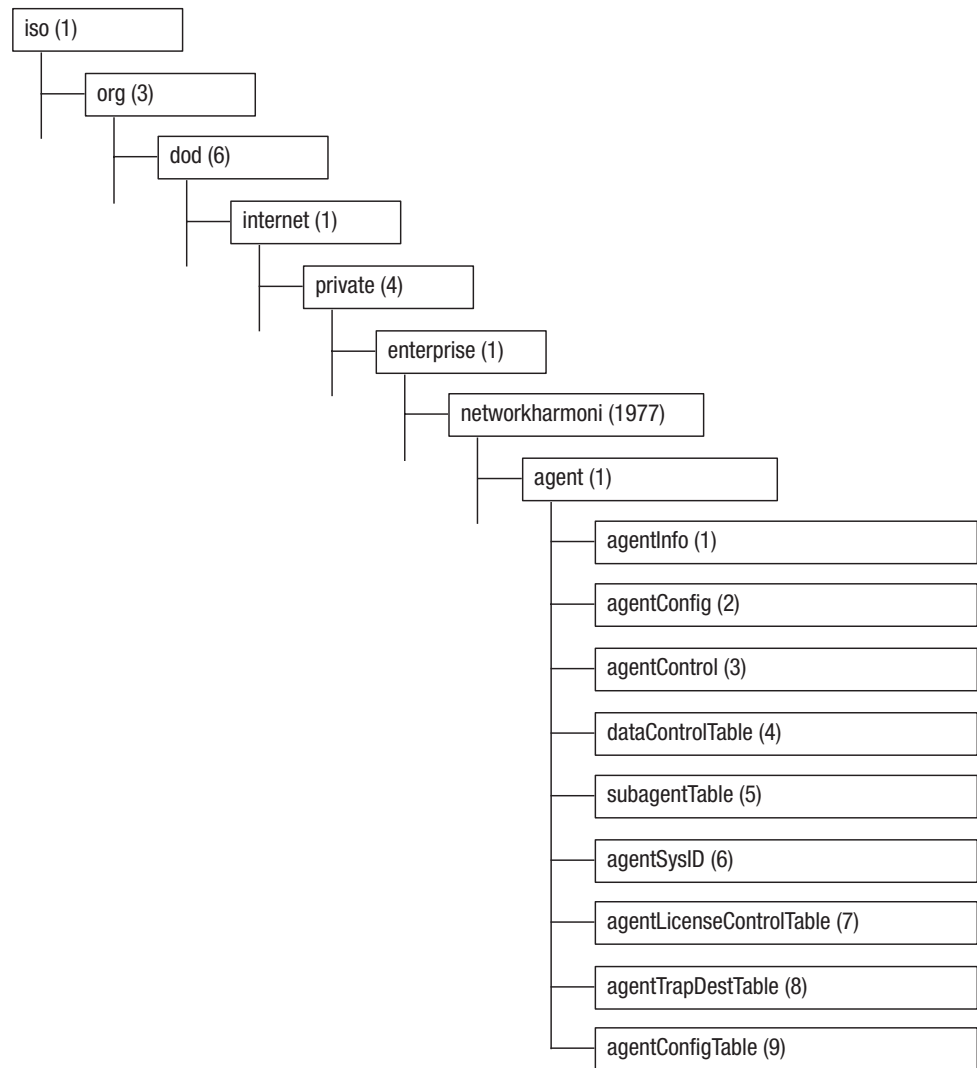
control row that generated the excessive number of matches.

---

## MIB module

The agent MIB is a subtree of `networkharmoni (1977)`.

The location of the agent subtree is shown in Figure 1.



*Figure 1. OID tree diagram of the agent MIB module*

This section provides a summary of the objects defined in the agent MIB module. For detailed information on all objects in the module, see the `agent-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB objects

The MIB module provides a standard SNMP interface.

The module contains the following objects:

- The agent information group, agentInfo
- The agent authorization group, agentAuth
- The agent configuration group, agentConfig
- The agent control group, agentControl
- The data control table, dataControlTable
- The subagent table, subagentTable
- The agent system identifier, agentSysID
- The agent license control table, agentLicenseControlTable
- The agent trap destination table, agentTrapDestTable
- The agent configuration table, agentConfigTable

### Agent information group

The agent information group (agentInfo) contains information on the agent build, agent identity, operating platform in use, location and geographical coordinates.

This group contains the following object groups:

- The agent build group (agentBuild)
- The agent identification group (agentId)
- The agent location group (agentLocation)

### Agent build group

Table 8 lists the objects in the agent build group.

*Table 8. Agent build group (agentBuild)*

Object	Description
agentBuildDate	The agent build date.
agentBuildDescription	A description of the agent build.
agentBuildNumber	The agent build number.
agentBuildVersion	The version of the agent build in the format <i>major-ver.minor-ver.service-ver</i> .

### Agent identification group

Table 9 lists the objects in the agent identification group.

*Table 9. Agent identification group (agentId)*

Object	Description
agentIdAddress	The primary IP address of the node on which the agent is running.
agentIdName	The primary host name of the node on which the agent is running.
agentIdPlatform	The operating system on which the agent is running.
agentIdPlatformDescription	A description of the operating system on which the agent is running.

Table 9. Agent identification group (agentId) (continued)

Object	Description
agentIdUniqueId	The agent's unique identifier.

## Agent location group

Table 10 lists the objects in the agent location group.

Table 10. Agent location group (agentLocation)

Object	Description
agentLocationDescr	Physical location of the host on which the agent is running.
agentLocationLatitude	Geographical location of the host on which the agent is running.
agentLocationLongitude	Geographical location of the host on which the agent is running.

## Agent authorization group

The agent authorization group provides objects containing agent authorization details.

Table 11 lists these objects.

Table 11. Agent authorization group (agentAuth)

Object	Description
agentAuthLastBadIPAddress	The IP address of the last unauthorized client.

## Agent configuration group

The agent configuration group (agentConfig) contains objects that control boot behavior, packet monitoring mode, heart beat interval and store the agent's current date and time.

Table 12 lists the objects in this group. Unless otherwise specified, changes to these objects only take effect after agent reboot.

Table 12. Agent configuration group objects

Object	Description
BootMode	Determines the agent's behavior on shutdown and when booting:  default(1) - The agent does not save its state information and on the next boot will only perform default initialization.  restore(2) - The agent saves its state information before shutting down and uses this information to its state when it next boots.
DateTime	Sets the agent's current date and time. Modifying this value may affect other objects. Restart the agent after setting this object.
HeartBeatInterval	Sets the interval for the agentHeartBeat trap. The value assigned to this object is effective immediately.

Table 12. Agent configuration group objects (continued)

Object	Description
IfMode	Sets the agent's default packet observation mode:  Promiscuous(1) - All packets on the shared media are captured.  Local(2) - All packets to/from the host that the agent is running on, including broadcasts and multicasts.  Directed(3) - All packets to/from the host the agent is running on, excluding broadcasts and multicasts.
NotificationCommunity	The community to which all agentConfig notifications are sent. The value assigned to this object is effective immediately.
TrapsOn	The master control for trap operation:  on(1) - Trap generation is enabled.  off(2) - Trap generation is disabled. The agent will not send any traps, even if other subagents, MIBs, or objects are configured to do so.

## Agent control group

The agent control group contains objects for rebooting the agent, saving current configuration files and triggering events.

Table 13 lists the objects in this group.

Table 13. Agent control group objects

Object	Description
PulseEvent	Holds the event index of the last event pulsed. Setting this object to a non-zero value will cause an event to be triggered. Any actions associated with that event will be executed (for example, turning on an RMON1 channel row). If the event does not exist, then the set will fail with the error inconsistentValue. Setting this object to zero (0) will succeed, but no event will be triggered.
Reboot	Setting the value of this object to reboot(2) reboots the agent.
SaveConfig	Writing any value to this object saves the agent's current configuration to its default configuration files. The agent then uses this configuration when it next boots.

## Data control table

The data control table dataControlTable provides a facility for event-based control of data collection in subagent control rows that provide a data control object (with name tableNameDataControl).

Each row in the data control table identifies a data control object to be switched on or off in response to a 'turn on' or 'turn off' event. Table 14 on page 15 lists the row objects in dataControlTable.

The dataControlNextEntry and dataControlNextEntryDelay objects allow a number of data control table rows to be chained together so that data control objects can be switched in sequence with a specific delay between each.

By specifying a regular expression in `dataControlInstanceFilter`, you can control multiple data control objects in a single table simultaneously.

*Table 14. Subagent data control table*

Row object	Description
FilterMode	Specifies the operation of the <code>dataControlInstanceFilter</code> object:  <code>regex(1)</code> - <code>dataControlFilterMode</code> is interpreted as a regular expression that is applied to the last sub-OID in the OID specified by <code>dataControlObject</code> .  <code>varbindOID(2)</code> - <code>dataControlFilterMode</code> is interpreted as a control string specifying how to extract an OID string from the OID of one of the varbinds contained in the events specified by <code>dataControlTurnOffEventIndex</code> and <code>dataControlTurnOnEventIndex</code> .  See “Specifying data control object instance filters” on page 7 for more details about using the <code>dataControlInstanceFilter</code> and <code>dataControlFilterMode</code> objects.
Index	Uniquely identifies a row in the table.
InstanceFilter	Specifies either a regular expression or a control string that selects the instance of the data control object identified by <code>dataControlObject</code> .  See “Specifying data control object instance filters” on page 7 for more details about using the <code>dataControlInstanceFilter</code> and <code>dataControlFilterMode</code> objects.
NextEntry	The value of the <code>dataControlIndex</code> object of the next data control table row to turn on or off.
NextEntryDelay	Sets the delay (in time ticks) between switching the data control of the entry defined in <code>dataControlNextEntry</code> object.
Object	The OID of the data control object that this control row operates on.
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Status	The status of this entry. When the value of this object is not <code>active(1)</code> , the value of <code>dataControlTriggerCount</code> is set to zero.
TriggerCount	The total number of times that the data control has been turned on or off.
TriggerTime	The value of <code>sysUpTime</code> when the data control was last turned on or off.
TurnOffEventIndex	The RMON event index of the event that switches data control object off.
TurnOnEventIndex	The RMON event index of the event that switches data control object on.

## Subagent control table

The subagent control table (subAgentTable) provides a list of subagents available for loading.

Table 15 lists the objects contained in this table.

Table 15. Subagent control table

Row object	Description
BuildDate	The date that the subagent was built.
BuildNumber	A label representing the build number of the subagent.
Description	A short description of the MIB that the subagent implements.
Name	The name of the subagent. Use this name to obtain the name of module that implements the subagent as follows: On UNIX systems, add the prefix lib and the suffix .so to the value of this object. On Windows systems, append the suffix .dll to the value of this object. The module identified by this object is loaded when the corresponding subagentStatus object is set to active(1).
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Status	<p>The status of this subagent entry:</p> <p>active(1) loads the subagent. When the status is active(1), and is set to any value other than active(1), the subagent will be signalled to release any resources it may have been allocated, and is then unloaded from memory.</p> <p>The status automatically changes from notReady(3) to notInService(2) only if the subagentName object represents a potentially valid subagent implementation.</p>
Vendor	Name of the subagent's vendor.

## Agent system ID

The agent system ID is the root node for a collection of identification objects.

This object is a sub-OID that is used to form sysObjId in MIB-II. The format for this object is agentSysId.xxxx.n, where xxxx refers to the vendor, and n denotes the product.

## Trap destination table

The trap destination table (agentTrapDestTable) augments the RMON2 trap destination table.

Each entry in the table also appears in the RMON2 trap destination table (trapDestTable). For implementation reasons the two tables are coupled only by description rather than by definition. Table 16 lists the row objects in agentTrapDestTable.

Table 16. Trap destination table

Row object	Description
Address	This object has the same format and semantics as trapDestAddress in the RMON2 Trap Destination Table.
Community	This object has the same format and semantics as trapDestCommunity in the RMON2 Trap Destination Table.



Table 16. Trap destination table (continued)

Row object	Description
Index	This object corresponds exactly to the trapDestIndex in RMON2.
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Status	The status of this trap destination entry. An entry may not exist in the active state unless all objects in the entry have an appropriate value.
Transport	This object is no longer supported. The transport mapping is always UDP.
Version	Selects the type of notification sent:  snmpv1(1) - SNMPv1 Trap  snmpv2(2) - SNMPv2 Trap

## Agent configuration table

The agent configuration table (agentConfigTable) is used by the master agent to execute and manage configuration files.

Each row in the table represents one configuration file loaded on the agent. When the agentConfigControl field is set to active(1) and the row is activated, the associated configuration file is parsed by the agent and the contents executed by the relevant subagent.

All of the control rows created as a result of activating a configuration file have the owner field set to the value of the agentConfigFileName object. This owner field identifies the configured control rows with the particular row in the agentConfigTable, so that when the row is deactivated either by setting the agentConfigControl to deactivate(2) or deactivating the entire row, all the corresponding control rows are automatically destroyed, enabling removal of entire configurations with a single operation.

Table 17 lists the row objects in agentConfigTable.

Table 17. Agent configuration table

Row object	Description
ActivateTime	The value of sysUpTime.0 when the value of agentConfigControl was last set to active(1).
Control	When this object is set to the value inactive(2), an attempt will be made to clear any objects configured by this entry. This is best facilitated using the high level MIB configuration commands, rather than the primitive snmp set.
CreateTime	The value of sysUpTime.0 when this row was last activated.
FileName	The configuration file (console script) for this configuration entry. When the value of agentConfigControl is set to active(1), this file will be executed as if invoked by the administrator console as config execute filename.
Index	Uniquely identifies this row in the table.
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Status	The status of this agent configuration entry. An entry may not exist in the active state unless all objects in the entry have an appropriate value.

Table 17. Agent configuration table (continued)

Row object	Description
Variables	The number of rows in the agentConfigVarTable allocated to this control row. These rows are indexed contiguously starting from 1.

## Agent configuration variable table

The agent configuration table (agentConfigVarTable) stores the variables associated with each configuration file listed in agentConfigTable.

Each table row represents one variable, and is indexed by an agent configuration table row. These variables are instantiated by the agent when the corresponding configuration file is executed. Table 18 lists the row objects in agentConfigTable.

Table 18. Agent configuration variable table

Row object	Description
Name	Matches variables in the corresponding configuration file. If a name is specified twice for the same agentConfigIndex, then the entry with the greatest agentConfigVarIndex is used.
Value	The value which is substituted for any variable with the name indicated by agentConfigVarName.

## Agent inivar table

The agent inivar table (agentInivarTable) stores the values of initialization variables (*inivars*) defined on the agent.

Each row in the table represents an inivar name-value pair. Table 19 lists the row objects in agentInivarTable.

Table 19. Agent inivar table

Row object	Description
Name	The name of the inivar.
Value	The value of the inivar.

## Notification types

The agent MIB module defines several notification types.

These notification types are for events generated by the agentreg and agentconfig subagents. These types are shown in Figure 2 on page 19 and listed in Table 20 on page 19.

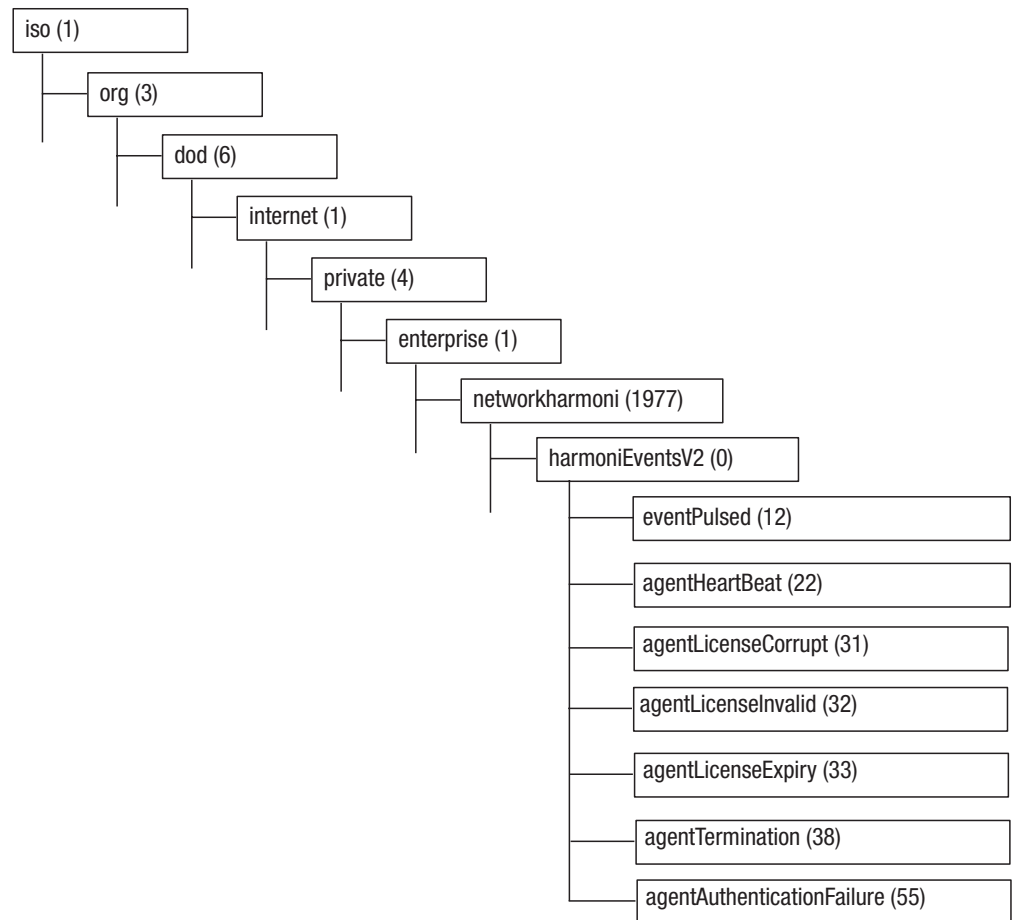


Figure 2. OID tree diagram for agent notification types

Table 20. Notification types for agent MIB

Notification type	Description
agentAuthenticationFailure	Generated whenever an unknown community string is used to attempt access with the agent. The target community of the notification is determined by agentConfigNotificationCommunity.  Variable bindings: agentAuthLastBadIPAddress
agentHeartBeat	Generated on a heartbeat event. The target community of the notification is determined by agentConfigNotificationCommunity.  Variable bindings: agentIdUniqueId agentIdAddress agentIdName ifPhysAddress

Table 20. Notification types for agent MIB (continued)

Notification type	Description
agentTermination	Generated when the agent process is about to terminate.  Variable bindings:  agentIdUniqueId  agentIdAddress  agentIdName  ifPhysAddress
eventPulsed	Generated whenever the agentControlPulseEvent object is successfully set to a non-zero value.  Variable bindings:  agentControlPulseEvent

---

## Chapter 3. Application flow

The appflow subagent measures information on application connections, called flows. The metrics gathered include connection start-time, setup-time, duration, teardown time, inter-packet arrival times, and retransmissions— information useful for application policy management, quality of service (by applications such as voice and video) monitoring and capacity planning.

The appflow subagent and its associated appflow MIB provide facilities for monitoring connection-oriented (TCP) sessions. The metrics provided by the MIB include connection start, setup, duration and teardown times as well as interpacket arrival and retransmission times.

The type of information provided by the MIB is useful in policy management, quality of service monitoring and capacity planning for applications.

---

### Component files

The appflow subagent and MIB module are comprised of a set of component files.

Table 21 lists the appflow subagent and MIB module component files.

*Table 21. Appflow component files*

File	Location	Description
appflow.dll (Windows) libappflow.so/.sl (UNIX)	bin	Binary implementation of the appflow subagent.
appflow-mib.mib	mibs	appflow MIB definition document.
appflow.oid	config/oid	appflow subagent object identifier file.

---

### Guidelines

Use the appflow subagent and MIB to monitor application connections over networks.

To load the appflow subagent, use the command:

```
subagent load appflow
```

### Client/server filtering

You can configure the subagent to monitor only selected client and server addresses.

The appflowControlClientAddr and appflowControlServerAddr objects set the monitored client and server addresses. By default, the client address is initialized to the localhost on which the subagent is running, while the server address is initialized to \*, null, or 0.0.0.0, which matches all hosts so that data will be collected for all servers.

To filter out information about connections from other hosts on the network that are not of interest, set the client and server addresses to specific IP addresses. Using specific addresses ensures that data will only be collected for those addresses.

**Note:** Network numbers are not recognized, but if you provide a name that resolves to multiple network addresses, then data will be collected for all of those addresses. Network addresses that encompass a range of host addresses are not supported.

## Protocol filtering

You can configure control rows to monitor a specific protocol or a protocol range.

To do so, set the `appflowControlMinPort` and `appflowControlMaxPort` objects to port numbers that correspond to the protocol range. The default values of these objects are 0 and 65535 respectively, which represents monitoring of all protocols. If you wish to monitor only a single protocol, set both objects to the same port value. Only connection-oriented, TCP-based protocols may be monitored.

## Event-controlled activation and deactivation

Control rows in `appflowControlTable` can be activated and deactivated by RMON events.

Set the `appflowControlTurnOnEventIndex` and `appflowControlTurnOffEventIndex` objects to the indexes of the RMON events that will activate or deactivate the control row.

## SNMPv1 compatibility

The `appflow` MIB module uses the Counter64 data type, so it is not compatible with SNMPv1.

Only use the Application Flow subagent with SNMPv2 and later.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The appflow commands create rows in the control table (appflowControlTable).

The general syntax of these commands is:

```
appflow property=value  
appflow create property=value ...  
appflow reset
```

Table 22 lists the properties supported in these commands.

Table 22. Configuration command parameters - appflow

Property	Type	Description	Sets MIB object
buckets	int	The number of summary table rows requested.	SummaryBucketsRequested
client	string	The client address.	ClientAddr
datacontrol	enum	Data control:  on - enables data collection  off - suspends data collection	DataControl
datasource	OID	The interface ID of the data source.	DataSource
description	string	A description of the control row.	Description
history	int	The history table size limit.	ConnHistMaxEntries
maxport	int	The maximum port from which data is collected.	MaxPort
minport	int	The minimum port from which data is collected.	MinPort
offevent	int	The control row deactivation event index.	TurnOffEventIndex
onevent	int	The control row activation event index.	TurnOnEventIndex
server	string	The server address.	ServerAddr
summary	int	The summary interval.	SummaryInterval
timeout	int	The connection timeout interval.	ConnTimeout

## Alarm table

The appflowalarm commands create rows in the alarm table (appflowAlarmTable).

The general syntax of these commands is:

```
appflowalarm property=value  
appflowalarm create property=value ...  
appflowalarm reset
```

Table 23 lists the properties supported in these commands.

Table 23. Configuration command parameters - appflowalarm

Property	Type	Description	Sets MIB object
appflowindex	int	The index of the associated appflow control row.	ControlIndex

Table 23. Configuration command parameters - appflowalarm (continued)

Property	Type	Description	Sets MIB object
eventstatus	enum	Event flow control:  alwaysready  fired  ready	EventStatus
fall event	int	The event index of the falling alarm event.	FallingEventIndex
fallthresh	int	The falling alarm threshold.	FallingThreshold
riseevent	int	The event index of the rising alarm event.	RisingEventIndex
risethresh	int	The rising alarm threshold.	RisingThreshold
startup	enum	The startup alarm type:  falling  rising  risingOrFalling	StartupAlarm
type	enum	The sample type.	SampleType
var	enum	The monitored metric:  duration  inAvgApplicationTime  inAvgNetworkTime  inAvgPacketArrivalDelay  inLastApplicationTime  inLastNetworkTime  inLastPacketArrivalDelay  inRetransmissions  outAvgApplicationTime  outAvgNetworkTime	Variable



Table 23. Configuration command parameters - appflowalarm (continued)

Property	Type	Description	Sets MIB object
var (continued)	enum	outAvgPacketArrivalDelay outLastApplicationTime outLastNetworkTime outLastPacketArrivalDelay outRetransmissions releaseFailure releasePercentage setupFailure setupPercentage setupTime teardownTime	Variable

## Examples

These examples demonstrate how to perform simple monitoring tasks using the appflow subagent.

### Monitoring all hosts

Monitor all application flow statistics for all hosts on the network.

```
subagent load appflow
appflow reset
appflow client=*
appflow create
```

### Monitoring telnet connections

Monitor telnet connections made by any client:

```
subagent load appflow
appflow reset
appflow client=*
appflow minport=23
appflow maxport=23
appflow create
```

**Tip:** To specify all client IP addresses, set the ClientAddr to \*, null or 0.0.0.0.

### Alarm generation

Monitor application flow statistics for all hosts on the network and generate an alarm if the setup duration of any connection, occurring only once per connection, is greater than 10 milliseconds:

```
event reset
event type=snmp-trap
event community=public
event description="Connection setup time > 10ms"
event create
```

```

violationevent=$?

subagent load appflow
appflow reset
appflow client=*
appflow create
crowindex=$?

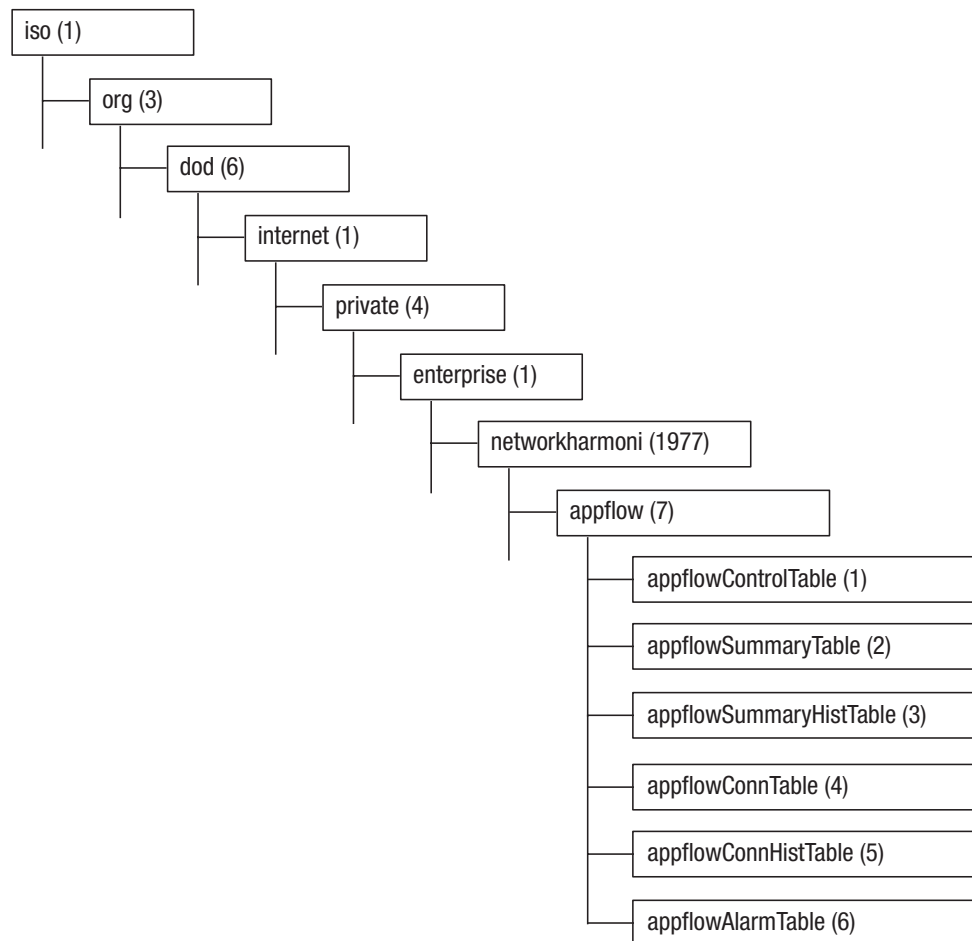
appflowalarm reset
appflowalarm appflowindex=$crowindex
appflowalarm var=setupTime
appflowalarm riseevent=$violationevent
appflowalarm risethresh=10
appflowalarm eventstatus=4
appflowalarm create

```

## MIB module

The appflow MIB is a subtree of networkharmoni (1977).

The location of the appflow subtree is shown in Figure 3.



*Figure 3. OID diagram of the appflow MIB module*

This section provides a summary of the objects defined in the appflow MIB module. For detailed information on all objects in the module, see the appflow-mib.mib document located in the mibs subdirectory of the Netcool/SSM

installation.

## MIB tables

The MIB module provides a standard SNMP interface with control row semantics.

The MIB contains the following tables:

- The control table, `appflowControlTable`
- The summary table, `appflowSummaryTable`
- The summary history table, `appflowSummaryHistTable`
- The connection table, `appflowConnTable`
- The connection history table, `appflowConnHistTable`
- The alarm table, `appflowAlarmTable`

### Control table

The control table (`appflowControlTable`) defines control parameters for monitoring application flows over networks.

Each control row specifies the client, server and protocol to be monitored as well as the amount of storage allocated in the summary and connection tables for storing the data that it generates. Table 24 lists the row objects in `appflowControlTable`.

Table 24. *appFlow control table (appFlowControlTable)*

Row object	Description
ClientAddr	Specifies the client IP name or address for which data is collected. If this object is set to * then data is collected for all clients. Network numbers are not recognized, but if the name specified by this object resolves to multiple network addresses, data will be collected for any of those addresses. This object cannot be modified if the associated <code>appflowControlStatus</code> object has the value <code>active(1)</code> . The default value of this object is the IP address of the first available interface on the host.
ConnHistMaxEntries	Sets the maximum number of entries in the <code>appflowConnHistTable</code> .
ConnHistTableSize	Sets the number of rows in <code>appflowConnHistTable</code> allocated to this control row.
ConnTableSize	Sets the number of rows in <code>appflowConnTable</code> allocated to this control row.
ConnTimeout	Sets the timeout value (in seconds) for active connections in <code>appflowConnTable</code> .
CreateTime	Stores the value of <code>sysUpTime</code> when this entry was last activated.
DataControl	Sets the data collection status of the control row: <code>on(1)</code> - Enables data collection <code>off(2)</code> - Suspends data collection
DataSource	Identifies the source of the data (that is, the interface) monitored by the control row. This source can be any interface on the device hosting the subagent. The value of this object identifies the interface by its <code>ifIndex</code> object. For example, for a control row to monitor data using interface #1, this object must be set to <code>ifIndex.1</code> .

Table 24. *appFlow control table (appFlowControlTable)* (continued)

Row object	Description
Description	A description of the application flow being monitored (containing details such as protocol and application).
Index	Uniquely identifies the control table row.
MaxPort	Defines the maximum port for which data is collected.
MinPort	Defines the minimum port for which data is collected.
Owner	Indicates the owner of the control row.
ServerAddr	Specifies the server IP name or address for which data is collected. If this object is set to * then data will be collected for all servers. Network numbers are not recognized, but if the name specified by this object resolves to multiple network addresses, data will be collected for any of those addresses. This object cannot be modified if the associated appflowControlStatus object has the value active(1). The default value of this object is * (any server).
Status	An SNMPv2 row status object controlling creation, activation and deletion of the control row.
SummaryBucketsGranted	Indicates the number of rows in appflowSummaryTable granted by the subagent to this control row.
SummaryBucketsRequested	Sets the number of rows in appflowSummaryTable requested for storing the data associated with this control row.
SummaryInterval	Sets the sampling interval (in seconds). Each time this interval elapses, rows in appflowSummaryTable are moved to appflowSummaryHistTable. If the value of this object is 0, rows in appflowSummaryTable are kept indefinitely.
TurnOffEventIndex	Contains the RMON eventIndex of the event configured to deactivate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then deactivation via an event is disabled.
TurnOnEventIndex	Contains the RMON eventIndex of the event configured to activate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then activation via an event is disabled.

## Summary table

The summary table (appflowSummaryTable) provides summary statistics for the connections monitored by appflowControlTable rows.

Statistics consist of counters for successful and unsuccessful setups and successful and unsuccessful releases. Summary table rows are associated with a control rows on a one-to-one basis. Table 25 lists the row objects in appflowSummaryTable.

Table 25. *Application flow summary table (appflowSummaryTable)*

Row object	Description
ReleaseFailure	The total number of connections released unsuccessfully, where the TCP session was aborted by TCP_RST.
ReleasePercentage	The number of connections that were released successfully, expressed as a percentage of all connections released.

Table 25. Application flow summary table (*appflowSummaryTable*) (continued)

Row object	Description
ReleaseSuccess	The total number of connections that were released successfully.
SetupFailure	The total number of connections that were not established successfully, where the TCP setup was aborted by TCP_RST.
SetupPercentage	The number of connections established successfully, expressed as a percentage of all connections established.
SetupSuccess	The total number of connections established successfully.

## Summary history table

The *appflowSummaryHistTable* contains statistics copied at regular intervals from the summary table.

The interval at which these statistics are copied is set by the corresponding control row's *appflowControlSummaryInterval* object. The number of rows in the table associated with a control row is set by the control row's *appflowControlSummaryBucketsRequested* object. Table 26 lists the row objects in *appflowSummaryHistTable*.

Table 26. Application flow summary history table (*appflowSummaryHistTable*)

Row object	Description
IntervalStart	The time at which the history interval started.
ReleaseFailure	The total number of connections released unsuccessfully.
ReleasePercentage	The number of connections that were released successfully, expressed as a percentage of all connections released.
ReleaseSuccess	The total number of connections that were released successfully.
SampleIndex	Identifies the table row.
SetupFailure	The total number of connections established unsuccessfully.
SetupPercentage	The number of connections established successfully, expressed as a percentage of all connections established.
SetupSuccess	The total number of connections established successfully.

## Connection table

The connection table (*appflowConnTable*) provides statistics on currently active connections observed on the network.

The number of rows in the table allocated to a control row is set by the control row's *appflowControlConnTableSize* object.

Rows in the connection table are indexed by a combination of the objects:

- *appflowConnServerPort*
- *appflowConnServerAddr*
- *appflowConnClientPort*
- *appflowConnClientAddr*

Table 27 on page 30 lists the row objects in *appflowConnTable*.

Table 27. Application flow connection table (appflowConnTable)

Row object	Description
AppName	The application name, resolved from client or server port address.
ClientAddr	The client IP address. The client is the initiator of the connection request.
ClientPort	The client's port.
Duration	The duration of the session connection (in milliseconds).
InAvgApplicationTime	The average application time from server to client (in milliseconds).
InAvgNetworkTime	The average network time from server to client (in milliseconds).
InAvgPacketArrivalDelay	The average packet inter-arrival time from server to client (in milliseconds).
InBestApplicationTime	The best application time from server to client (in milliseconds).
InBestNetworkTime	The best network time from server to client (in milliseconds).
InBestPacketArrivalDelay	The best packet inter-arrival time from server to client (in milliseconds).
InInvalidTransactions	The number of invalid transactions from server to client due to packets dropped by the agent.
InLastApplicationTime	The last application time from server to client (in milliseconds).
InLastNetworkTime	The last network time from server to client (in milliseconds).
InLastPacketArrivalDelay	The last packet inter-arrival time from server to client (in milliseconds).
InOctets	The number of octets from server to client.
InPkts	The number of packets from server to client.
InRetransmissions	The number of packet retransmissions from server to client.
InValidTransactions	The number of valid transactions from server to client.
InWorstApplicationTime	The worst application time from server to client (in milliseconds).
InWorstNetworkTime	The worst network time from server to client (in milliseconds).
InWorstPacketArrivalDelay	The worst packet inter-arrival time from server to client (in milliseconds).
OutAvgApplicationTime	The average application time from client to server (in milliseconds).
OutAvgNetworkTime	The average network time from client to server (in milliseconds).
OutAvgPacketArrivalDelay	The average packet inter-arrival time from client to server (in milliseconds).
OutBestApplicationTime	The best application time from client to server (in milliseconds).

Table 27. Application flow connection table (appflowConnTable) (continued)

Row object	Description
OutBestNetworkTime	The best network time from client to server (in milliseconds).
OutBestPacketArrivalDelay	The best packet inter-arrival time from client to server (in milliseconds).
OutInvalidTransactions	The number of invalid transactions from client to server due to packets dropped by the agent.
OutLastApplicationTime	The last application time from client to server (in milliseconds).
OutLastNetworkTime	The last network time from client to server (in milliseconds).
OutLastPacketArrivalDelay	The last packet inter-arrival time from client to server (in milliseconds).
OutOctets	The number of octets from client to server.
OutPkts	The number of packets from client to server.
OutRetransmissions	The number of packet retransmissions from client to server.
OutValidTransactions	The number of valid transactions from client to server.
OutWorstApplicationTime	The worst application time from client to server (in milliseconds).
OutWorstNetworkTime	The worst network time from client to server (in milliseconds).
OutWorstPacketArrivalDelay	The worst packet inter-arrival time from client to server (in milliseconds).
ServerAddr	The server IP address. The server is the receiver of the connection request.
ServerPort	The server's port.
SetupTime	The duration of the connection setup (in milliseconds).
StartTime	The value of sysUpTime when connection started.
Status	The current status of the connection:  setup(1)  active(2)  closing(3)  closed(4)  attempted(5)  aborted(6)  expired(7)
TeardownTime	The duration of the connection release (in milliseconds).

## Connection history table

The `appflowConnHistTable` contains statistics about connections that have closed, aborted or expired.

Rows in the `appflowConnTable`, which provide statistics on connections that are currently active, are moved to the `appflowConnHistTable` once the connection closes, aborts or expires.

The connection history table essentially maintains a log of connections observed on the network according to the parameters defined in each control row. The number of rows in `appflowConnHistTable` associated with a control row is defined by the control row's `appflowControlConnHistMaxEntries` object.

Rows in the connection history table are indexed by a combination of the objects:

- `appflowConnHistCloseTime`
- `appflowConnHistServerPort`
- `appflowConnHistServerAddr`
- `appflowConnHistClientPort`
- `appflowConnHistClientAddr`

The statistics provided by the connection history table are identical to those in the connection table.

## Alarm table

The application flow alarm table (`appflowAlarmTable`) provides a mechanism for generating events based on statistics in the summary table `appflowSummaryTable` and the connection table `appflowConnTable`.

Each row specifies an object in the summary or history table to be monitored, defines thresholds for that object and indicates the events to be generated if those thresholds are violated. Table 28 lists the row objects in `appflowAlarmTable`.

*Table 28. Application flow alarm table (`appflowAlarmTable`)*

Row object	Description
ControlIndex	Specifies the <code>appflowControlIndex</code> object of the monitored control row.
EventStatus	Sets the event flow control for the events associated with this row:  eventReady(1) - A single event may be generated, after which the value of this object changes to eventFired(2).  eventFired(2) - Disables event generation. No events may be generated until the object is modified to eventReady(1) or eventAlwaysReady(3).  eventAlwaysReady(3) - Disables the flow control, allowing events to be generated without restriction. Using this setting is not recommended as it can result in high network traffic or other performance problems.  eventConnection(4) - Limits event generation to a single event per connection session. This value is not permitted when monitoring variables in <code>appflowSummaryTable</code> .



Table 28. Application flow alarm table (appflowAlarmTable) (continued)

Row object	Description
FallingEventIndex	<p>Contains the RMON eventIndex of the event generated if the sample value of the monitored object crosses the falling threshold.</p> <p>If this index does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then no event is generated.</p>
FallingThreshold	<p>Specifies the falling threshold for the sampled value of the object specified by appflowAlarmVariable.</p> <p>When the sampled value is less than or equal to this threshold and the value of the previous sample was greater than this threshold, the event specified by appflowAlarmFallingEventIndex is generated.</p> <p>This event is also generated if the first sample taken after this row becomes valid is less than or equal to the rising threshold and the value of the appflowAlarmStartupAlarm object is equal to fallingAlarm(2) or risingOrFallingAlarm(3).</p> <p>After a falling event is generated, another such event cannot be generated until the sample value has risen above this threshold and reached the rising threshold.</p>
Index	Uniquely identifies the row.
Owner	The owner of this row.
RisingEventIndex	<p>Contains the RMON eventIndex of the event generated if the sample value of the monitored object crosses the rising threshold.</p> <p>If this index does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then no event is generated.</p>
RisingThreshold	<p>Specifies how the rising threshold for the sampled value of the object specified by appflowAlarmVariable.</p> <p>When the sampled value is greater than or equal to this threshold and the value of the previous sample was less than this threshold, the event specified by appflowAlarmRisingEventIndex is generated.</p> <p>This event is also generated if the first sample taken after the row becomes valid is greater than or equal to the rising threshold and the value of the appflowAlarmStartupAlarm object is equal to risingAlarm(1) or risingOrFallingAlarm(3).</p> <p>After a rising event is generated, another such event cannot be generated until the sample value has fallen below this threshold and reached the falling threshold.</p>
SampleType	<p>Specifies the how the value of the object specified by appflowAlarmVariable is sampled and compared to the thresholds:</p> <p>absoluteValue(1) - The value of the object is compared directly with the thresholds.</p> <p>deltaValue(2) - The value of the object at the previous sample is subtracted from the current value of the object then compared with the thresholds.</p>

Table 28. Application flow alarm table (appflowAlarmTable) (continued)

Row object	Description
StartupAlarm	<p>Specifies the alarm activation behavior when the status of this row is first set to valid.</p> <p>If the first sample taken after this row becomes valid is greater than or equal to the rising threshold and the value of this object is equal to risingAlarm(1) or risingOrFallingAlarm(3), then a single rising alarm is generated.</p> <p>If the first sample taken after this row becomes valid is less than or equal to the falling threshold and the value of this object is equal to fallingAlarm(2) or risingOrFallingAlarm(3), then a single falling alarm is generated.</p>
Status	The status of this row.
Variable	<p>Specifies the object monitored, from the appflowSummaryTable or appflowConnTable:</p> <p>setupFailure(1)</p> <p>setupPercentage(2)</p> <p>releaseFailure(3)</p> <p>releasePercentage(4)</p> <p>duration(5)</p> <p>setupTime(6)</p> <p>teardownTime(7)</p> <p>inLastNetworkTime(8)</p> <p>inAvgNetworkTime(9)</p> <p>inLastApplicationTime(10)</p> <p>inAvgApplicationTime(11)</p> <p>inLastPacketArrivalDelay(12)</p> <p>inAvgPacketArrivalDelay(13)</p> <p>inRetransmissions(14)</p> <p>outLastNetworkTime(15)</p> <p>outAvgNetworkTime(16)</p> <p>outLastApplicationTime(17)</p> <p>outAvgApplicationTime(18)</p> <p>outLastPacketArrivalDelay(19)</p> <p>outAvgPacketArrivalDelay(20)</p> <p>outRetransmissions(21)</p>

## Notification types

The appflow MIB module defines several notification types for events generated by the subagent.

The notification types are shown in Figure 4.

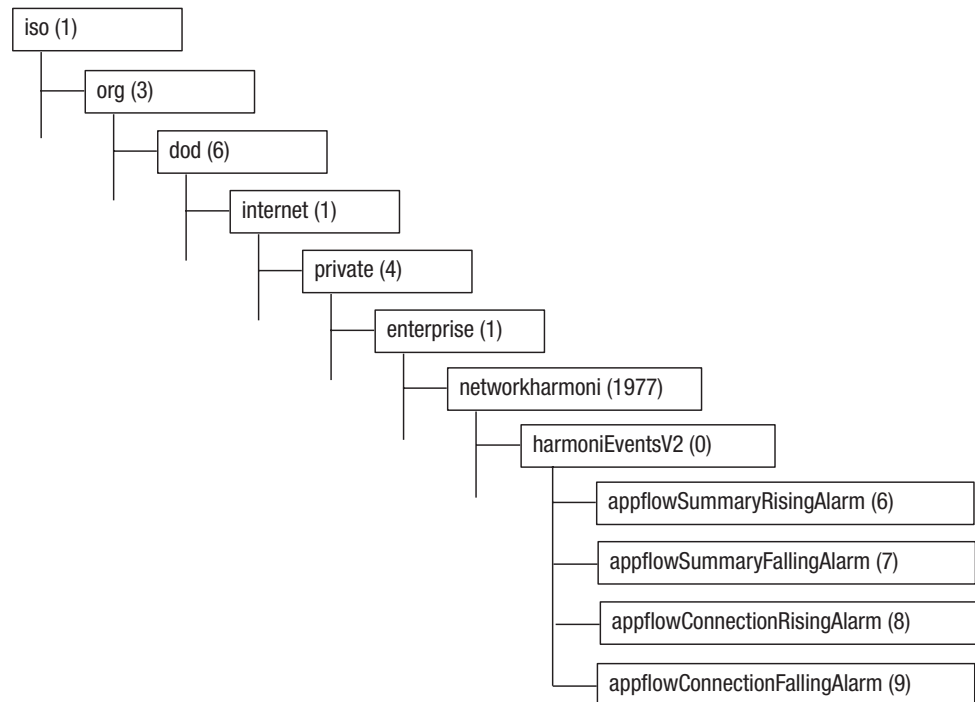


Figure 4. OID tree diagram for appFlow notification types

Table 29 lists the appflow notification types and their variable bindings.

Table 29. Notification types for appFlow MIB

Notification type	Description
appflowConnectionFallingAlarm	<p>Generated when the falling threshold defined in an appflowAlarmTable row for an object in appflowConnTable is violated.</p> <p>Variable bindings:</p> <p>appflowAlarmVariable</p> <p>appflowAlarmControlIndex</p> <p>appflowConnClientAddr</p> <p>appflowConnClientPort</p> <p>appflowConnServerAddr</p> <p>appflowConnServerPort</p> <p>appflowConnStatus</p>

Table 29. Notification types for appFlow MIB (continued)

Notification type	Description
appflowConnectionRisingAlarm	<p>Generated when the rising threshold defined in an appflowAlarmTable row for an object in appflowConnTable is violated.</p> <p>Variable bindings:</p> <p>appflowAlarmVariable</p> <p>appflowAlarmControlIndex</p> <p>appflowConnClientAddr</p> <p>appflowConnClientPort</p> <p>appflowConnServerAddr</p> <p>appflowConnServerPort</p> <p>appflowConnStatus</p>
appflowSummaryFallingAlarm	<p>Generated when the falling threshold defined in an appflowAlarmTable row for an object in appflowSummaryTable is violated.</p> <p>Variable bindings:</p> <p>appflowAlarmVariable</p> <p>appflowAlarmControlIndex</p> <p>appflowSummarySetupPercentage</p> <p>appflowSummaryReleasePercentage</p>
appflowSummaryRisingAlarm	<p>Generated when the rising threshold defined in an appflowAlarmTable row for an object in appflowSummaryTable is violated.</p> <p>Variable bindings:</p> <p>appflowAlarmVariable</p> <p>appflowAlarmControlIndex</p> <p>appflowSummarySetupPercentage</p> <p>appflowSummaryReleasePercentage</p>

---

## Chapter 4. Application usage

The appusage subagent and the associated appusage MIB provide facilities for monitoring applications that are currently running, or loaded, on the Windows desktop.

The MIB provides information such as CPU utilization and active use of applications running on the host machine.

The appusage subagent adapts to site configuration changes, discovers newly executed applications, and removes data about applications that are no longer being used.

---

### Component files

The appusage subagent and MIB module are comprised of a set of component files.

Table 30 lists the appusage subagent and MIB module component files and their installed locations.

*Table 30. Appusage component files*

File	Location	Description
appusage.dll (Windows) libappusage.so/.sl (UNIX)	bin	Binary implementation of the appusage subagent.
appusage-mib.mib	mibs	appUsage MIB definition document.
appusage.oid	config/oid	appusage subagent object identifier file.

---

### Guidelines

Use the appusage subagent to monitor the activity of individual users.

The information that appusage generates is useful, for example, to determine how many copies of a particular software application to buy or how much time is spent on particular activities.

To load the subagent, use the command:

```
subagent load appusage
```

### Inivars

The appusage subagent provides a single inivar for configuring its operation.

The appusage subagent inivar is listed in Table 31 on page 38.

Table 31. appusage subagent Inivars

Inivar	Type	Description
AppUsageMaxWindowSize	int	Sets the maximum allowed value of the appusageControlWindowSize object, which determines the length of time (in seconds) for which usage data is retained.  Default value: 86400 (24 hours).

For general information about inivars, see the *Netcool/SSM Administration Guide*.

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The appusage commands create control rows in the application usage control table (appUsageControlTable).

The general syntax of these commands is:

```
appusage property=value
appusage create property=value ...
appusage reset
```

Table 32 lists the properties supported in these commands.

Table 32. Configuration command parameters - appusage

Property	Type	Description	Sets MIB object
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl
description	string	A description of the control row.	Description
windowSize	int	The sampling window size (in seconds).	WindowSize

---

## Examples

This example demonstrates how to perform simple monitoring tasks using the appusage subagent.

### Monitoring unused applications

Monitor application usage, discarding any data about applications that go unused for more than one day:

```
subagent load appusage
appusage reset
appusage description="Example application monitor"
appusage windowsize=86400
appusage create
```

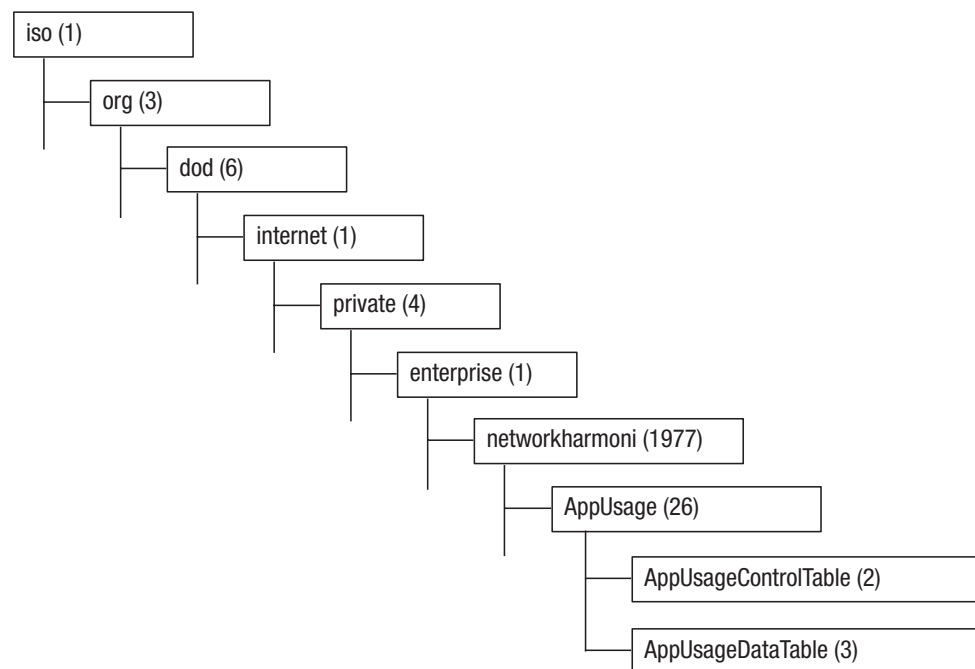
**Tip:** 24 hours = 86400 seconds

---

## MIB module

The appusage MIB is a subtree of networkharmoni (1977).

The location of the appusage subtree is shown in Figure 5.



*Figure 5. OID tree diagram for appUsage*

This section provides a summary of the objects defined in the appusage MIB module. For detailed information on all objects in the module, see the appUsage-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.



## MIB tables

The appUsage MIB provides a control table for setting up monitoring tasks and a data table that stores the monitor data.

### Application usage control table

The application usage control table (appUsageControlTable) contains configuration information for monitoring application usage.

Table 33 describes the row objects in appUsageControlTable.

*Table 33. Application usage control table (appUsageControlTable)*

Row object	Description
DataRows	The current number of rows in the data table that correspond to this control row.
Description	A textual description of the applications being monitored.
Index	An SNMPv2 row status object controlling creation, activation and deletion of the control row.
Owner	Indicates the owner of the control row.
Status	A user-defined description of the control row.
WindowSize	Sets the maximum duration (in seconds) of application inactivity before its record is deleted from the data table. The default value is 86400, which corresponds to a duration of 24 hours.

### Application usage data table

The application usage data table (appUsageDataTable) contains data about each active application.

Each data row represents an application that is currently running in the foreground window. The application usage data table is updated every second. Each row may represent multiple instances of the same application, each with a different process ID. Table 34 describes the row objects in appUsageDataTable.

*Table 34. Application usage data table (appUsageDataTable)*

Row object	Description
ApplicationName	A unique descriptive name for the application.
BinaryName	The binary name of an application.
CpuPercent	The percentage of CPU time occupied by this application during the previous window (appUsageControlWindowSize).
CpuTime	The total CPU time consumed by the application.
InputPercent	The percentage of real time for which the user was interacting with this application during the previous window (appUsageControlWindowSize).
InputTime	The total amount of real time that the user spent interacting with the application.
UserName	The name of the user who started the application.

---

## Chapter 5. Arithmetic

The arithmetic subagent and its MIB module provide facilities for flexible monitoring and alarm generation using a combination of free-form arithmetic expressions, logical comparisons, and built-in mathematical functions.

---

### Component files

The arithmetic subagent and MIB module are comprised of a set of component files.

Table 35 lists the component files and their installed locations.

*Table 35. Arithmetic component files*

File	Location	Description
arithmetic.dll (Windows) libarithmetic.sl/.so (UNIX)	bin	Binary implementation of the arithmetic subagent.
arithmetic-mib.mib	mibs	MIB definition document.
arithmetic.oid	config/oid	arithmetic subagent object identifier file.

---

### Guidelines

Use the arithmetic subagent and its MIB module to monitor values, thresholds or other behavior of MIB objects using free-form arithmetic expressions.

To load the subagent, use the command:

```
subagent load arithmetic
```

### General monitoring procedure

To use the arithmetic subagent for monitoring and event generation, create a control row in `arControlTable`.

#### Procedure

To create and configure a control row:

1. Set `arControlStatus` to `CreateAndWait(5)`.
2. Set `arControlExpression` to a control expression that represents the condition or threshold that you wish to monitor, for example:  
`.srSystemCPUUsageAverage.0 > 95`  
See “Defining control expressions” on page 42 for more information.
3. Set `arControlEvent` to the RMON eventIndex of the event to be triggered when the monitored condition or threshold is violated.
4. Activate the control row by setting `arControlStatus` to `active(1)`.

## Defining control expressions

Control expressions are free-form combinations of arithmetic expressions, logical comparisons, and mathematical functions. They define the condition that a control row monitors.

### Expression evaluation

The subagent evaluates control expressions at each sample interval, or in response to a triggering event.

A simple expression for monitoring average CPU usage on a host machine is as follows:

```
.srSystemCPUsageAverage.0 > 95
```

At each sample interval, defined by the `arControlInterval` object, the subagent evaluates the expression by comparing the value of the MIB object against the value 95. The results of the most recent expression evaluation are stored in the evaluation table, `arEvalTable`.

If required, you can configure an expression to be evaluated in response to an event, instead of at regular sample intervals. To do so, set the `arControlInterval` object to the value `-index`, where *index* is the `eventIndex` value of the desired RMON event. For example, setting `arControlInterval` to `-6` forces the expression to be evaluated whenever the RMON event with index value 6 is triggered.

**Tip:** When using event-triggered evaluation, you can also access the string value of varbinds in the notification associated with the triggering event. Use the `@n` syntax to refer to such varbinds.

### Variables

Variables are denoted by the `@` character.

The following expression assigns the numerical value 5 to the variable named `foo`, creating it if it does not already exist:

```
@foo = 5
```

Variables may be freely accessed and modified. The following expression assigns the value of the variable `@foo` to `@bar`, creating `@bar` if it does not already exist:

```
@bar = @foo
```

Variables retain their value until they are explicitly changed, even across samples. Variables that have not been assigned a value return zero when accessed. You can test if a variable has ever been assigned a value by using the expression:

```
@?foo
```

This expression returns true if `@foo` has been assigned a value, or false otherwise.

## Type rules

The arithmetic subagent applies type rules to both expressions and data.

These rules operate in the following ways:

- All numerical expressions are calculated using signed 64-bit arithmetic, which covers the full range of possible 32-bit signed and unsigned values, and the full range of 64-bit signed values.
- Logical expressions always return a value of either 0 (false) or 1 (true).
- When the subagent implicitly converts a string to an integer inside an expression, the value of the string evaluates to the length of the string.

## Scalars and arrays

Arithmetic expressions operate on two distinct data classes: scalars and arrays. The distinction between scalars and arrays is important in using them correctly in expressions.

Scalars are SNMP data types such as Integer32, Counter64, and OCTET STRING. Arrays are sequences of scalars resulting from expansion of OIDs that contain wildcard characters, or from operations on a history of scalar values.

If the `srStorageTable` contains five rows with indexes 1, 2, 101, 102, and 103, monitoring the five instances of the `srStorageUsedPercent` object requires either five expressions to operate on the individual scalar values:

```
.srStorageUsedPercent.1 > 90  
.srStorageUsedPercent.2 > 90  
.srStorageUsedPercent.101 > 90  
.srStorageUsedPercent.102 > 90  
.srStorageUsedPercent.103 > 90
```

or one expression to operate on the array of values:

```
.srStorageUsedPercent.* > 90
```

Both approaches generate five results (that is, five rows in `arEvalTable`), one for each instance of the `srStorageUsedPercent` object. In the array expression, the `*` wildcard expands to match all sub-OIDs of `srStorageUsedPercent`, and the expression is evaluated once for each instance.

**Tip:** In this example, the `?` wildcard could be used in place of `*`. It matches only one sub-OID, while `*` matches any number of sub-OIDs. You can place `?` at any position in the OID, while `*` may only be placed in the end (rightmost) position.

## Combining arrays and scalars

Expressions can combine array and scalar values. For example, the following expression divides an array of values by a scalar value:

```
.psRunningSize.* / 1024
```

The expression creates one row in the `arEvalTable` for each instance of the `psRunningSize` object. Each row stores the result of dividing a `psRunningSize` value by 1024. The effect of this expression is to convert each instance of the `psRunningSize` object into a megabyte value and return true for each value that is 1MB or greater.

## Operations on multiple arrays

Expressions cannot operate on more than one array directly. For example, the following expression for determining the proportion of total processor time spent on executing kernel code is not valid because it attempts to add several arrays:

```
delta( .srProcessorSys.* ) * 100 / delta(.srProcessorSys.* + .srProcessorIdle.*  
    + .srProcessorUser.* + .srProcessorWait.*)
```

In this situation, the correct implementation is:

```
@s=sum(.srProcessorSys.*);  
@i=sum(.srProcessorIdle.*);  
@u=sum(.srProcessorUser.*);  
@w=sum(.srProcessorWait.*);  
delta(@s) * 100 / delta(@s+@i+@u+@w)
```

This implementation is valid because the sum functions convert each array to a scalar result so that the add operation is then performed on scalars.

If per-CPU values were required, the original expression could be modified to use `[*]` references to values in the `srProcessorSys.*` array:

```
delta( .srProcessorSys.* ) * 100 / delta(.srProcessorSys.[*] + .srProcessorIdle.[*]  
    + .srProcessorUser.[*] + .srProcessorWait.[*])
```

The syntax `.objectName.[*]` refers to specific instances (scalars) that correspond to individual values in an array. If `.srProcessorSys.*` matches the instances

```
.srProcessorSys.1  
.srProcessorSys.2  
.srProcessorSys.3
```

then each occurrence of `[*]` expands to the sub-OIDs 1, 2, and 3.

**Note:** The resulting expansion of the `*` wildcard has the type `ObjectID`, even if the corresponding index value was of another type, such as `OCTET STRING`, because it is possible that the wildcard substitutes a combination of index types.

The alternative reference `[n]` extracts the *n*th sub-OID matched by a wildcard. This format is zero-referenced, so `[0]` refers to the first sub-OID of the match, `[1]` the second sub-OID and so on.

For example, in the Netcool/ASM for Websphere data table (`wsDataTable`) rows are indexed by a combination of `wsControlIndex`, `wsCounterIndex`, and `wsCollectionIndex` objects. When referring to the expression

```
.wsDataValue.*
```

the reference `[0]` corresponds to the `wsControlIndex` value, `[1]` to the `wsCounterIndex` value, and `[2]` to the `wsCollectionIndex` value. When a table has a variable length index, such as an octet string, `[n]` returns individual elements in the octet string.

## Expression grouping

Two or more separate expressions may be grouped using either semicolons (;) or commas (,). Grouped expressions combine a number of operations, but evaluate to the result of only one of those operations.

Grouped expressions have the general formats:

```
expr1 ; expr2 ; ... ; exprN  
expr1 , expr2 , ... , exprN
```

In both formats, *expr1* is evaluated first, followed by *expr2*, and finally *exprN*. When expressions are grouped using semicolons, the result is the value of the rightmost expression (*exprN*). When expressions are grouped using commas, the result is the value of the leftmost expression (*expr1*).

**Tip:** The semicolon always returns the value of the expression to its right. The comma always returns the value of the expression to its left.

Expressions may be joined using the logical operators && and ||. For example, the following expression evaluates to true if both *expr3* and *expr4* are true (non-zero):

```
expr1 && expr2 ; expr3 && expr4
```

**Note:** Be careful not to confuse the && and & operators. && is the logical expression operator and & is the binary arithmetic operator. The same applies to the || and | operators.

Enclosing expressions in parentheses () changes the order of the expression grouping. For example, the following expression evaluates to true if *expr1*, *expr3*, and *expr4* are all true:

```
expr1 && ( expr2 ; expr3 ) && expr4
```

## Conditional expressions

Conditional expressions provide an if-then-else style of construct for selectively evaluating sub-expressions based on particular conditions.

The general format of a conditional expression is:

```
condition ? exprWhenTrue : exprWhenFalse
```

If *condition* is true then *exprWhenTrue* is evaluated and returned, otherwise *exprWhenFalse* is evaluated and returned.

For example, the following expression tests the CPU usage and generates an event if the value exceeds 80%, in which case the result of the expression is a message string:

```
.srSystemCPUUsage.0 > 80 ? @result="Excessive CPU usage" : @result=0 ; @result
```

This expression can be shortened to:

```
.srSystemCPUUsage.0 > 80 ? "Excessive CPU usage" : 0
```

The expression can be shortened in this way because a string evaluates to its length, which is non-zero and true.

## Sample interval and control window

Sample intervals and control windows affect the period over which the subagent gathers data for evaluation.

Functions such as `trend()`, `max()`, and `sum()` are able to operate on data gathered over a specific time window, which is set by the `arControlWindow` object. Data is gathered at sample intervals set by the `arControlInterval` object; the first sample is taken when the control row is activated. The function itself is first evaluated after  $n$  samples, where  $n$  is the value of `arControlWindow`. From this point on, the function is evaluated at every sample interval, using the last  $n$  samples.

For example, if values of `srSystemCPUUsage` over the first eight samples are 5, 2, 6, 8, 9, 1, 7, 6, then the expression:

```
sum(.srSystemCPUUsage.0)
```

with `arControlWindow` equal to 4 returns the values 21, 25, 24, 25, 23 in samples 4 through to 8. During samples 1 through to 3, the expression does not return a result because the `arControlWindow` value requires that four samples be collected before the expression is evaluated.

## Variable persistence between samples

Variables defined in control expressions persist between samples. This is useful for storing state and other historical information.

For example, the expression:

```
.srSystemCPUUsage.0 > @max ? @max = .srSystemCPUUsage.0 : @max
```

evaluates to the highest value of `srSystemCPUUsage` over the lifetime of the control row. This expression is not the same as `max(.srSystemCPUUsage.0)`, which returns the maximum value of `srSystemCPUUsage` over the last  $n$  samples only, where  $n$  is the value of `arControlWindow`.

## Debugging control expressions

The arithmetic MIB module provides facilities for debugging common problems in control expressions.

If an expression contains syntax errors, the value of the corresponding `arControlStatus` object remains at `notReady`, and the control row cannot be activated. In this situation, the `arControlErrorPos` object indicates the position of the first syntax error in the expression.

You can also use the `arControlDebug` object to help you locate errors. When this object has the value `on(1)`, the subagent creates additional rows in the evaluation table, `arEvalTable`, one for each subexpression in the control expression. Each row that represents a subexpression has an `arEvalId` value starting from 10000, and indicates the offset of the subexpression within the control expression, as well as the value to which the subexpression evaluated during the most recent sample. You can use this information to verify that sub-expressions are parsed as you intended, and that expressions containing wildcards match the expected number of instances.

**Note:** `arEvalValue` objects in rows that are created using the `arControlDebug` object are not inserted into notifications as variable bindings.



If a control expression contains wildcards, `arEvalTable` contains a row for each expanded instance of the wildcard.

## Expression syntax elements

Expression syntax elements provide ways of representing literal values, defining and evaluating variables, and ordering sub-expressions.

Table 36 lists the expression syntax elements supported by the subagent.

Table 36. Control expression syntax

Format	Description	Type
<i>n</i>	A signed 64-bit decimal integer literal.  Sample usage: 123	int
0xn	A signed 64-bit hexadecimal integer literal.  Sample usage: 0x7B	int
.n.n.n	Evaluates the value of the MIB object identified by its OID. This is equivalent to performing an <b>SNMP GET</b> on the given OID.  Sample usage: .1.3.6.1  OIDs may contain wildcard characters: <ul style="list-style-type: none"> <li>• ? matches a single sub-OID.</li> <li>• * matches multiple sub-OIDs. It may only be placed at last sub-OID position.</li> </ul> Only one OID in an expression may contain a wildcard, however other OIDs may use [ <i>n</i> ] bracket syntax to refer to the wildcard (see the following entry).  The OID may also have the format .n.n.n.( <i>expr</i> ).n, where <i>expr</i> evaluates to either an integer or an OID.  Sample usage: .1.3.6.5.* + .1.3.6.7.[0]  To refer to a MIB object by its symbolic name use the format . <i>object-name</i> .n, for example .srSystemCPUUsage.0.	obj
[ <i>n</i> ]	Refers to the <i>n</i> th sub-OID of a wildcard instance.  Sample usage: when matching the OID .1.2.3.7.8.9, the expression.1.2.3.* / .1.4.5.[0].[2] evaluates to.1.2.3.7.8.9 / .1.4.5.7.9 .The following alternative forms of sub-OID references are supported: <ul style="list-style-type: none"> <li>• [n-] All sub-OIDs from index <i>n</i> onward (0+)</li> <li>• [n-m] All sub-OIDs from index <i>n</i> to <i>m</i></li> <li>• [*] All sub-OIDs matched by * (equivalent to [0-])</li> </ul> References to single sub-OIDs in the form [ <i>n</i> ] may be used as an integer in further calculations. References to multiple sub-OIDs in the forms [n-m] or [*], may only be used inside an OID.	int

Table 36. Control expression syntax (continued)

Format	Description	Type
<i>str</i> #/ <i>regex</i> /	Extracts a substring from the string <i>str</i> matching the regular expression <i>regex</i> . If no match occurs, the expression returns an empty string, which is equivalent to the value <code>false</code> . The argument <i>str</i> may be a MIB object, a variable, or a literal value. Non-string values are converted to their string equivalents.  Sample usage: .1.3.6.1.4.1.1977.20.1.1.4.4#/.*exe/ @foo#/bar/ 'stringvalue'#/.*val.*/	str
' <i>string</i> ' " <i>string</i> "	A string literal.  Sample usage: "foo"	str
@ <i>var</i> = <i>expr</i>	Assigns the value of expression <i>expr</i> to the variable named <i>var</i> . If the variable does not already exist, it is created. Variables persist between samples.  Variable names must match the regular expression [A-Za-z_][A-Za-z0-9_]*.  Sample usage: @two=1+1	int
@ <i>var</i>	Evaluates the variable <i>var</i> . Uninitialized variables always have the value 0 (zero).  Variables are local to each control row except where one expression triggers another using the <code>arControlLink</code> object. In that case, the triggered row inherits the variables of the triggering row.  Sample usage: @foo	int
@#	Returns the number of instances that matched the wildcard expansion used in an expression. If wildcards are not used in the expression, the value is 1.	int
@~	Represents an undefined value.  Sample usage: (@x >= 0) ? (@x * 2) : @~	?
@ <i>n</i>	If expression evaluation is controlled by an event, this expression refers to the string value of the <i>n</i> th varbind of the notification associated with that event. If the referenced varbind does not exist, the reference evaluates to null (undefined value).  Sample usage: @1 (refers to the first varbind)	str
@? <i>var</i>	Returns true if the variable <i>var</i> is defined, and false otherwise.  Sample usage: @?foo	int
;	Separates expressions. When an expression contains semi-colon, the value of the entire expression is the value of the expression on the right side of the semi-colon.  Sample usage: (@two=1+1; @two)	int
, (comma character)	Separates expressions. When an expression contains a comma, the value of the entire expression is the value of the expression on the left side of the comma.  Sample usage: (@result=1, @two=2)	int

Table 36. Control expression syntax (continued)

Format	Description	Type
{ <i>expr</i> }	Selects sub-expressions to be recorded as individual results in <code>arEvalTable</code> . Results are ordered according to the order of opening braces, for example <code>1 + {{1} + 1}</code> yields three results with values 3, 2, and 1 in that order. The result of the entire expression is always the first result (the evaluation table row with <code>arEvalID</code> of 0).	<code>expr</code>
<code>A -&gt; B</code>	Writes the string form of expression A followed by a newline to the file represented by the string form of expression B. If A is undefined then the contents of the file B are erased. Returns the number of lines written to the file.  Sample usage: <code>@msg -&gt; "/tmp/msg.txt"</code>	<code>int</code>
<code>A -&gt;&gt; B</code>	Appends the string form of A followed by a newline to the end of file represented by the string form of expression B. If A is undefined then this expression has no effect. Returns the number of lines written to the file.  Sample usage: <code>@msg -&gt; "c:\temp\msg.txt"</code>	<code>int</code>
Return types: <i>expr</i> = value of enclosed expression; <i>int</i> = integer; <i>?</i> = undefined; <i>obj</i> = MIB object value; <i>str</i> = string		

## Operators

The subagent supports a range of arithmetic and logical operators in expressions.

Table 37 lists the operators that you may use in expressions.

Table 37. Expression operators

Operator	Operation
( )	Parentheses for simple expression grouping.
-	Unary negation.
+	Addition, and string concatenation.
-	Subtraction.
*	Multiplication.
/	Division.
%	Modulus (integer remainder).
&	Bitwise AND.
	Bitwise OR.
^	Bitwise XOR.
~	Bitwise NOT.
<<	Bitwise shift left.
>>	Bitwise shift right.
!	Logical NOT.
&&	Logical AND.
	Logical OR.
==	Equals.
!=	Not equal to.

Table 37. Expression operators (continued)

Operator	Operation
>	Greater than.
<	Less than.
>=	Greater than or equal to.
<=	Less than or equal to.
a -> b	Write the string form of expression a to the file name represented by string expression b. Returns an integer which is the number of lines successfully written. If a is undefined (such as an integer divided by zero), file b is truncated to zero size.
a ->> b	Append the string form of expression a to the file name represented by string expression b. Returns an integer which is the number of lines successfully appended. If a is undefined, file b remains unchanged.
x ? y : z	Conditional evaluation; if x is true then returns the value of y, otherwise returns the value of z. The return type of this operator is the type of y or z.

Operators have a defined order of precedence. The order of precedence for expression operators, in descending order, is:

```
( ) { }
~ ! -(unary) #/regex/
* / %
+ -(binary)
<< >> -> ->>
< > >= <= != ==
&
^
|
&&
||
?:
=
,
;
```

## Built-in functions

The subagent provides a number of built-in arithmetic and statistics functions.

Table 38 lists the built-in functions, and indicates their return type, as well as the input forms that they support.

Table 38. Built-in functions

Function	Description	Return Type	Form
abs( <i>x</i> )	Returns the absolute value of expression <i>x</i> .	int	scalar
added( <i>x</i> )	Returns 1 if the MIB object <i>x</i> was defined within the last sample interval, and 0 otherwise. This is useful in determining if any new objects have been created in the MIB.	int	scalar
count( <i>x</i> )	Returns the number of values of the expression <i>x</i> . <b>Tip:</b> Use this function to determine the number of rows in a table by passing it a wildcarded OID.	int	array

Table 38. Built-in functions (continued)

Function	Description	Return Type	Form
datestr( <i>x</i> )	Converts a DateTime MIB object into a human-readable date and time string of the format: Sun Jan 01 12:00:00 2006  When expression <i>x</i> is an integer value, the function interprets it as a UNIX timestamp. If no conversion is possible, the function returns an empty string.	str	scalar
delta( <i>x</i> )	Calculates the difference between the current value of expression <i>x</i> and its value from the previous sample.	int	scalar
exists( <i>x</i> )	Returns 1 if the value of expression <i>x</i> is defined, or 0 if the value is not defined. <b>Tip:</b> Use this function to determine whether a given OID exists in the MIB.	int	scalar
int( <i>x</i> )	Converts a string value into its integer representation, or a null value if no conversion is possible.	int	scalar
ipaddr( <i>x</i> )	Converts an IPAddress MIB object into an Internet standard dotted notation format, for example 172.0.0.1. If no conversion is possible, the function returns an empty string.  The function can also be applied to integer values, in which case the value is interpreted as a binary IP address, for example ipaddr(0x7f000001) returns 127.0.0.1.	str	scalar
max( <i>x</i> )	In scalar form, returns the maximum of the last <i>n</i> samples of expression <i>x</i> , where <i>n</i> is the value of arControlWindow.  In array form, returns the maximum of the set of values represented by expression <i>x</i> .	int	scalar array
mean( <i>x</i> )	In scalar form, calculates the mean of the last <i>n</i> samples of expression <i>x</i> , where <i>n</i> is the value of arControlWindow.  In array form, it calculates the mean of the set of values represented by expression <i>x</i> .	int	scalar array
min( <i>x</i> )	In scalar form, returns the minimum of the last <i>n</i> samples of expression <i>x</i> , where <i>n</i> is the value of arControlWindow.  In array form, returns the minimum of the set of values represented by expression <i>x</i> .	int	scalar array
removed( <i>x</i> )	Returns 1 if the MIB object <i>x</i> was removed within the last sample interval, and 0 otherwise. This is useful in determining if any objects have been deleted from the MIB.	int	scalar
stddev( <i>x</i> )	In scalar form, calculates the standard deviation of the last <i>n</i> samples of expression <i>x</i> , where <i>n</i> is the value of arControlWindow.  In array form, calculates the standard deviation of the values of <i>x</i> .	int	scalar array
stddev100( <i>x</i> )	Identical to stddev() but scaled by 100 to retain two decimal places of accuracy.	int	scalar

Table 38. Built-in functions (continued)

Function	Description	Return Type	Form
<code>str(x)</code>	Returns the string representation of expression <i>x</i> , or an empty string if no conversion is possible.	str	scalar
<code>sum(x)</code>	In scalar form, calculates the sum of the last <i>n</i> samples of expression <i>x</i> , where <i>n</i> is the value of <code>arControlWindow</code> .  In array form, calculates the sum of the set of values represented by expression <i>x</i> .	int	scalar array
<code>time(x)</code>	Converts a <code>DateTime</code> MIB object into a UNIX timestamp, or 0 if no conversion is possible.	int	scalar
<code>trend(x, [y])</code>	Performs linear extrapolation.  In scalar form, the extrapolation is performed over the last <i>n</i> samples of expression <i>x</i> , where <i>n</i> is the value of <code>arControlWindow</code> .  In array form, the extrapolation is performed over the set of values represented by expression <i>x</i> .  The second parameter <i>y</i> is optional. It indicates the number of seconds into the future to extrapolate the value. This parameter may be specified using time-specifier modifiers to the integer as follows: <ul style="list-style-type: none"> <li>• s seconds</li> <li>• m minutes</li> <li>• h hours</li> <li>• d days</li> <li>• W weeks</li> <li>• M months (28 days)</li> </ul> For example, <code>trend(.sysUpTime.0, 30d)</code> extrapolates the value of <code>sysUpTime</code> 30 days into the future. <b>Tip:</b> To improve accuracy, pre-scale the input to the function, for example <code>trend(1000*.sysUpTime.0, 30d)</code> , then divide the result by the same scaling factor when using it in subsequent calculations.	int	scalar array
<code>utc()</code>	Returns the current system time in UTC, formatted as a UNIX timestamp. A value of -1 indicates the current system time is outside the supported range, which is January 1, 1970 to 3:14:07 January 19, 2038.	int	n/a

## Generating events

The subagent triggers an event at each sample interval in which a control expression evaluates to true.

If a control row's `arControlObject` contains the value of an RMON `eventIndex`, the event is fired. In expressions containing wildcards, more than one instance of the expanded expression may evaluate to true, so more than one event can be triggered per sample.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### Control table

The arithmetic commands create rows in the arithmetic control table (arControlTable).

The general syntax of these commands is:

```
arithmetic property=value  
arithmetic create property=value ...  
arithmetic reset
```

Table 39 lists the properties supported in these commands.

*Table 39. Configuration command parameters - arithmetic*

Property	Type	Description	Sets MIB object
countdown	int	The number of samples before the value arControlToggle is automatically set to off.	Countdown
debug	enum	Controls debug behavior of the evaluation table, arEvalTable:  on  off	Debug
description	string	A description of the control expression.	Description
event	int	The RMON eventIndex of event fired when the control expression in the current row evaluates to true.	Event
expression	string	The control expression defining the alarm or monitor condition, with maximum length 1024 characters.	Expression
interval	int	The sample interval (in seconds).  If a positive value, sets the interval (in seconds) at which the expression defined in arControlExpression is evaluated.  If a negative value, sets the event which triggers the evaluation of arControlExpression. For example, setting arControlInterval to -6 forces the expression to be evaluated whenever the RMON event with index value 6 is triggered.  If the value is 0, evaluates arControlExpression for the first interval only.	Interval

Table 39. Configuration command parameters - arithmetic (continued)

Property	Type	Description	Sets MIB object
link	int	The index of another arControlTable row whose arControlToggle object is set to on when the control expression evaluates to true.	Link
toggle	enum	Controls generation of events and data:  on  off	Toggle
window	int	The number of samples in the sample window.	Window

## Examples

These examples demonstrate how to use the arithmetic subagent to perform specific tasks.

### CPU usage alarm with hysteresis

Create an genalarm-style hysteresis-mode alarm for the host system's CPU usage with rising and falling thresholds of 80% and 60% respectively, evaluated at one-minute intervals with rising and falling durations of five minutes:

```
subagent load mib2
subagent load rmonc
subagent load sysres
subagent load arithmetic

event type=snmp-trap
event community=public
event description="CPU usage event"
event create
eventIdx=$?

arithmetic reset
arithmetic interval=60
arithmetic window=5
arithmetic description="CPU Usage"
arithmetic event=$eventIdx
arithmetic expression="
    @val = .srSystemCPUUsage.0;
    @rise = 80;
    @fall = 60;
    (min(@val) > @rise && @state != 2) ?
        @result = "CPU usage above @rise",
        @state = 2
    : (max(@val) <= @fall && @state != 0) ?
        @result = "CPU usage returned to normal (less than @fall)",
        @state = 0
    : (min(@val) <= @rise && max(@val) >= @fall) ?
        @result = 0,
        @state = 1
    : @result = 0;
    @result
"
arithmetic create
```

### Monitoring growth rates

Monitor the growth rate of storage on the local file system, and generate an alarm if it changes by more than 5% in one hour:



```

subagent load mib2
subagent load rmonc
subagent load hostres
subagent load sysres
subagent load arithmetic

event type=snmp-trap
event community=public
event description="Storage growth rate event"
event create
eventIdx=$?

arithmetic reset
arithmetic interval=3600
arithmetic window=1
arithmetic event=$eventIdx
arithmetic description="Storage growth rate"
arithmetic expression="
    @fixed = 4;
    @used = .srStorageLogicalUsedPercent.(@fixed).?;
    @delta = delta(@used);
    @delta ?
        @part=.hrStorageDescr.[0];
        @result = "Growth rate on partition @part was @delta % in the last hour"
    : @result = 0;
    @result
"
arithmetic create

```

## Predicting disk usage

Monitor disk usage on the host system and generate an alarm if the fixed disk usage exceeds 80% and the usage trend indicates that it will exceed 95% in the next day:

```

subagent load mib2
subagent load rmonc
subagent load hostres
subagent load sysres
subagent load arithmetic

event reset
event type=snmp-trap
event community=public
event description="Disk usage trend critical"
event createeventIdx=$?

arithmetic reset
arithmetic event=$eventIdx
arithmetic interval=3600
arithmetic window=12
arithmetic expression="
    @fixed = 4;
    @usedmax = 80;
    @trendmax = 95;
    @used = .srStorageLogicalUsedPercent.(@fixed).?;
    @trend = trend(@used,1d);
    (@used > 80 && @trend > 95) ?
        @disk = .hrStorageDescr.[0];
        @result = "Usage on @disk is @used % and will exceed @trendmax % in 1 day"
    : @result = 0;
    @result
"
arithmetic create

```

## Monitoring Windows services

Monitor all Windows services that are configured to start automatically, and generate an alarm if any of them stop:

```
subagent load mib2
subagent load rmonc
subagent load ntscm
subagent load arithmetic

event reset
event type=snmp-trap
event community=public
event description="Windows service failure"
event create
eventIdx=$?

arithmetic reset
arithmetic event=$eventIdx
arithmetic interval=10
arithmetic window=1
arithmetic expression="
((.ntServiceTableCurrentState.*==7)&&(.ntServiceTableStartType.[*]==1))?
  @name = .ntServiceTableDisplayName.[*];
  @result = "Automatic service @name has stopped"
  : @result = 0;
  @result
"
arithmetic create
```

## Calculating total swap usage

Calculate total swap usage in kilobytes, using data from hrStorageTable. Swap space data in this table is indicated by the type hrStorageVirtualMemory (OID .1.3.6.1.2.1.25.2.1.3):

```
subagent load hostres
subagent load arithmetic

arithmetic reset
arithmetic description="Total swap usage in KB"
arithmetic expression='
sum(
  @hrstoragetype=str(.hrStorageType.*);
  @swap=".1.3.6.1.2.1.25.2.1.3";
  @hrstoragetype == @swap ?
    @units=.hrStorageAllocationUnits.[0];
    @used=.hrStorageUsed.[0];
    @byteused=@used * @units;
    @byteused / 1024
  : 0
)
'
arithmetic create
```

## Miscellaneous monitor expressions

This section provides a list of expression definitions for a selection of monitoring tasks:

1. Monitor only DB2 file systems and generate an alarm on 85% usage:

```
arithmetic expression="
(.hrStorageDesc.*#\db2.*)
&&
(.hrStorageUsed.[*] * 100 / .hrStorageSize.[*] > 85)
"
```

2. Monitor CPU usage and generate an alarm containing a severity code based on the level of use:

```
arithmetic expression="
@val=.srSystemCPUUsage.0 ;
(@val >= 75) ?
  @code = "CPU usage critical (@val%)"
: (@val >= 60) ?
  @code = "CPU usage high (@val%)"
: (@val >= 50) ?
  @code = "CPU usage moderate (@val%)"
: @code = 0;
@code
"
```

3. Monitor the free space ratio of Oracle tablespaces. Generate an alarm if the ratio falls below 10%. Embed the percentage of free space, the name of the tablespace, and the SID of the corresponding Oracle instance as varbinds in the events generated:

```
arithmetic expression="
({.oraTableSpaceFreeSpaceRatio.*} < 10) ?
  {@name = .oraTableSpaceName.[*]};
  {@sid = .oraInstanceSID.[0]};
  @result = "Free space ratio low on @name, @sid"
: @result = 0;
@result
"
```

---

## MIB module

The arithmetic MIB is a subtree of networkharmoni (1977).

The arithmetic subtree is shown in Figure 6.

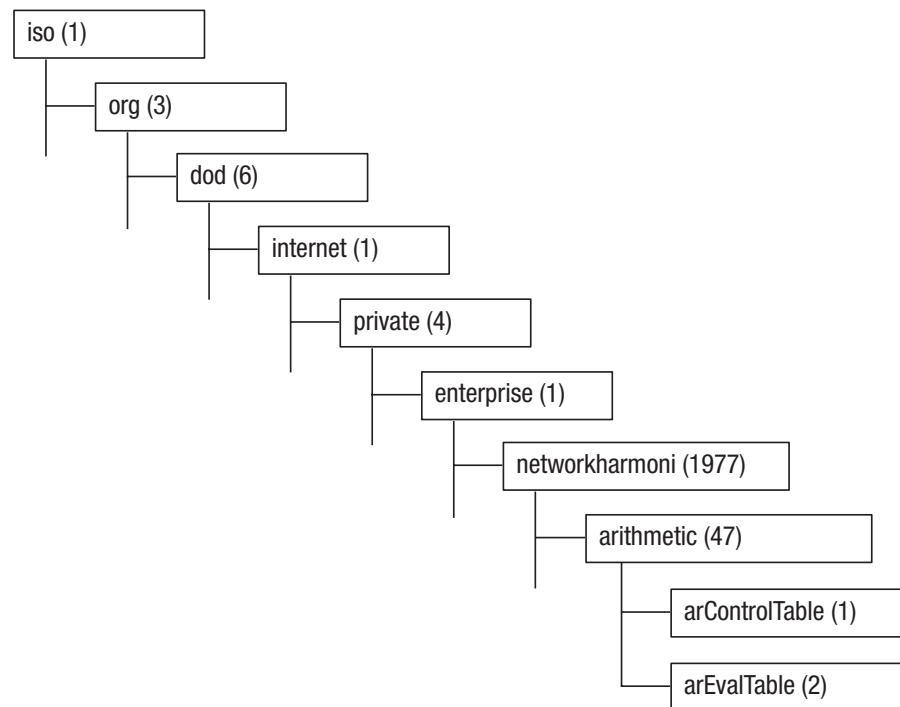


Figure 6. OID tree diagram of the arithmetic MIB module

This section provides a summary of the objects defined in the arithmetic MIB module. For detailed information on all objects in the module, see the arithmetic-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB tables

The arithmetic MIB provides a standard SNMP interface with control row semantics.

The MIB module contains a control table, `arControlTable`, for defining monitors, and an evaluation table, `arEvalTable`, that stores results.

### Control table

The control table (`arControlTable`) contains a list of control information for configuring arithmetic expression monitoring.

Table 40 lists the objects in `arControlTable`.

*Table 40. Arithmetic control table (arControlTable)*

Row object	Description
Countdown	<p>Sets the number of samples that the control row performs before it is automatically toggled off (<code>arControlToggle</code> set to <code>off(2)</code>). A value of -1 disables the countdown behavior.</p> <p>This object is equivalent to event flow control objects available in other MIB modules.</p>
Debug	<p>Controls the storage of debug information in <code>arEvalTable</code>:</p> <p><code>on(1)</code> - Results of individual subexpressions and the parse trees of those expression are stored in <code>arEvalTable</code>.</p> <p><code>off(2)</code> - Only the result of the entire expression is stored in the <code>arEvalTable</code>.</p> <p>Use this object to assist in debugging expressions.</p>
Description	A text description of the control row.
ErrorPos	<p>Indicates the position within the expression defined in <code>arControlExpression</code> of the first syntax error encountered. A value of -1 indicates that the expression is valid.</p> <p>If the expression contains syntax errors, the value of the <code>arControlStatus</code> object remains at <code>notReady</code>.</p>
Event	Index of an RMON event to fire when the expression defined in <code>arControlExpression</code> evaluates true. If the value of this object is zero, no event is fired.
EventCount	<p>The total number of times that the expression defined in <code>arControlExpression</code> has evaluated to true since the control row was last activated.</p> <p><b>Note:</b> In expressions containing wildcards, more than one instance of the expanded expression may evaluate to true, so more than one event can be triggered per sample.</p>

Table 40. Arithmetic control table (arControlTable) (continued)

Row object	Description
Expression	<p>Defines the expression to be evaluated. When the expression defined in this object evaluates to true (a non-zero value), an alarm is triggered.</p> <p>The expression is evaluated at intervals set by arControlInterval. The expression is not fully evaluated until the number of samples indicated by arControlWindow have occurred, and only when arControlToggle is set to on(1).</p> <p>See “Defining control expressions” on page 42 for more information about the syntax of control expressions.</p>
Index	Uniquely identifies the row.
Instances	<p>Reports the number of object instances that were matched by the expression defined in arControlExpression during the previous sample.</p> <p>If the expression contains a wildcard, the value of this object indicates the number of object instances matching the expression. If the expression contains no wildcards, a value of 1 indicates that all referenced objects exist. In either case, a value of 0 indicates that at least one referenced object was missing.</p>
Interval	<p>If a positive value, sets the interval (in seconds) at which the expression defined in arControlExpression is evaluated.</p> <p>If a negative value, sets the event which triggers the evaluation of arControlExpression. For example, setting arControlInterval to -6 forces the expression to be evaluated whenever the RMON event with index value 6 is triggered.</p> <p>If the value is 0, evaluates arControlExpression for the first interval only.</p>
LastUpdate	The value of sysUpTime when the expression was last evaluated.
Link	<p>Identifies another control row to toggle when the expression defined in arControlExpression evaluates to true.</p> <p>Linking control rows together in this way allows you to build a state machine. Setting this object to a negative value creates a reverse link. Setting it to -n causes control row n's arControlLink object to be set to the current control row index. You can control whether the control row remains active when a link is followed by setting the arControlCountdown field.</p>
Owner	Identifies the owner/creator of the control row.
Status	SNMPv2 row status. Controls creation, activation and deletion of the control row.
Toggle	<p>Toggles evaluation of the control expression:</p> <p>on(1) - Evaluation is enabled (control row is toggled on).</p> <p>off(2) - Evaluation is disabled (control row is toggled off).</p> <p><b>Note:</b> If the control row is configured to perform window-based evaluation of expressions, data collection is not suspended when the control row is toggled off. This behavior ensures that complete data is available as soon as possible when the row is toggled on again. A side-effect is that data is still acquired from MIB objects even though the control row is toggled off. To completely toggle data acquisition, use the arControlStatus object instead.</p>

Table 40. Arithmetic control table (arControlTable) (continued)

Row object	Description
Window	Specifies the number of samples collected before the expression defined in arControlExpression is evaluated. A sample window of size greater than 1 is required for built-in statistical functions such as mean() and stddev() that operate on a set of samples.

## Evaluation table

The evaluation table (arEvalTable) stores the results of expression evaluation.

Evaluation table rows are indexed by a combination of arControlIndex, subexpression identifier, and monitored MIB instance identifier.

If the arControlDebug object of the corresponding control row is set to on(1), the table contains one row for each subexpression in the arControlExpression object; otherwise, the table contains a single row for the result of the entire expression. Table 41 lists the objects in arEvalTable.

Table 41. Arithmetic evaluation table (arEvalTable)

Row object	Description
Expression	A copy of the subexpression that the row represents, for reference and debugging.
Id	Unique identifier of a result or subexpression. For result rows, this value ranges from 0 to $n$ where $n$ is the number of subexpression enclosed by braces.  The value of the entire arControlExpression is 0 and the $n$ th expression enclosed in braces is $n$ . Identifier values for debug rows commence at 10000.
Instance	The object identifier of the wildcarded MIB object being monitored. If there are no wildcards in arControlExpression then the arEvalTable contains one row for each result and the arEvalInstance object is empty (that is, it contains a zero-length OID).
Offset	The character offset within arControlExpression at which the expression starts. Offsets start at zero.
Value32	The value of the expression or subexpression in the most recent sample, or zero if the data was missing.
Value64	The value of the expression or subexpression in the most recent sample, or zero if the data was missing.
Value0id	The value of the expression or subexpression in the most recent sample, if the value was an OID, or the null object identifier .0.0.
ValueStr	The value of the expression or subexpression in the most recent sample, if the value was a string value.

## Notification types

The arithmetic MIB module defines a notification type for events generated by the subagent.

This notification type is shown in Figure 7.

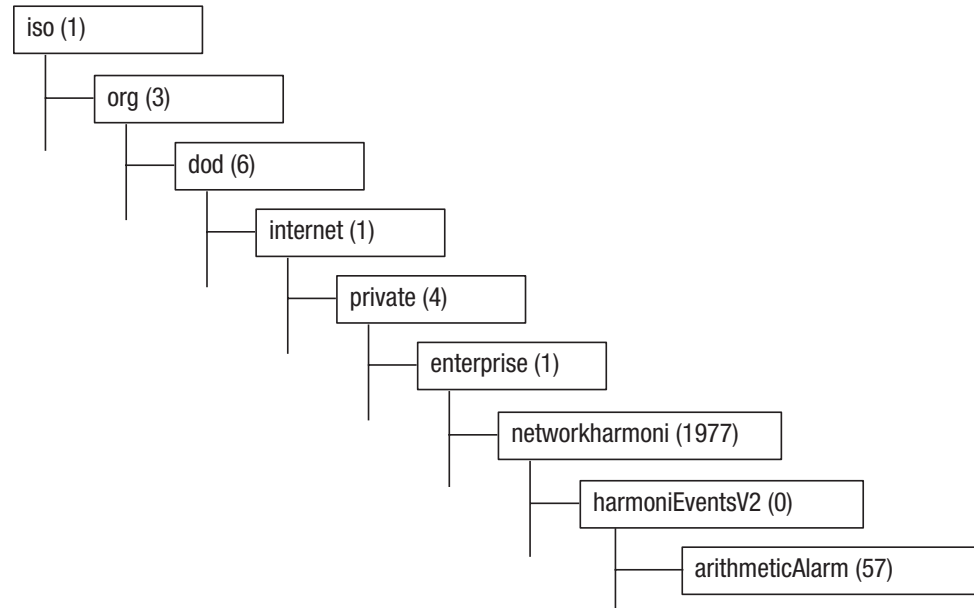


Figure 7. OID tree diagram for arithmetic notification types

arithmeticAlarm notifications are generated when arControlExpression evaluates to true. The subagent only sends notifications if the arControlEvent object is non-zero, and the corresponding eventType object in the RMON eventTable is set to either snmp-trap(3) or log-and-trap(4).

The arithmeticAlarm notification type contains the following variable bindings:

- arControlDescription
- arControlExpression
  - By default, the notification is blank unless overridden by the ArithmeticTrapSelf=true inivar.
- The result of the expression, which is either arEvalValue32, arEvalValue64, arEvalValueStr, or arEvalValue0id depending on the result type of the expression.





---

## Chapter 6. Availability

The availability subagent and the associated availability MIB provide a mechanism for determining whether an application endpoint or network host is available.

Application availability determines whether a specific port is reachable and network availability determines whether the network address is reachable. The availability MIB allows queries to be performed periodically and events to be generated if an application or host is unavailable.

---

### Component files

The availability subagent and MIB module are comprised of a set of component files.

Table 42 lists the availability subagent and MIB module component files and their installed locations.

*Table 42. Availability component files*

File	Location	Description
availability.dll (Windows) libavailability.so/.sl (UNIX)	bin	Binary implementation of the availability subagent.
availability-mib.mib	mibs	availability MIB definition document.
availability.oid	config/oid	availability subagent object identifier file.

---

### Guidelines

Use the availability MIB and subagent to determine network address and port availability.

To load the subagent, use the command:

```
subagent load availability
```

### Availability method

The availability method selects between network and application queries.

The `availabilityControlMethod` specifies whether the query determines network availability or application availability:

- Network availability determines if the destination IP address is reachable.
- Application availability determines if an application protocol is listening at the destination IP address and destination IP port.

## Polling and history

The subagent performs traces at regular intervals and stores a history of these traces.

The `availabilityControlSampleInterval` object configures the control row to perform traces at regular intervals. After each interval the data in `availabilityDataTable` is updated with the results of the most recent availability query.

The availability results may be moved into `availabilityHistoryTable` at regular intervals by requesting buckets and setting the history sample interval `availabilityControlHistoryInterval`. At every history-sampling interval, each `availabilityDataEntry` is moved into the `availabilityHistoryTable`.

## Event activation and deactivation

Using the `availabilityControlTurnOnEventIndex` and `availabilityControlTurnOffEventIndex` objects, you can configure availability control rows to be activated and deactivated by RMON events.

## Event generation

The availability subagent generates events to indicate that a host or service is no longer available, or that its availability has changed.

The event control objects in `availabilityControlTable` determine the type of events generated:

- `availabilityControlEventIndex` sets the event generated if the availability query fails. This event indicates that a host or service is unavailable.
- `availabilityControlFailureEventIndex` sets the event generated if availability query fails, but had succeeded during the previous sample interval. The event corresponds to a transition in the value of `availabilityDataAvailable` from `true(1)` to `false(2)`. This indicates that a host or service has become unavailable since the previous query.
- `availabilityControlSuccessEventIndex` sets the event generated if availability query succeeds, but had failed during the previous sample interval. The event corresponds to a transition in the value of `availabilityDataAvailable` from `true(1)` to `false(2)`. This indicates that a host or service has become available since the previous query.
- `availabilityControlEventStatus` provides event flow control. The value `eventReady(1)` provides one-shot triggering; `eventAlwaysReady(3)` provides automatic retriggering.

Figure 8 on page 65 shows the relationship between event generation and the value of the `availabilityDataAvailable` object, which stores the result of the most recent availability query. The event generation sequence shown in the diagram assumes that `availabilityControlEventStatus` has the value `eventAlwaysReady(3)`.

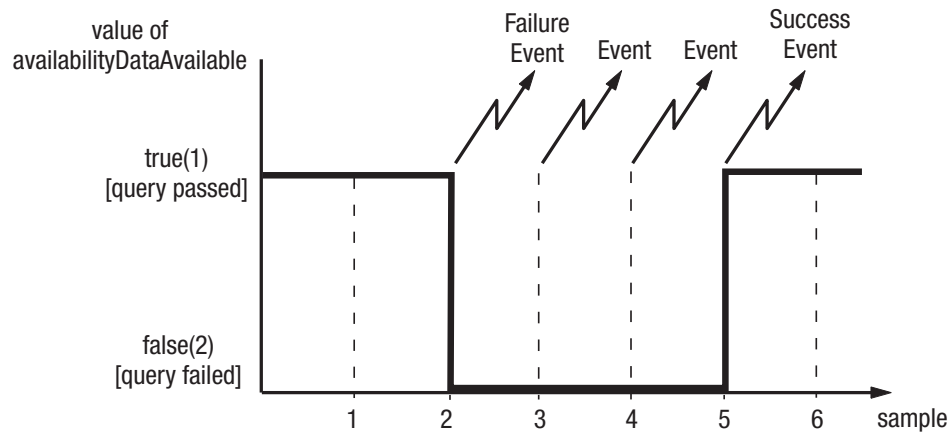


Figure 8. Availability event generation

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The availability commands create rows in the control table (`availabilityControlTable`).

The general syntax of these commands is:

```
availability property=value
availability create property=value ...
availability reset
```

Table 43 lists the properties supported in these commands.

Table 43. Configuration command parameters - availability

Property	Type	Description	Sets MIB object
buckets	int	The number of rows in <code>availabilityHistoryTable</code> requested for storing the query data.	BucketsRequested
datacontrol	enum	Data control: on - Enables data collection off - Suspends data collection	DataControl
dest	IP	The destination IP address for the query.	DestAddress
event	int	The index of the event generated when the availability test fails.	EventIndex

Table 43. Configuration command parameters - availability (continued)

Property	Type	Description	Sets MIB object
eventstatus	enum	Event flow control:  alwaysready  fired  ready	EventStatus
failevent	int	The index of the event generated when the result of the availability query changes from success to failure.	FailureEventIndex
history	int	Sets the interval (in seconds) at which data is moved into the history table.	HistoryInterval
method	enum	Sets the availability detection method:  application - Determines whether an application is listening on the port specified  network - Determines whether the destination address is reachable via ICMP protocol.	Method
offevent	int	The control row deactivation event index.	TurnOffEventIndex
onevent	int	The control row activation event index.	TurnOnEventIndex
passevent	int	The index of the event generated when the result of the availability query changes from failure to success.	SuccessEventIndex
port	int	The destination port for the query.	Port
sample	int	Sets the sampling interval for the availability test (in seconds).	SampleInterval

## Examples

This example demonstrates how to perform a simple monitoring task using the availability subagent.

### Monitoring telnet availability

Query the availability of the telnet service (port 23) on the host with IP address 192.168.2.2 every hour. Generate three types of event, indicating when the service is unavailable, has just gone down, or has just come back up:

```
subagent load availability
subagent load rmonc
```

```
event reset
event type=snmp-trap
event community=public
#triggered whenever the service is down
event description="Telnet service is unavailable"
event create
failidx=$?
```

```
#triggered when the availability *changes* from pass to fail
event description="Telnet service has just gone down"
event create
```

```
changetofailidx=$?
```

```
#triggered when the availability *changes* from fail to pass  
event description="Telnet service has just come up"  
event create  
changetopassidx=$?
```

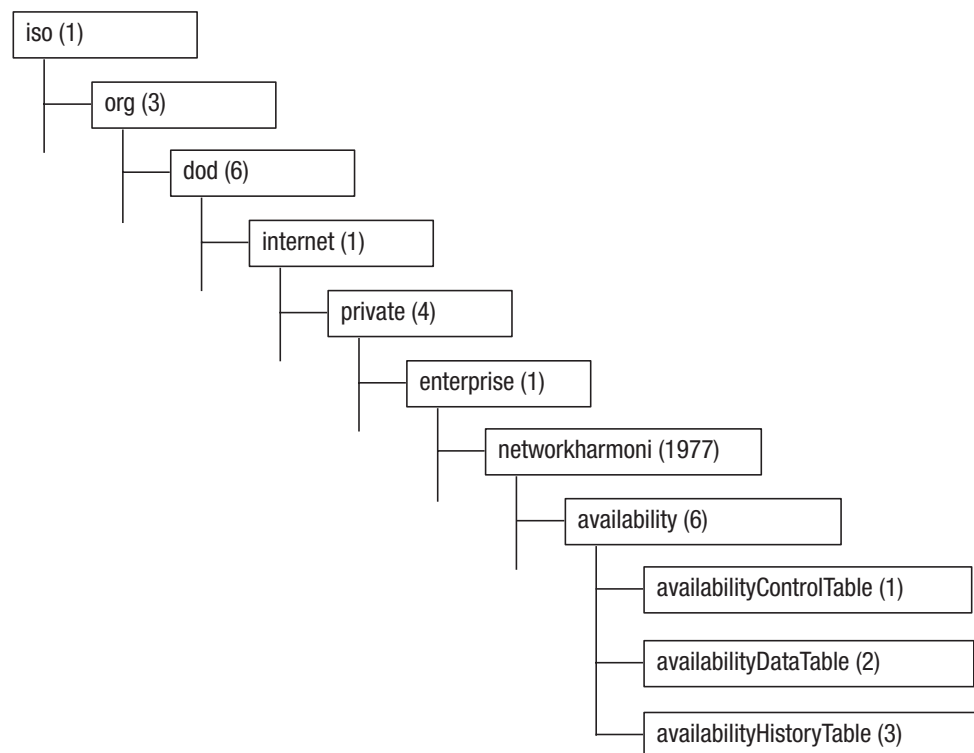
```
availability reset  
availability dest=192.168.2.2  
availability port=23  
availability sample=20  
availability method=application  
availability eventstatus=alwaysready  
availability event=$failidx  
availability failevent=$changetofailidx  
availability passevent=$changetopassidx  
availability create
```

---

## MIB Module

The availability MIB is a subtree of networkharmoni (1977).

The availability subtree is shown in Figure 9.



*Figure 9. OID tree diagram of the availability MIB module*

This section provides a summary of the objects defined in the availability MIB module. For detailed information on all objects in the module, see the `availability-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB tables

The subagent provides a standard SNMP interface with control row semantics.

The module contains the following tables:

- A control table, `availabilityControlTable`
- A data table, `availabilityDataTable`
- A history table, `availabilityHistoryTable`

### Control table

The availability control table (`availabilityControlTable`) defines availability queries.

Each row represents one query, which defines the IP address and port queried, the type of query - either application of network availability, and the interval at which the query is performed.

Data generated by availability control rows is stored in the data table, then transferred to the history table at intervals set by the `availabilityControlHistoryInterval` object.

Table 44 lists the row objects in `availabilityControlTable`.

*Table 44. Availability control table (`availabilityControlTable`)*

Row object	Description
BucketsGranted	Indicates the number of rows <code>availabilityHistoryTable</code> granted by the subagent to this control row.
BucketsRequested	Sets the number of rows in <code>availabilityHistoryTable</code> requested for storing the data associated with this control row.
CreateTime	Stores the value of <code>sysUpTime</code> when this entry was last activated.
DataControl	Sets the monitoring status of this control row:  on(1) - Enables data collection  off(2) - Suspends data collection
DestAddress	Specifies the destination IP address of the availability monitor.
EventIndex	Identifies the event generated when the availability test defined by this control row fails. If the value of this object does not correspond to an entry in the <code>RMON eventTable</code> or is 0 then no event is generated.
EventStatus	Sets the event flow control for the events associated with this row:  eventReady(1) - A single event may be generated, after which the value of this object changes to <code>eventFired(2)</code> .  eventFired(2) - Disables event generation. No events may be generated until the object is modified to <code>eventReady(1)</code> or <code>eventAlwaysReady(3)</code> .  eventAlwaysReady(3) - Disables the flow control, allowing events to be generated without restriction. Using this setting is not recommended as it can result in high network traffic or other performance problems.

Table 44. Availability control table (availabilityControlTable) (continued)

Row object	Description
FailureEventIndex	Identifies the event generated when the result of the availability query changes from true(1) to false(2). If the value of this object does not correspond to an entry in the RMON eventTable or is 0 then no event is generated.
HistoryInterval	Sets the interval (in seconds) at which data table rows associated with this control row are moved into the history table. If the value of this object is 0 no history is recorded and the data simply accumulates in the data table.
Index	Uniquely identifies this control row.
Method	Sets the method of availability detection:  application(1) - Determines whether an application is listening on the port specified  network(2) - Determines whether the destination address is reachable via ICMP protocol
Owner	The owner of this row.
Port	Specifies the port number for the application method. The default value of this object is 7, which is the echo port.
SampleInterval	Sets the sampling interval (in seconds). Availability is tested at the beginning of each sampling interval. Setting this object is set to 0 means that only one test is performed and also sets the availabilityControlHistoryInterval object to 0.
Status	The SNMPv2 row status object controlling creation, activation and deletion of the control row.
SuccessEventIndex	Identifies the event generated when the result of the availability query changes from false(2) to true(1). If the value of this object does not correspond to an entry in the RMON eventTable or is 0 then no event is generated.
TurnOffEventIndex	Contains the RMON eventIndex of the event configured to deactivate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or is 0 then deactivation via an event is disabled.
TurnOnEventIndex	Contains the RMON eventIndex of the event configured to activate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or is 0 then activation via an event is disabled.

## Data table

The availability data table (availabilityDataTable) stores the result of each availability query.

Table 45 lists the row objects in availabilityDataTable.

Table 45. Availability data table (availabilityDataTable)

Row object	Description
Available	The result of the availability query:  true(1) - a response was received during the sample interval.  false(2) - no response was received by the end of the sample interval.

Table 45. Availability data table (availabilityDataTable) (continued)

Row object	Description
Percentage	The average number of successful connections since the last history interval expired, expressed as a percentage of all connections attempted.
ResponseTime	The average time taken for the availability query to return a positive response. A value of -1 means the query has not been performed yet or that the query failed. No response time is recorded for queries that fail.
Time	The value of sysUpTime at which the most recent availability test was started.

## History table

The availability history table (availabilityHistoryTable) contains sampled entries from the availabilityDataTable.

Results of availability queries are transferred to this table from the data table at an interval set by availabilityControlHistoryInterval.

Table 46. Availability history table (availabilityHistoryTable)

Row object	Description
IntervalStart	The value of sysUpTime at which the sample started.
Percentage	The number of successful connections in this history interval, expressed as a percentage of all connections attempted.
ResponseTime	The average time taken for the availability query to return a positive response. A value of -1 means the query has not been performed yet or that the query failed. No response time is recorded for queries that fail.
SampleIndex	Identifies the history table row.
Time	The value of sysUpTime at which the query was performed.

The scalar object availabilityFlushHistory indicates the state of the history table. Setting this object to flushing(2) removes all rows from the availabilityHistoryTable.

## Notification types

The availability MIB module defines notification types for events generated by the availability subagent.

The notification types implemented by availability MIB are shown in Figure 10 on page 71.



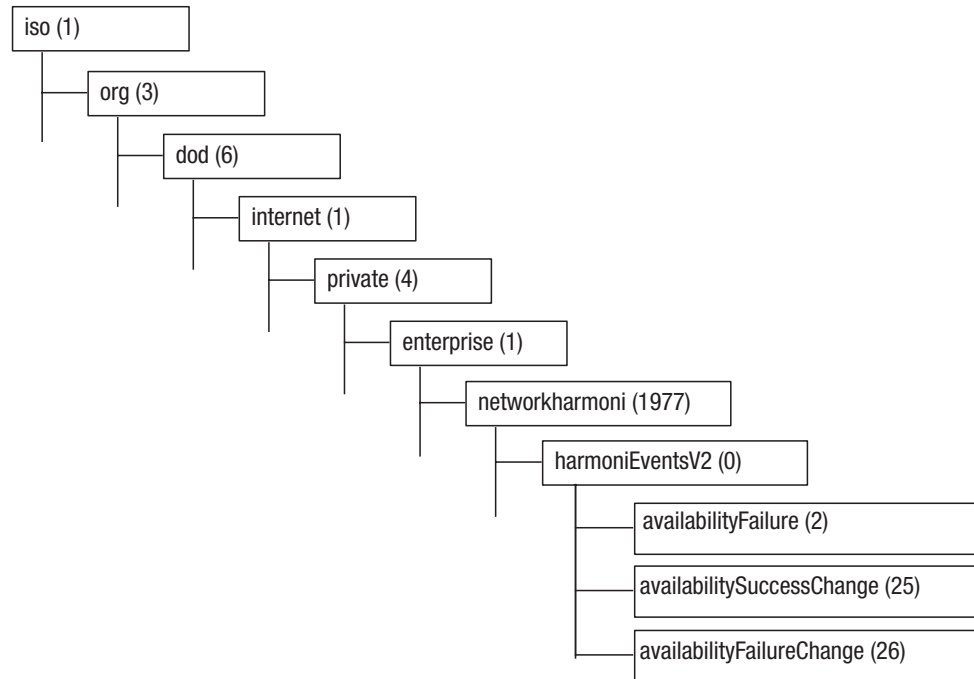


Figure 10. OID tree diagram for availability notification types

Table 47 lists the availability notification types and their variable bindings.

Table 47. Notification types for availability MIB

Notification type	Description
availabilityFailure	<p>The SNMP trap generated when an availability test fails.</p> <p>Variable bindings:</p> <p>availabilityControlDestAddress</p> <p>availabilityControlPort</p> <p>availabilityControlMethod</p> <p>availabilityDataTime</p> <p>availabilityDataPercentage</p>
availabilityFailureChange	<p>The SNMP trap generated when result of an availability test changes from available to unavailable.</p> <p>Variable bindings:</p> <p>availabilityControlDestAddress</p> <p>availabilityControlPort</p> <p>availabilityControlMethod</p> <p>availabilityDataTime</p> <p>availabilityDataPercentage</p> <p>availabilityDataAvailable</p>

Table 47. Notification types for availability MIB (continued)

Notification type	Description
availabilitySuccessChange	<p>The SNMP trap generated when result of an availability test changes from unavailable to available.</p> <p>Variable bindings:</p> <p>availabilityControlDestAddress</p> <p>availabilityControlPort</p> <p>availabilityControlMethod</p> <p>availabilityDataTime</p> <p>availabilityDataPercentage</p> <p>availabilityDataAvailable</p>

---

## Chapter 7. Crontab

The crontab subagent and the associated crontab MIB module enable events to be fired or commands to be executed at specific, scheduled times.

The functionality that this subagents provides is similar to that of the crontab command available on most UNIX operating systems. The subagent also provides a facility similar to the UNIX at command and is fully functional on all platforms supported by Netcool/SSM, including Windows platforms.

---

### Component files

The crontab subagent and MIB module are comprised of a set of component files.

Table 48 lists the crontab subagent and MIB module component files and their installed locations.

*Table 48. crontab component files*

File	Location	Description
crontab.dll (Windows) libcrontab.so/.sl (UNIX)	bin	Binary implementation of the crontab subagent.
crontab-mib.mib	mibs	crontab MIB definition document.
crontab.oid	config/oid	crontab subagent object identifier file.

---

### Guidelines

Use the event time to emulate the behavior of the UNIX crontab command, or a combination of start time and period to emulate the UNIX at command.

To load the subagent, use the command:  
subagent load crontab

### Creating schedules

Create schedules using the control table (crontabControlTable).

#### Procedure

To create a schedule:

1. Create a control row in crontabControlTable by setting crontabControlStatus to CreateAndWait(5).
2. Set either crontabControlStartTime and crontabControlPeriod, or crontabControlEventTime according to the required schedule.
3. Set crontabControlEvent to the index of an RMON event row to be triggered at each schedule point.
4. If required, set crontabControlExecutionCmd to the command to execute at schedule points.
5. Activate the control row by setting crontabControlStatus to active(1).

## Results

At each schedule point, the command is executed and an event is generated.

## Event times

Event times set a recurring schedule time.

The event time defined by the `crontabControlEventTime` object consists of five whitespace-delimited fields (in order): minute, hour, day of month, month, day of week. Table 49 lists the range of values permitted in these fields.

*Table 49. Date and time fields for crontabControlCronEventTime*

Field	Allowed range
day of month	1-31
day of week	0-7, where both 0 and 7 represent Sunday.
hour	0-23
minute	0-59
month	1-12

The following constraints apply:

- Any field can contain the wildcard character \* which represents all values in the allowed range.
- Ranges are permitted. A range consists of two numbers separated by a hyphen -. For example, the range 9-11 when used as an hours entry specifies the hours 9, 10 and 11.
- Values can be specified as lists. A list is a comma-delimited set of numbers or ranges. For example, 1,3,4,6 and 1-3,6-9 are valid lists.
- Step values can be used in combination with ranges. Appending the sequence /x to a range (where x represents a number). For example, 0-23/2 represents the values 0, 2, 4 and so on, up to 22 (inclusive). Steps can also be used in combination with the wildcard character, so the expression \*/3 represents the values 0,3,6 and so on. The general rule is that the expression n/x represents all values of n for which  $n \bmod x$  evaluates to 0.
- Two day fields are provided. The schedule occurs when at least one of these fields matches the current day.
- If the date or time specified do not exist (for example, 31 February) the schedule will never occur.
- A scheduled time and date may occur twice due to changes from summer time to winter time.
- Names are permitted in the month and day of week fields. Names values consist of the first three letters of the name of the day or month (for example, mon for Monday and jan for January). Names are not case-sensitive; that is, Mon, MON and moN all evaluate to Monday.
- If the specified event time is not valid, the subagent sets the control row status to notReady.

**Note:** The scheduled day can be specified by both of the fields day of month and day of week. If both fields are restricted (that is, if neither field contains a wildcard), the schedule consists of all values indicated by the two fields. For example, the schedule 30 4 1,15 \* 5 defines a schedule of 4:30am on the 1st and 15th day of each month as well as every Friday.

## Start time

Start times set the initial schedule point and operate in combination with a period.

The date and time format specified by the `crontabControlStartTime` object is `MMDDHHMM[[CC]YY]`. Table 50 describes this format and the allowed values.

Table 50. Date and time fields for `crontabControlStartTime`

Field	Description	Allowed range
CC	2-digit century	00-99 (omitting this value indicates the current century)
DD	2-digit day of month	01-31
HH	2-digit hour of day	00-23
MM	2-digit month of year	01-12
MM	2-digit minute of hour	00-59
YY	2-digit year	00-99 (omitting this value indicates the current year)

For example:

- The value 072514482003 represents 2:48pm, 25 July 2003.
- The value 07251448 represents 2:48pm, 25 July of the current year.

**Note:** If the specified start time is not valid, the subagent sets the control row status to `notReady`.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The `crontab` commands create rows in the control table (`crontabControlTable`).

The general syntax of these commands is:

```
crontab property=value
crontab create property=value ...
crontab reset
```

Table 51 lists the properties supported in these commands.

Table 51. Configuration command parameters - `crontab`

Property	Type	Description	Sets MIB object
description	string	A description of the control row.	Description
event	int	The index of the event generated when the command finishes executing.	EventIndex

Table 51. Configuration command parameters - crontab (continued)

Property	Type	Description	Sets MIB object
eventstatus	enum	Event flow control:  alwaysready  fired  ready	EventStatus
eventtime	string	Sets the scheduled time for command execution or event firing in crontime mode. See "Event times" on page 74 for details about the time format for this parameter.	CronEventTime
executioncmd	string	The command to be executed at the scheduled time.	ExecutionCmd
period	int	Specifies the interval (in minutes) at which an event is generated after it is initially triggered.	Period
starttime	string	Sets the scheduled time for command execution or event firing in periodic mode. See "Start time" on page 75 for details about the time format for this parameter.	StartTime
type	enum	The event firing behavior:  crontime - Emulates the UNIX crontab command  periodic - Emulates the UNIX at command	CronType

## Examples

This example describes how to use the crontab subagent to create a scheduling window during which a monitoring task is active.

### Scheduling window

Create a scheduling window for the availability subagent and use the availabilityControlDataControl control row object to enable and disable the data collection performed by that subagent. When each control row's data control is set to on(1) the availability subagent is able to collect data and generate events; when the object has the value off(2) the subagent is effectively disabled.

The window created for the availability subagent is open from 8:00am to 8:00pm, Monday to Friday. During this time the subagent will be able monitor data and generate events.

To implement this:

1. Load the rmonc, datactrl and crontab subagents.
2. Create two event rows in RMON. One event sets availabilityControlDataControl to on(1); the other sets it to off(2).
3. Configure the datactrl subagent to 'switch on' or 'switch off' the availability subagent via the availabilityControlDataControl object. The RMON event rows created in the previous step provide the triggering events for datactrl. The filter expression .\* instructs datactrl to operate on all control rows in the availability MIB.

4. Create a crontab control row that schedules when availability is enabled. This row 'opens' the window at 8:00am every weekday; that is, at minute 0 of hour 8 (8AM) on any day of the month and any month of the year, given that the day is in the range 1-5 (Monday - Friday). The event onevent provides the trigger for datactrl to 'switch on' the availability subagent.
5. Create a crontab control row that schedules when the availability subagent will be disabled. This row 'closes' the window at 8:00pm every weekday; that is, at minute 0 of hour 20 (8PM) on any day of the month and any month of the year, given that the day is in the range 1-5 (Monday - Friday). The event offevent provides the trigger for datactrl to 'switch off' the availability subagent.

The configuration commands required to implement this example are:

```
subagent load rmonc
subagent load datactrl
subagent load crontab

event reset
event type=snmp-trap
event description="Turn on net usage monitor window via crontab"
event create
onevent = $?
event reset
event type=snmp-trap
event description="Turn off net usage monitor window via crontab"
event create
offevent = $?

datactrl reset
datactrl object=$availabilityControlDataControl.1
datactrl onevent=$onevent
datactrl offevent=$offevent
datactrl filter=.*
datactrl create

#Open window for net usage at 8am weekday mornings (event index $onevent)
crontab reset
crontab type=crontime
crontab eventtime="0 8 ** 1-5"
crontab event=$onevent
crontab eventstatus=alwaysready
crontab description="Open window for net usage at 8am every weekday morning"
crontab create

#Close window for net usage at 8pm weekday nights (event index $offevent)
crontab reset
crontab type=crontime
crontab eventtime="0 20 ** 1-5"
crontab event=$offevent
crontab eventstatus=alwaysready
crontab description="Close window for net usage at 8pm every weekday night"
crontab create
```

---

## MIB module

The crontab MIB is a subtree of networkharmoni(1977).

The crontab subtree is shown in Figure 11.

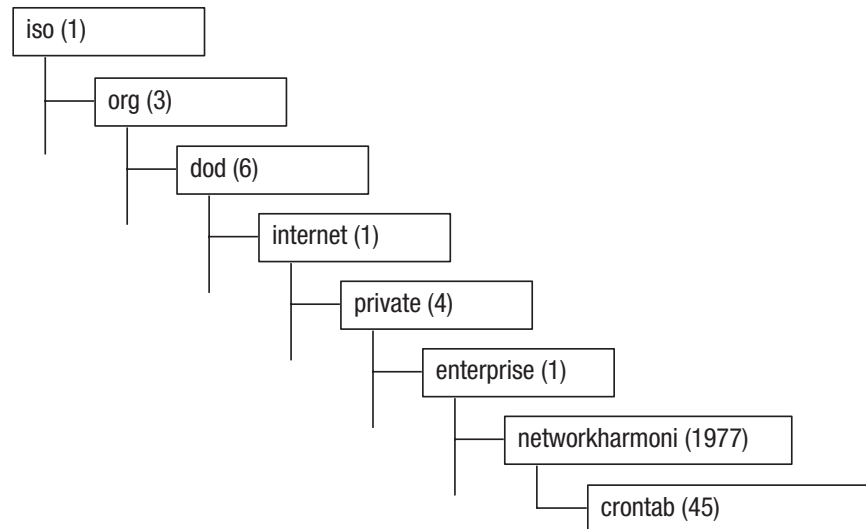


Figure 11. OID tree diagram of the crontab MIB module

This section provides a summary of the objects defined in the crontab MIB module. For detailed information on all objects in the module, see the `crontab-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB objects

The crontab MIB provides a standard SNMP interface and control row semantics.

The MIB module contains the following objects:

- `crontabCurrentDateTime`  
This object indicates the current local date and time.
- `crontabControlTable`  
Defines the actions performed by crontab.

### Control table

The crontab control table (`crontabControlTable`) defines the actions performed by crontab.

Each control row schedules an event to be fired or a command to be executed at a specific point in time, as defined in the control row. Table 52 on page 79 describes the objects provided in `crontabControlTable`.



Table 52. crontab control table (crontabControlTable)

Row object	Description
CronEventTime	<p>Sets the scheduled time for command execution or event firing if crontabControlCronType is set to crontime(1). This object consists of 5 space-delimited fields: minute, hour, day of month, month, day of week. For more details on the date and time format used in this object see “Guidelines” on page 73.</p> <p>This object has no effect if crontabControlCronType is set to periodicwithstart(2).</p>
CronType	<p>Sets the event firing behavior:</p> <p>crontime(1) - Emulates the behavior of the UNIX crontab command. The crontabControlCronEventTime object schedules the command execution and event firing. The values of crontabControlStartTime and crontabControlPeriod have no effect.</p> <p>periodicwithstart(2) - Emulates the behavior of the UNIX at command. The crontabControlStartTime and crontabControlPeriod objects schedule the command execution and event firing. The value of crontabControlCronEventTime has no effect.</p>
EventCount	Indicates the number of times that an event has been triggered by this control row. This object is read-only.
EventIndex	<p>Identifies the event to be generated when the command or program specified by crontabControlExecutionCmd completes execution successfully. This event is identified by the value of the RMON eventTable eventIndex object contained in the row that defines the event. The value of crontabControlEventIndex should have the same value as the event's eventIndex object.</p> <p>If the value of this object does not correspond to an entry in the RMON eventTable, no event will be generated. If no event is required, set this object to 0.</p>
EventStatus	<p>This object provides event flow control for any events generated upon successful execution of the program:</p> <p>eventReady(1) - A single event will be generated, after which the value of this object changes to eventFired(2).</p> <p>eventFired(2) - No events are generated. This enables the management station to respond to event notification by changing the value to eventReady(1) (or eventAlwaysReady(3)).</p> <p>eventAlwaysReady(3) - Disables event flow control, allowing events to be generated in an uncontrolled fashion. Disabling event flow control using this value is not recommended because it can result in high network traffic or lead to other performance problems.</p>
ExecutionCmd	Contains the command to be executed at the scheduled time. If this object is empty, no command is executed.
Period	<p>Specifies the interval (in minutes) at which an event is generated after the event is initially triggered (at the date and time specified by crontabControlStartTime).</p> <p>This object has no effect if crontabControlCronType is set to crontime(1). The default value of this object is 0.</p>

Table 52. crontab control table (crontabControlTable) (continued)

Row object	Description
StartTime	Sets the scheduled date and time for initial command execution or event firing if crontabControlCronType is set to periodicwithstart(2). The date and time specified by this object are encoded in the format MMDDHHMM[[CC]YY]. For more details about the format of this object, see “Guidelines” on page 73.  This object has no effect if crontabControlCronType is set to crontime(1).
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Status	SNMPv2 row status. Controls creation, activation and deletion of the control row.
Description	A description of the control row.

## Notification types

The crontab MIB defines a notification type for event generated by the subagent.

Figure 12 and Table 53 on page 81 describe the notification type.

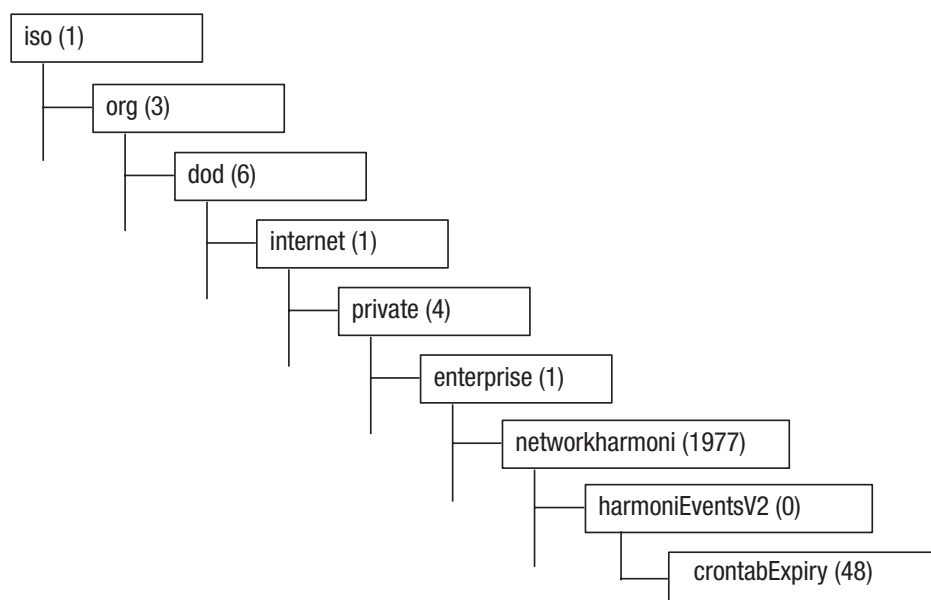


Figure 12. OID tree diagram for crontab notification types

Table 53. Notification types for crontab MIB

Notification type	Description
crontabExpiry	<p>Generated when the time and date specified by a crontab control row matches the current time.</p> <p>Variable bindings:</p> <p>crontabCurrentDateTime</p> <p>crontabControlCronType</p> <p>crontabControlCronEventTime</p> <p>crontabControlStartTime</p> <p>crontabControlExecutionCmd</p> <p>crontabControlPeriod</p> <p>crontabControlEventCount</p> <p>crontabControlDescription</p> <p>crontabControlOwner</p>



---

## Chapter 8. Desktop

The desktop subagent and the associated desktop MIB module provide a simple application, Desk Tool, that enables users of Windows machines to submit help desk queries.

The Desk Tool application maintains a profile of the user. The MIB module stores the user profiles, help desk queries and information about desktop activity.

---

### Component files

The desktop subagent and MIB module are comprised of a set of component files.

Table 54 provides a summary of the files associated with the desktop subagent and MIB module.

*Table 54. desktop component files*

File	Location	Description
desktop.dll	bin	Binary implementation of the desktop subagent.
desktop-mib.mib	mibs	desktop MIB definition document.
desktop.oid	config/oid	desktop subagent object identifier file.

---

### Using Desk Tool

Desk Tool enables users to send help desk queries.

To use the features of Desk Tool, load the desktop subagent using the command:  
subagent load desktop

### Starting Desk Tool

To send queries to the Help desk, and manage user profiles, you must first start Desk Tool.

#### Procedure

To start Desk Tool:

Select **Start > All Programs > Netcool > SSM > Desktop Help**.

The Desk Tool/Desktop Help icon appears in the System Tray, indicating that the Desk Tool is running.

## Sending help desk queries

Use Desk Tool to send queries to the Help desk.

### Procedure

To send a help desk query using Desk Tool:

1. In the System Tray, double-click the Desktop Help icon, or right-click it and select **Help**.

The Helpdesk Query window is displayed.

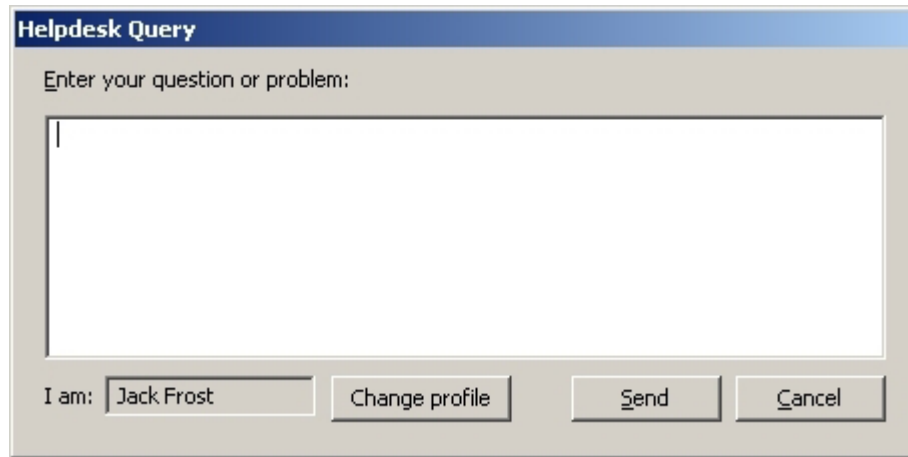


Figure 13. Helpdesk query window

2. In the text box, enter your request.
3. Click **Send**.

The query is transmitted.

## Changing the personal profile

The desktop MIB maintains details about the desktop user. These details are stored in the personal profile. You can view and update the personal profile using the Desk Tool.

### Procedure

To view or edit the personal profile:

1. In the System Tray, right-click the Desktop icon and select **Change Profile**.

The Personal Profile window is displayed.

**Personal Profile**

In order to provide effective technical support the helpdesk needs to know some information about you. If everything is correct already then just click OK.

**Personal Details**

User name: NETWORKHARMONI\ NETWORK

First name: Jack

Last name: Frost

Position: Senior Engineer

**Location**

Address:

City:

State:

Zip/Post code:

Country:

**Contact**

E-mail: cold@aircon.com

Work phone: 1234

Mobile phone: 4321

Home phone: 5678

Pager:

Fax: 8765

**Extra Information**

Air conditioning division

Autodetect OK

Figure 14. Personal profile window

2. Enter user details in the fields as required.  
Leave blank any fields that are not applicable.
3. Click **OK**.  
The Personal Profile window closes and any changes made to the user details are updated.

**Note:** To update the user details automatically, click **Autodetect**.

## MIB module

The desktop MIB is a subtree of networkharmoni (1977).

The desktop subtree is shown in Figure 15 on page 86.

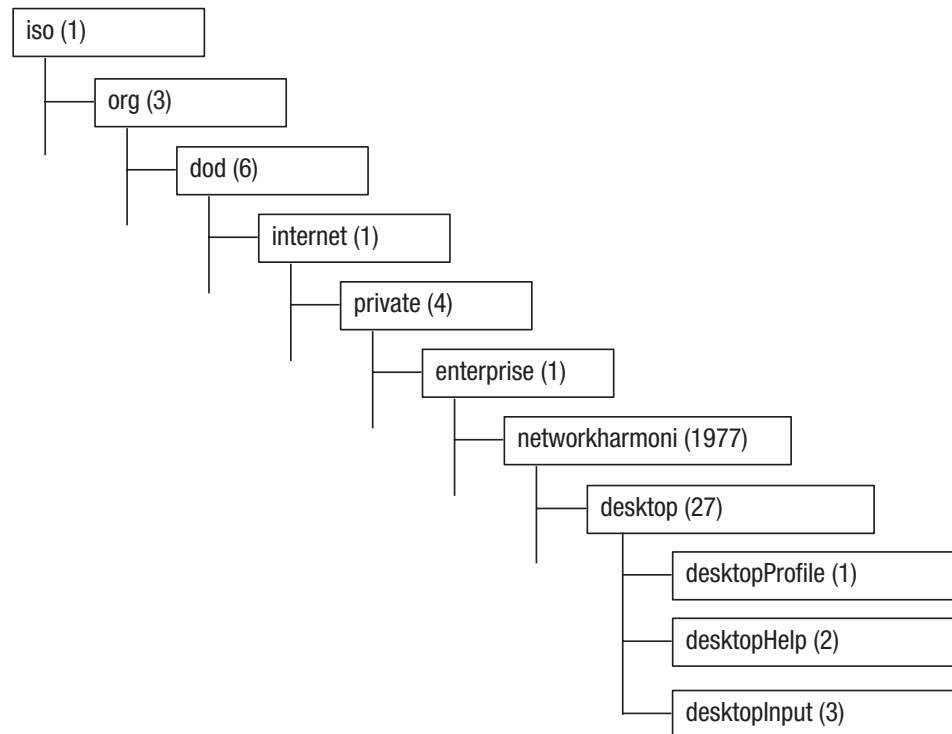


Figure 15. OID tree diagram of the desktop MIB module

This section provides a summary of the objects defined in the desktop MIB module. For detailed information on all objects in the module, see the `desktop-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB objects

The desktop MIB implements three groups of scalar objects for storing user profiles, helpdesk queries and desktop usage information.

### Desktop profile group

The Desktop Profile group (`desktopProfile`) contains personal details about the user currently logged onto the machine.

Some of the details are automatically determined by the desktop subagent, however users must enter most of them using Desk Tool. Fields that are not applicable may be left blank. If `desktopUserName` is blank, it indicates that there is no user logged in or that the desktop help tool has been disabled.

Table 55 lists the objects in this group.

Table 55. Desktop profile group objects

Object	Description
Address	The user's business address.
City	The city in which the user is employed.
Country	The country in which the user is employed.
Email	The user's email address.



Table 55. Desktop profile group objects (continued)

Object	Description
Extra	Any extra information required by the organization.
Fax	The user's fax number.
FirstName	The user's first name.
HomeDir	The path of the user's home directory (configured on the domain controller).
HomePhone	The user's home phone number.
LastName	The user's last name (surname).
LoginScript	The path of the user's login script (configured on the domain controller).
MobilePhone	The user's mobile (cell) phone number.
Pager	The user's pager number.
Path	The path of the user's Win32 profile directory.
Position	The user's position (occupation).
State	The state in which the user is employed.
UserName	The username (including domain) used to log onto the machine.
WorkPhone	The user's work (business) phone number.
Zip	The user's (business) zip code or post code.

## Desktop help group

The Desktop Help group (desktopHelp) provides objects associated with help desk requests, which users submit with Desk Tool.

Table 56 lists the objects in this group.

Table 56. Desktop help group objects

Object	Description
Community	Community to which desktop traps are sent.
Query	The last help query sent from the desktop help tool. Writing to this object sends a new query (trap).

## Desktop input group

The Desktop Input group (desktopInput) provides information about desktop activity.

Table 57 lists the objects in this group.

Table 57. Desktop input group objects

Object	Description
IdleTime	The amount of time in which no mouse or keyboard input has been detected on the main desktop. Resets to zero whenever a mouse or keyboard event occurs.

Table 57. Desktop input group objects (continued)

Object	Description
State	Provides an indication of whether the desktop is in use (with either mouse or keyboard):  unknown(0)  idle(1)  active(2)
Timeout	The amount of time in which no input must be received before the value of desktopInputState changes to idle(1).

## Notification types

The desktop MIB implements a number of notification types for the events generated by the subagent when a user logs on or off, changes personal details, or submit a help desk query.

These notification types are shown in Figure 16 and listed in Table 58 on page 89.

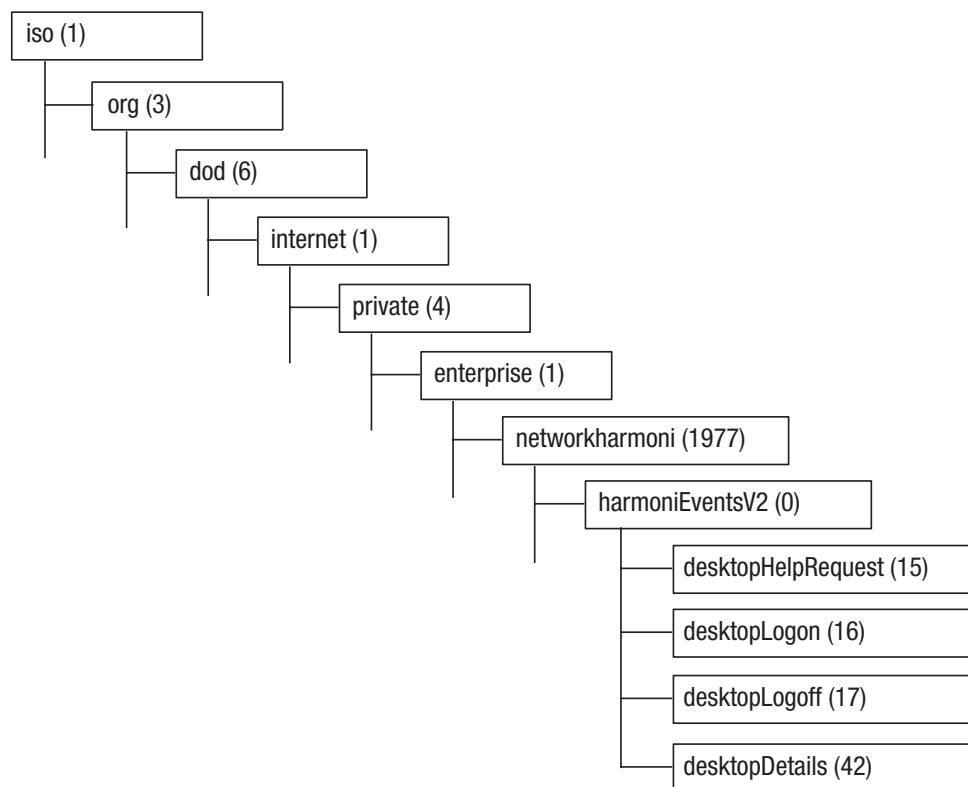


Figure 16. OID tree diagram for desktop notification types

Table 58. Notification types for desktop MIB

Notification type	Description
desktopDetails	Notification that the user has changed the personal profile (most of which must be read from the MIB).  Variable bindings:  desktopProfileUserName  desktopProfileFirstName  desktopProfileLastName  desktopProfileEmail
desktopHelpRequest	A request for support to the helpdesk.  Variable bindings:  desktopHelpQuery
desktopLogoff	Notification that a user has just logged off the machine.  Variable bindings:  desktopProfileUserName  desktopProfileFirstName  desktopProfileLastName  desktopProfileEmail
desktopLogon	Notification that a user has just logged onto the machine.  Variable bindings:  desktopProfileUserName  desktopProfileFirstName  desktopProfileLastName  desktopProfileEmail



---

## Chapter 9. File monitoring

The filemon subagent and its MIB module provide facilities for monitoring files and directories in a file system.

Using this subagent, you can monitor selected properties of files and directories, and generate events when those properties change.

---

### Component files

The filemon subagent and MIB module are comprised of a set of component files.

Table 59 lists the filemon subagent and MIB module component files, and their installed locations.

*Table 59. filemon component files*

File	Location	Description
filemon.dll (Windows) libfilemon.sl/.so (UNIX)	bin	Binary implementation of the filemon subagent.
filemon-mib.mib	mibs	MIB definition document.
filemon.oid	config/oid	filemon subagent object identifier file.

---

### Guidelines

Use the filemon subagent to monitor file systems and the files they contain.

To load the subagent, use the command:

```
subagent load filemon
```

### Monitoring procedure

Use the control table to monitor the files or directories in a file system.

#### Procedure

To set up a monitoring task:

1. Create a control row in fmControlTable by setting fmControlStatus to CreateAndWait(5).
2. Specify the file or directory that you wish to monitor.  
See "Selecting files and directories" on page 92 for more information about this step.
3. Set fmControlFileAttribute to the file system attribute that you wish to monitor.  
See "Selecting an attribute" on page 92 for more information about this step.
4. Set fmUpdateInterval to the interval (in seconds) at which you wish the nominated file system resources to be monitored.
5. Set fmEventIndex to the RMON event row index of the event triggered when a file or directory change occurs.

6. Activate the control row by setting `fmControlStatus` to `active(1)`.  
After each sample interval, the subagent updates the data table, `fmDataTable`, with results of any file system changes detected.

## Selecting files and directories

Using the control row objects, identify the file resources that you wish to monitor.

### Procedure

To select a file or directory:

1. Set `fmFileType` to the type of file system resources that you wish to monitor: files, directories, or both.
2. Set `fmControlDirectoryName` to the name of the base directory containing the file system resources that you wish to monitor.
3. Set `fmFileName` to the name of the file or directory that you wish to monitor. The name may contain a glob pattern.
4. If desired, set `fmControlNameFilter` and `fmControlFilterMode` to either include or exclude file or directory names matching a regular expression.

### Results

If you wish to monitor the attribute of a directory, set `fmControlDirectoryName` to the parent directory, and set `fmControlFileName` to the name of the directory itself. For example, to monitor the directory `/opt/IBM/db2`, set `fmControlDirectoryName` to `/opt/IBM/` and `fmControlFileName` to `db2`.

On UNIX platforms, you can use the `fmControlFollowSymLinks` object to force the subagent to monitor files and directories referenced by symbolic links.

### Monitoring remote file systems

If you wish to monitor remote file systems, you must enable remote file system monitoring on the master agent.

### About this task

To enable remote file system monitoring, set the `MonitorRemoteFS` agent inivar to the value `on`. Use the command:

```
set inivar MonitorRemoteFS=on
```

Remote file system monitoring is disabled by default.

## Selecting an attribute

The subagent enables you to monitor a range of file or directory attributes.

The `fmControlFileAttribute` object determines which attribute the control row monitors. If the monitored property changes then an event is triggered. Table 60 on page 93 lists the types of attribute that can be monitored and the corresponding `fmControlFileAttribute` object values.

Table 60. Monitored file and directory attributes

Attribute	Object value	Constraint
The size of a file, or in the case of directories, the total size of files in that directory.	size(1)	Monitors both increase and decrease in size.
An increase in the size of a file, or in the case of directories, the total size of files in that directory.	sizeIncreasing(2)	Monitors increase in size only.
A decrease in the size of a file, or in the case of directories, the total size of files in that directory.	sizeDecreasing(3)	Monitors decrease in size only.
The contents of a file or directory.	contents(4)	This uses a hash to monitor the file contents, so it can be computationally expensive.
The access time of a file.	accessTime(5)	Monitors any change in the access time.
The write time of a file or directory.	writeTime(6)	Monitors any change in the write time
The change time of a file or directory.	changeTime(7)	Available on UNIX platforms only.
The creation time of a file or directory.	createTime(8)	Available on Windows platforms only.
Any change in the security and permission attributes of a file or directory.	securityChange(9)	On Windows platforms, this mode monitors the access control list for a file or directory.  On UNIX platforms, it monitors the owner and group to which the file or directory belong, and the corresponding access permissions.
File creation.	newFiles(10)	Available for directories only.
File deletion	deletedFiles(11)	Available for directories only.
Number of files in a directory.	numberOfFiles(12)	Available for directories only.
Number of ticks (sysUpTime) since the monitored file or directory was last examined. Triggers an event on every sample, unless the file does not exist.	relativeUpdateTime(13)	
Triggers an event when a file or directory is created.	creation(14)	
Triggers an event when a file or directory is deleted.	deletion(15)	

## filemon inivars

The filemon subagent provides the inivars for configuring its operation.

The inivars are listed in Table 61.

Table 61. filemon subagent inivars

Inivar	Type	Description
MonitorRemoteFS	enum	To monitor remote file systems with filemon, set this inivar to on.  Default: off (filemon does not monitor remote file systems)

Table 61. filemon subagent inivars (continued)

Inivar	Type	Description
FmDataTableEnable	enum	To disable the fmDataTable and conserve system resources, set this inivar to off.  Default: on (fmDataTable is enabled)

For general information about inivars, see the *Netcool/SSM Administration Guide*.

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The filemon commands create rows in the filemon control table (fmControlTable).

The general syntax of these commands is:

```
filemon property=value
filemon create property=value ...
filemon reset
```

Table 62 lists the properties supported in these commands.

Table 62. Configuration command parameters - filemon

Property	Type	Description	Sets MIB object
datacontrol	enum	Data control: on - Enables data collection. off- Suspends data collection.	DataControl
description	string	A description of the control row.	Description
directoryname	string	The directory in which the target files and directories are located.	DirectoryName
event	int	The index of the RMON event generated when the monitored attribute changes.	Event
eventseverity	int	The severity code of any events generated.	EventSeverity
eventstatus	int	Event flow control: eventReady eventFired eventAlwaysReady	EventStatus



Table 62. Configuration command parameters - filemon (continued)

Property	Type	Description	Sets MIB object
fileattribute	enum	The file or directory attribute to be monitored: size sizeIncreasing sizeDecreasing contents accessTime writeTime changeTime createTime securityChange newFiles deletedFiles numberOfFiles relativeUpdateTime creation deletion	FileAttribute
filename	string	The name of the file or directory to monitor.	FileName
filetype	enum	The types of file system resource to monitor: file directory fileAndDirectory	FileType
filtermode	enum	Selects the file and directory name filter mode: include exclude	FilterMode
followsymlinks	enum	<i>UNIX only.</i> Determines whether symbolic links are followed: no yes	FollowSymLinks
maxfilematches	int	The maximum number of files and directories that the control row monitors.	MaxFileMatches
namefilter	string	A regular expression for filtering file or directory names.	NameFilter
updateinterval	int	The interval (in seconds) at which the monitored file system attribute is sampled.  If the interval is a negative number, that number is the negative of an event index. Only when the event is fired will filemon do a sample.  If the interval is 0, the control row collects only one sample.	Interval

---

## Examples

These examples demonstrate how to use the filemon subagent to monitor file systems.

### Monitor UNIX password file changes

Monitor the UNIX password file /etc/passwd at 10-minute intervals and generate an event if the file is written. Use the write time to detect any changes made to the file:

```
subagent load rmonc
subagent load filemon

event reset
event description="Password file changed"
event create type=snmp-trap community=public
pwdevent=$?

filemon reset
filemon description="UNIX password file change monitor"
filemon fileType=file
filemon directoryName="/etc/"
filemon fileName=passwd
filemon fileAttribute=writeTime
filemon updateInterval=600
filemon event=$pwdevent
filemon create
```

### Monitor UNIX home directory permission changes

Monitor the home directories on a UNIX system and generate an event if the permissions of any home directory change, excluding home directories for any accounts containing the name test:

```
subagent load rmonc
subagent load filemon

event reset
event description="Home directory permissions changed"
event create type=snmp-trap community=public
homedirevent=$?

filemon reset
filemon description="Home directory permission change monitor"
filemon fileType=directory
filemon directoryName="/home/"
filemon fileName=*
filemon nameFilter=".*test.*"
filemon filterMode=exclude
filemon fileAttribute=securityChange
filemon event=$homedirevent
filemon create
```

### Monitor creation and deletion

Monitor the document directory of an Apache Web server, C:\Program Files\Apache Group\Apache2\htdocs, and generate an event if any of its files or sub-directories is deleted or if any new files or sub-directories are created (that is, when the number of files and directories in the document directory changes):

```
subagent load rmonc
subagent load filemon
```

```

event reset
event description="Apache Web server file creation or deletion"
event create type=snmp-trap community=public
fileevent=$?

filemon reset
filemon description="Apache Web server file/dir create/delete monitor"
filemon fileType=fileAndDirectory
filemon directoryName="C:\\Program Files\\Apache Group\\Apache2\\"
filemon fileName=htdocs
filemon fileType=directory
filemon fileAttribute=numberOfFiles
filemon event=$fileevent
filemon create

```

## MIB module

The fileMon MIB is a subtree of networkharmoni (1977).

The fileMon subtree is shown in Figure 17.

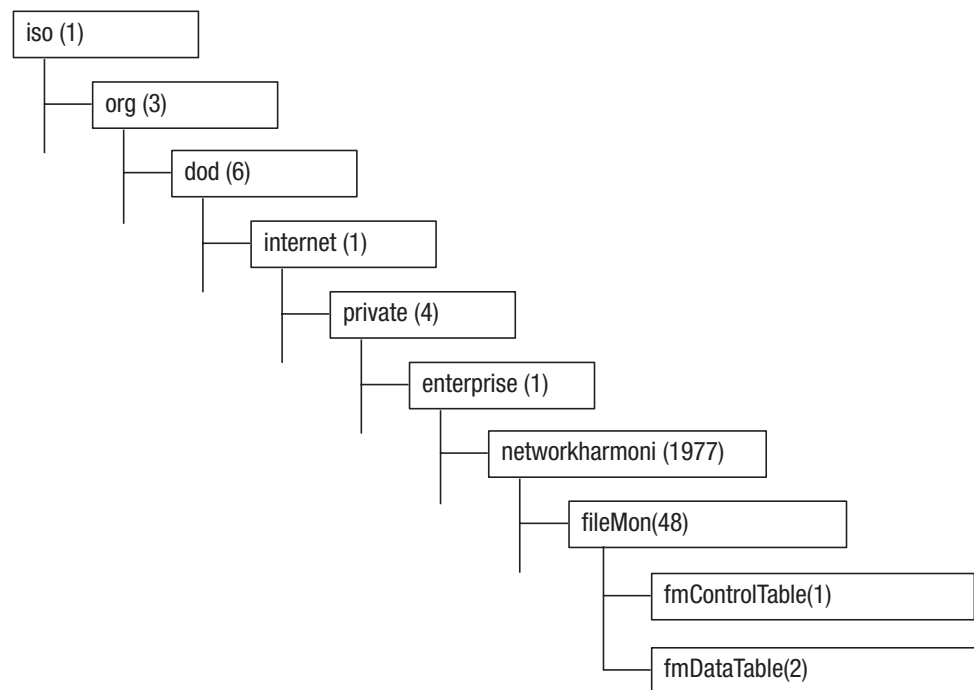


Figure 17. OID tree diagram for fileMon

This section provides a summary of the objects defined in the fileMon MIB module. For detailed information on all objects in the module, see the filemon-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB tables

The fileMon MIB provides a standard SNMP interface with control row semantics.

The MIB module contains:

- A control table, fmControlTable
- A data table, fmDataTable

### Control table

The control table (fmControlTable) defines file monitors.

Table 63 lists the objects in fmControlTable.

*Table 63. Control table (fmControlTable)*

Row object	Description
CreateTime	The value of sysUpTime when this entry was last activated.
DataControl	This object controls the monitoring status of the row: on(1) - Data collection is enabled. off(2) - Data collection is disabled.
Description	A description of the control row.
DirectoryName	The name of the directory in which the target files and directories are located.
Event	Event that is fired if the monitored file attribute changes. No event is fired if the value of this object is 0.
EventCount	The number of events that have been fired by this entry.
EventSeverity	Sets the severity level of notifications sent when the control row triggers an event. This object may contain any integer value.
EventStatus	RMON-style event throttle control: eventReady(1) - A single event will be generated, after which the value of this object changes to eventFired(2). eventFired(2) - No events are generated. This enables the management station to respond to event notification by changing the value to eventReady(1) or eventAlwaysReady(3). eventAlwaysReady(3) - Disables event flow control, allowing events to be generated in an uncontrolled fashion. Disabling event flow control using this value is not recommended because it can result in high network traffic or lead to other performance problems.
EventTime	The value of sysUpTime when an event was last fired.

Table 63. Control table (fmControlTable) (continued)

Row object	Description
FileAttribute	<p>Determines the file or directory property that the control row monitors. If the monitored property changes then an event will be triggered if the control row is configured to do so. The values of this object are:</p> <ul style="list-style-type: none"> <li>size(1)</li> <li>sizeIncreasing(2)</li> <li>sizeDecreasing(3)</li> <li>contents(4)</li> <li>accessTime(5)</li> <li>writeTime(6)</li> <li>changeTime(7)</li> <li>createTime(8)</li> <li>securityChange(9)</li> <li>newFiles(10)</li> <li>deletedFiles(11)</li> <li>numberOfFiles(12)</li> <li>relativeUpdateTime(13)</li> <li>creation(14)</li> <li>deletion(15)</li> </ul> <p>See “Selecting an attribute” on page 92 for more information about this object.</p>
FileMatchCount	The number of files and directories that the control row monitors.
FileName	<p>Name of the file or directory being monitored, relative to the directory specified by fmControlDirectoryName. This object may contain glob patterns. Glob expansion rules are platform dependent.</p> <p><b>Note:</b> Glob patterns do not match the files . and ..</p>
FileType	<p>Determines the types of file that the control row monitors:</p> <ul style="list-style-type: none"> <li>file(1) - Monitor files only.</li> <li>directory(2) - Monitor directories only.</li> <li>fileAndDirectory(3) - Monitor both files and directories.</li> </ul>
FilterMode	<p>Determines whether files and directories matching the regular expression defined in fmControlNameFilter are included or excluded files from the set of monitored files or directories:</p> <ul style="list-style-type: none"> <li>include(1) - Files matching the regular expression are included in the set of monitored files and directories.</li> <li>exclude(2) - Files matching the regular expression are excluded from the set of monitored files and directories.</li> </ul>
FollowSymLinks	<p><i>UNIX only.</i> Determines whether files referenced using symbolic links are monitored:</p> <ul style="list-style-type: none"> <li>yes(1) - Files referenced by symbolic links are monitored.</li> <li>no(2) - Files referenced by symbolic links are not monitored.</li> </ul>
Index	Uniquely identifies the row.
MaxFileMatches	Specifies the maximum number of files and directories that the control row monitors. When the limit is reached, any further matches are ignored.

Table 63. Control table (fmControlTable) (continued)

Row object	Description
NameFilter	Specifies a regular expression that is applied to the filenames returned by the fmControlFileName glob. Files and directories matching this regular expression are either included or excluded in the set of monitored files and directories according to the value of the fmControlFilterMode object.  The regular expression is matched against the base name specified in fmControlFileName and does not include the directory name specified in fmControlDirectoryName.
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Status	SNMPv2 row status. Controls creation, activation and deletion of the control row.
UpdateInterval	Specifies the interval (in seconds) at which the monitored file system attribute is sampled.  Default: 300

## Data table

The data table (fmDataTable) stores the results generated by file monitors defined in the control table.

Each data table row contains the monitor results for a single monitored file or directory. Only those objects relevant to the type of monitor are populated.

Data table rows are indexed by a combination of control row index and file or directory name. Table 64 lists the objects in fmDataTable.

Table 64. Data table (fmDataTable)

Row object	Description
AccessTime	When the control row object fmControlFileAttribute is set to accessTime(5), this object stores the time at which the monitored resource was last accessed.
Attribute	Indicates the file attribute that is monitored by the corresponding control row.
ChangeTime	<i>UNIX only.</i> When the control row object fmControlFileAttribute is set to changeTime(7), this object stores the time at which the file was last changed.
Contents	When the control row object fmControlFileAttribute is set to contents(4), this object stores the hash value of the contents of the monitored file.
CreateTime	<i>Windows only.</i> When the control row object fmControlFileAttribute is set to createTime(8), this object stores the time at which the file was created.
DeletedFileName	When the control row object fmControlFileAttribute is set to deletedFiles(11), this object stores the name of the deleted file that triggered the most recent file deletion event.
EventCount	The number of times that the monitored attribute of the resource has changed.
EventTime	The value of sysUpTime when the monitored attribute last changed.

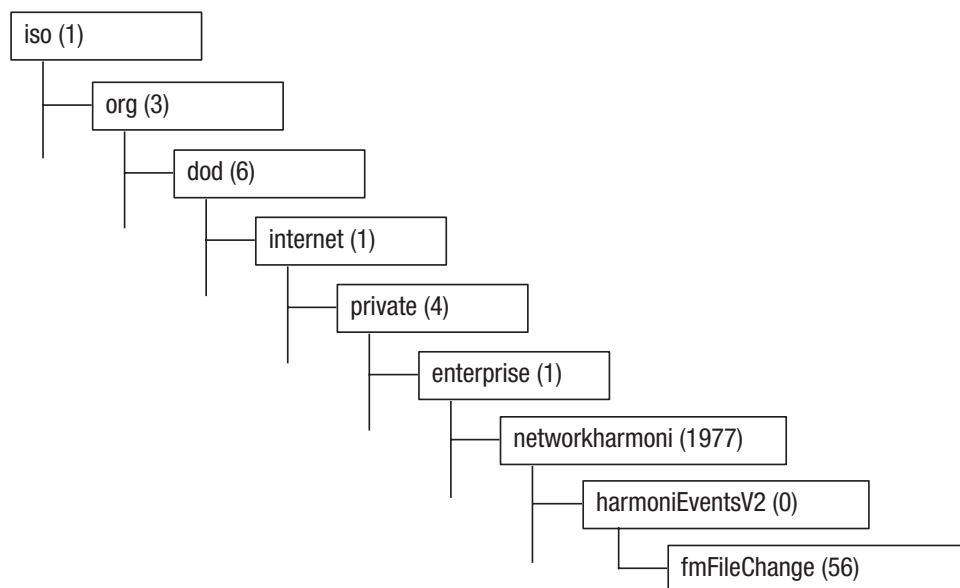
Table 64. Data table (fmDataTable) (continued)

Row object	Description
Name	Name of the monitored file or directory. Paths are relative to fmControlDirectoryName.
NewFileName	When the control row object fmControlFileAttribute is set to newFiles(10), this object stores the name of the newly created file that triggered the most recent file creation event.
NumberFiles	When the control row object fmControlFileAttribute is set to numberOfFiles(12), this object stores the number of files contained in the monitored directory.  If the control row object fmDataType has the value file(1)—that is, if the monitored resource is a file—this object has the value 0.
RelativeUpdatetime	When the control row object fmControlFileAttribute is set to relativeUpdateTime(13), this object stores the number of ticks since the monitored file or directory was last examined.
Security	When the control row object fmControlFileAttribute is set to securityChange(9), this object stores the current security attributes of the monitored resource, including the username, groups and access permissions.
Size	When the control row object fmControlFileAttribute is set to size(1), sizeIncreasing(2), or sizeDecreasing(3), this object stores the size of the monitored resource. <b>Note:</b> This is not guaranteed to be an increasing value; it may decrease over time.
Type	Type of resource monitored:  file(1) - The monitored resource is a file.  directory(2) - The monitored resource is a directory.
WriteTime	When the control row object fmControlFileAttribute is set to writeTime(6), this object stores the time at which the monitored resource was last written to.

## Notification types

The fileMon MIB implements the fmFileChange notification type.

Figure 18 on page 102 shows the location of this type.



*Figure 18. OID tree diagram for fileMon notification types*

The fmFileChange notification type is associated with events generated by the filemon subagent when it detects a change in a monitored file or directory. This notification type contains the following variable bindings:

- fmControlDirectoryName
- fmDataName
- fmControlDescription
- fmControlFileAttribute
- fmControlEventSeverity
- The fmDataTable object containing the value of the monitored attribute. For example, if a control row monitors the size attribute, this variable binding contains the fmDataSize object.



---

## Chapter 10. File transfer

The filetransfer subagent and the associated filetransfer MIB module facilitate reliable and secure file transfer from a remote server to a network node.

---

### Component files

The filetransfer subagent and MIB module are comprised of a set of components files.

Table 65 lists the filetransfer subagent and MIB module component files and their installed locations.

*Table 65. filetransfer component files*

File	Location	Description
filetransfer.dll (Windows) libfiletransfer.so/.sl (UNIX)	bin	Binary implementation of the filetransfer subagent.
filetransfer-mib.mib	mibs	filetransfer MIB definition document.
filetransfer.oid	config/oid	filetransfer subagent object identifier file.

---

### Guidelines

Use the filetransfer subagent to transfer files from a remote machine to an agent.

To load the subagent, use the command:

```
subagent load filetransfer
```

The subagent does not provide any configuration commands. Use SNMP set commands or the SNMP Query dialog in the MIB Explorer to configure file transfer.

### Inivars

The filetransfer subagent provides a set of inivars that configure its operation.

Table 66 lists the filetransfer subagent inivars.

*Table 66. filetransfer subagent inivars*

Inivar	Type	Description
FileDescriptorLimit	int	<p>Sets the number of file descriptors available to the filetransfer subagent. If no value is specified, then the operating system default value is used. For example, on Solaris systems the default value is 256.</p> <p>The valid range of this variable is 256-20480. If it has a value outside this range, the value is bound to the range limits, that is, a value less than 256 is bound to 256 and a value greater than 20480 is bound to 20480.</p>

Table 66. filetransfer subagent inivars (continued)

Inivar	Type	Description
FileTransferDataDir	string	Sets the default directory used for file transfer when the value of the fileTransferTableFileBase object is data. The directory specified by this inivar is relative to the Netcool/SSM bin directory. The variable has no default value.
FTThreads	int	<p>Sets the number of concurrent file transfers that the filetransfer subagent can perform. The default value is 32 in Netcool/SSM version 2.x and 1 in later versions.</p> <p>If the number of threads allocated is never reached, the impact is negligible; nevertheless, avoid specifying unnecessarily large values. If there are more jobs than threads, the jobs are queued and then serviced when resources become available.</p> <p>The value of this variable is bound to its range limits. That is, in the range 1-256, a value less than 1 is interpreted as 1 and a value greater than 256 is interpreted as 256.</p>

For general information about inivars, see the *Netcool/SSM Administration Guide*.

## Examples

This example demonstrates how to use the filetransfer subagent.

### Transferring a file

Transfer the configuration file `nuports.cfg` located in the directory `data/agent/config/` on the server `admin.mynet.com` to an agent via HTTP (using the default port 80), overwriting the agent's current copy of the file, which is located in the `config` directory. Remote access to `admin.mynet.com` requires authentication with the username `admin` and the password `admin`.

The subagent configuration commands required to perform this operation are:

```
subagent load filetransfer
snmp set $ftFileControl.1 i 1
snmp set $ftFileBase.1 i 1
snmp set $ftFilePathAndName.1 s /nuports.cfg
snmp set $ftFileMode.1 i 1
snmp set $ftFileServerAddr.1 s admin.mynet.com
snmp set $ftFileRemotePath.1 s /data/agent/config/nuports.cfg
snmp set $ftFileControl.1 i 2
```

---

## MIB module

The filetransfer MIB is a subtree of networkharmoni (1977).

The filetransfer subtree is shown in Figure 19.

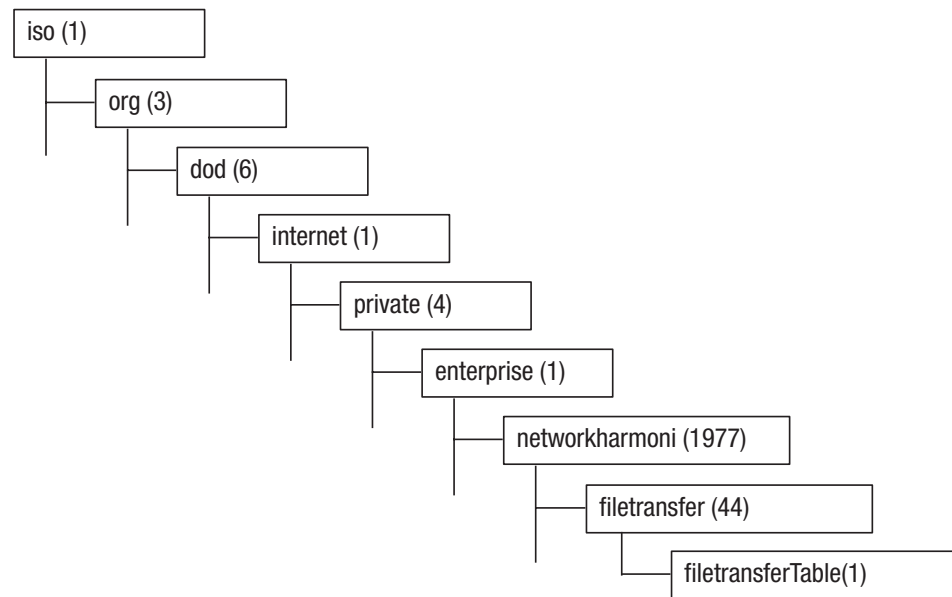


Figure 19. OID tree diagram of the filetransfer MIB module

This section provides a summary of the objects defined in the filetransfer MIB module. For detailed information on all objects in the module, see the filetransfer-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

### MIB tables

The filetransfer MIB provides one table for transferring files to agents.

#### File transfer table

The file transfer table (filetransferTable) defines the configuration and control settings necessary in transferring files to an agent.

Table 67 on page 106 describes the objects provided in the filetransferTable.

Table 67. File transfer table (filetransferTable)

Row object	Description
Base	<p>Selects a base directory that, in combination with the ftFilePathAndName object, defines the directory on the agent's host where the transferred file will be stored:</p> <p>cfg(1) - Selects the agent's config directory</p> <p>bin(2) - Selects the agent's bin directory</p> <p>data(3) - Selects the directory defined by the FileTransferDataDir variable in the init.cfg file</p> <p>When you set this object, the subagent checks whether the file specified by the combination of the ftFileBase and ftFilePathAndName objects already exists. If the file exists, the subagent reports its CRC and size.</p>
Control	<p>Controls the activation of the file transfer operation:</p> <p>none(1) - No action is taken</p> <p>get(2) - Starts the file transfer process using the information defined in the other objects of this row</p> <p>delete(3) - Deletes the file specified by the combination of the ftFileBase and ftFilePathAndName objects from the host machine</p> <p>The default value of this object is none(1). The ftFileBase, ftFilePathAndName, ftFileMode, ftFileServerAddr, ftFileRemotePort and RemotePath objects must be set before you select the value get(2).</p>
CRC	<p>Displays the CRC of the file referenced by the ftFileBase and ftFilePathAndName objects. The agent updates this value when a file transfer has completed successfully, or when the ftFileBase or ftFilePathAndName objects are set to point to an existing file.</p>
Mode	<p>Sets the file transfer mode:</p> <p>http(1) - Selects HTTP transfer protocol without SSL, provided that the selected HTTP service supports HTTP without SSL.</p> <p>https(2) - Specifies transfer using HTTP over SSL, provided that the HTTP service supports HTTP over SSL.</p> <p>The default value of this object is https(2).</p>
Owner	<p>Indicates the owner/creator of the filetransferTable row.</p>
PathAndName	<p>Defines the path and filename that, in combination with the Base object, defines the full path and filename under which the file will be stored on the agent's host. The maximum length of this object is 127 characters.</p> <p>When you set this object, the agent checks whether the file specified by the combination of the ftFileBase and ftFilePathAndName objects already exists. If the file exists, the agent reports its CRC and size.</p>
RemotePass	<p>Defines the password supplied for authentication on the remote server during file transfer. Only Basic Authentication is supported. The maximum length of this object is 127 characters.</p>
RemotePath	<p>Specifies the path of the file to be transferred. The maximum length of this object is 127 characters.</p>
RemotePort	<p>Defines the remote port to connect to for file transfer. The valid range of this object is 0 to 65535 and its default value is 80.</p>

Table 67. File transfer table (filetransferTable) (continued)

Row object	Description
RemoteUser	Defines the username supplied for authentication on the remote server during file transfer. Only Basic Authentication is supported. The maximum length of this object is 127 characters.
RespTime	Indicates the date and time when the file transfer was completed.
ServerAddr	Sets the IP address or server name of the HTTP server from which the file will be transferred. Before you start the file transfer by setting the ftFileControl object to get(2), this object must contain either a valid IP address or domain name of the server.
Size	Indicates the size of the file referenced by the ftFileBase and ftFilePathAndName objects. The agent updates this value when a file transfer has completed successfully, or when the ftFileBase or ftFilePathAndName objects are set to point to an existing file.
Status	Indicates the status of the current file transfer process:  idle(1) - No activity related to the file transfer.  transferring(2)- The file is currently being transferred to the agent's host.  transferred(3) - The file has been successfully stored on the agent's host.  failed(4)- The file transfer request was unsuccessful.  The default value of this object is idle(1).



---

## Chapter 11. Generic alarm

The genalarm subagent and the associated genAlarm MIB module provide a generic facility for threshold monitoring and alarm generation.

Using the genalarm subagent you can perform the following monitoring activities:

- Monitor 32- and 64-bit scalar variables or table objects.

The behavior of genalarm is based on the RMON alarm and event group [RFC 1757] with the exception that it allows you to specify a single OID for monitoring or to use wildcards to specify a group of OIDs.

- Generate an event when a threshold condition is held for a specified period.

You can specify an interval in seconds over which the data is sampled and compared with the rising and falling thresholds.

- Specify the alarm generation mechanism.
- Configure the information reported in a notification.

Using the variable binding functionality you can customize a notification to capture the variable data you require.

**Note:** Threshold monitoring is only available for 32-bit values.

---

### Component files

The genalarm subagent and MIB module are comprised of a set of component files.

Table 68 lists the genalarm subagent and MIB module component files and their installed locations.

*Table 68. genalarm component files*

File	Location	Description
genalarm.dll (Windows) libgenalarm.so/.sl (UNIX)	bin	Binary implementation of the genalarm subagent.
genalarm-mib.mib	mibs	genalarm MIB definition document.
genalarm.oid	config/oid	genalarm subagent object identifier file.

---

### Guidelines

Use the genAlarm control table (genAlarmControlTable) to configure threshold monitoring and alarm generation.

#### Before you begin

To load the subagent, use the command:

```
subagent load genalarm
```

#### Procedure

To create a simple threshold monitor:

1. Create a row in `genAlarmControlTable` by setting `genAlarmControlStatus` to `CreateAndWait(5)`.
2. Set `genAlarmControlVariable` to the OID of the object whose value you wish to monitor.
3. Set `genAlarmControlAlarmMode` to the type of threshold monitoring you wish to perform.
4. Set the `typeThreshold`, `typeDuration`, and `typeEventIndex` objects appropriate to the type of alarm mode that you selected.
5. Activate the control row by setting `genAlarmControlStatus` to `active(1)`. At each sample interval, the subagent compares the target MIB object's value to the selected threshold values, and generates an event if a threshold violation occurs.

## Alarm modes

A number of alarm modes are available for threshold monitoring.

The `genAlarmControlAlarmMode` object controls the method of alarm generation. This section describes the various alarm modes and provides diagrams that show when a rising alarm event (RA) or falling alarm event (FA) will occur in a series of sample periods. The dot symbols in each diagram indicate the start of a sample period.

### Hysteresis

The Hysteresis alarm mode operates as follows:

- Generates a rising alarm when the sample value first becomes greater than or equal to the rising threshold. Subsequent rising alarms will only be generated once the sample value becomes less than or equal to the falling threshold.  
If the value of the `genAlarmControlRisingDuration` object is non-zero, the rising alarm is only generated once the sample value has satisfied the rising alarm condition for the period (in seconds) set by this object.
- Generates a falling alarm whenever the sample value becomes less than or equal to the falling threshold and a rising alarm was previously generated.  
If the value of the `genAlarmControlFallingDuration` object is non-zero, the falling alarm is only generated once the sample value has satisfied the falling alarm condition for the period (in seconds) set by this object.

Figure 20 on page 111 demonstrates the operation of the hysteresis alarm mode with `genAlarmControlRisingDuration` and `genAlarmControlFallingDuration` set to zero.



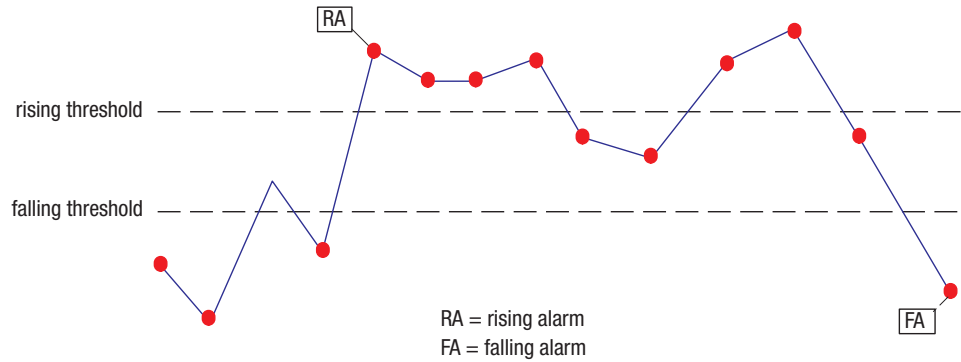


Figure 20. Hysteresis alarm mode

## Rising Continuous

The Rising Continuous alarm mode operates as follows:

- Generates a rising alarm for each sample value that is greater than or equal to the rising threshold.  
If the value of the `genAlarmControlRisingDuration` object is non-zero, the rising alarm is only generated once the sample value has satisfied the rising alarm condition for the period (in seconds) set by this object.
- Generates a single falling alarm when the sample value first becomes less than or equal to the falling threshold after a rising alarm was previously generated.

The `genAlarmControlStartupAlarm` and `genAlarmControlFallingDuration` objects do not apply to the Rising Continuous alarm mode.

Figure 21 demonstrates the operation of the Rising Continuous alarm mode with `genAlarmControlRisingDuration` set to zero.

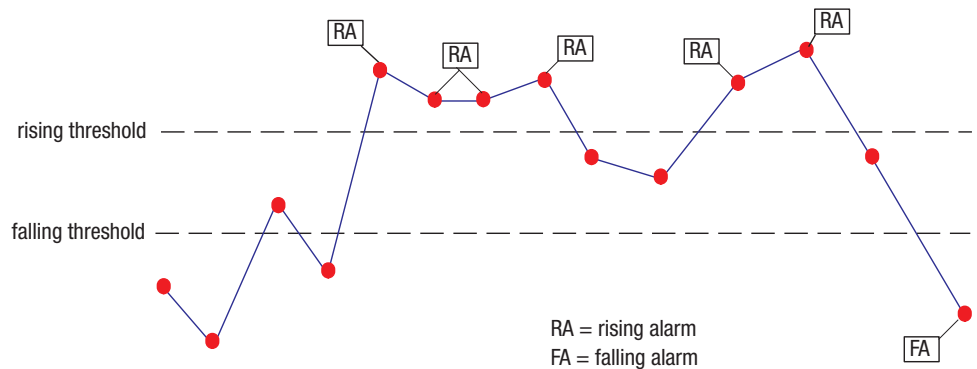


Figure 21. Rising Continuous alarm mode

## Single Edge

The Single Edge alarm mode operates as follows:

- Generates a rising alarm when the sample value becomes greater than or equal to the rising threshold.

If the value of the `genAlarmControlRisingDuration` object is non-zero, the rising alarm is only generated once the sample value has satisfied the rising alarm condition for the period (in seconds) set by that object.

- Generates a falling alarm event when the sample value becomes less than or equal to the falling threshold and a rising event was previously generated.

If the value of the `genAlarmControlFallingDuration` object is non-zero, the falling alarm is only generated once the sample value has satisfied the falling alarm condition for the period (in seconds) set by this object.

**Note:** A falling alarm can only occur after a rising alarm.

The `genAlarmControlStartupAlarm` object does not affect the Single Edge alarm mode. The first possible point at which a rising alarm can be generated is at the end of the first sample interval.

Figure 22 demonstrates the operation of the Single Edge alarm mode with `genAlarmControlRisingDuration` and `genAlarmControlFallingDuration` set to zero.

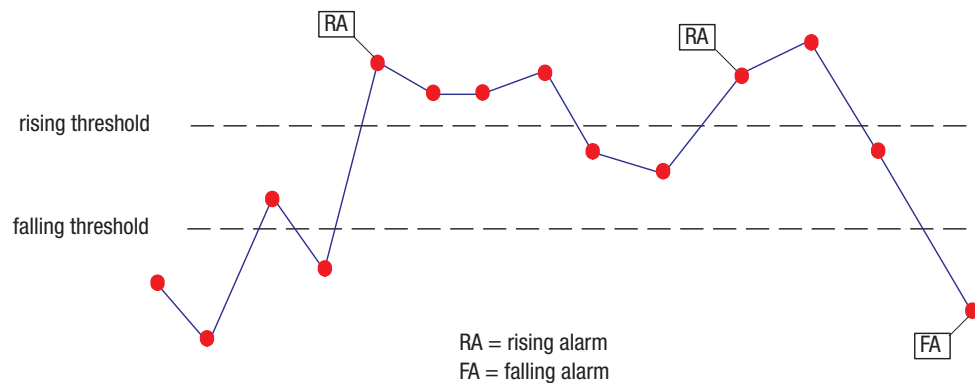


Figure 22. Single Edge alarm mode

Figure 23 on page 113 demonstrates the operation of the Single Edge alarm mode with `genAlarmControlRisingDuration` set to a value equivalent to three sample intervals and `genAlarmControlFallingDuration` set to zero.

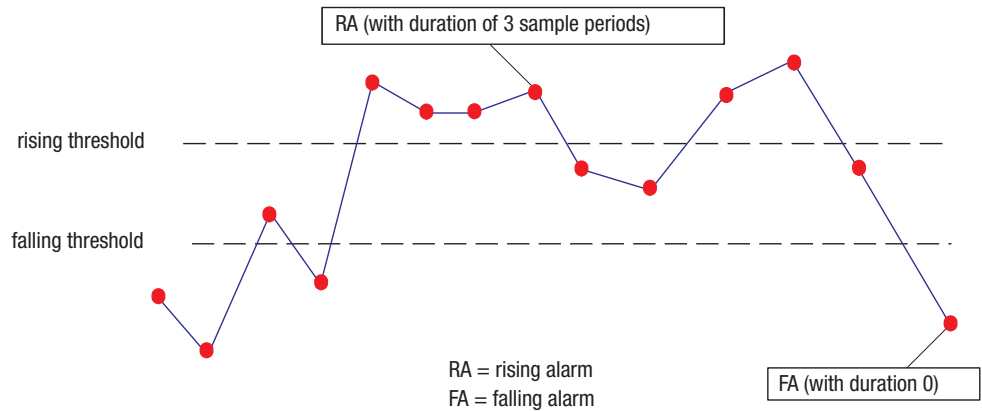


Figure 23. Single Edge Alarm mode (non-zero rising duration)

## Falling Continuous

The Falling Continuous alarm mode operates as follows:

- Generates a falling alarm for each sample value that is less than or equal to the falling threshold.  
If the value of the `genAlarmControlFallingDuration` object is non-zero, the falling alarm is only generated once the sample value has satisfied the falling alarm condition for the period (in seconds) set by this object.
- Generates a single rising alarm when the sample value first becomes greater than or equal to the rising threshold after a falling alarm was previously generated.

The `genAlarmControlStartupAlarm` and `genAlarmControlRisingDuration` objects do not apply to the Falling Continuous alarm mode.

Figure 24 demonstrates the operation of the Falling Continuous alarm mode with `genAlarmControlFallingDuration` set to zero.

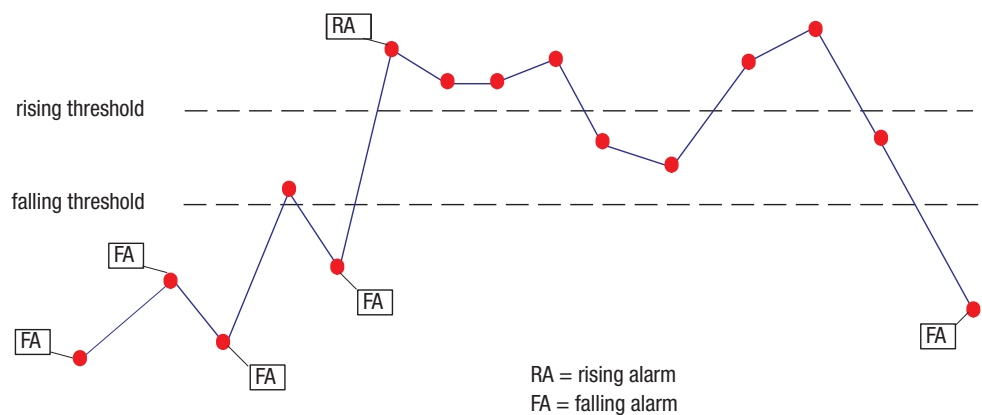


Figure 24. Falling Continuous alarm mode

## Match

The Match alarm mode generates a match alarm event for each sample equal to the match value. A single match alarm is generated whenever the sample equals the match value.

Figure 25 demonstrates the operation of the Match alarm mode.

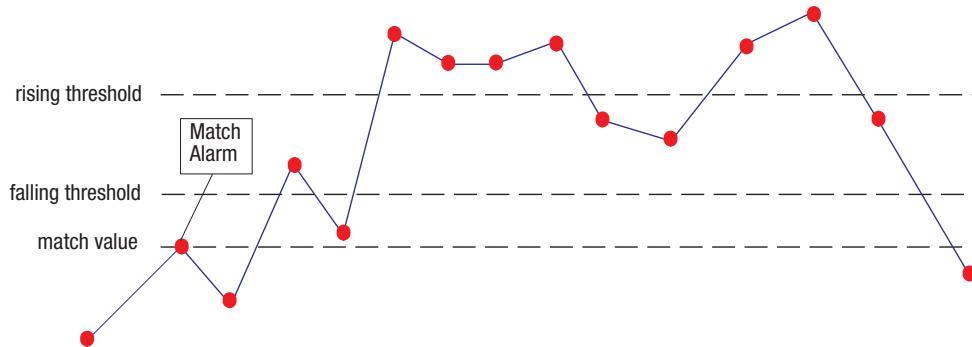


Figure 25. Match alarm mode

## Push

The Push alarm mode generates a push alarm event in every sample period. No comparison operation is performed and an alarm is always generated.

## Monitoring multiple instances

The subagent can monitor the values of multiple instances of an object.

To monitor multiple instances of either scalar or table objects, use wildcards in the `genAlarmControlVariable` object. A wildcard is denoted by either a question mark character (?) or an asterisk character (\*). The ? wildcard represents a single sub-object identifier and the \* wildcard represents a variable length object identifier.

The rules for specifying OIDs in the `genAlarmControlVariable` object are as follows:

- Only numeric characters or the dot character (.) are permitted.
- The first character must be a dot character (.).
- The single character wildcard (?) may be used in place of any sub-OID.
- The multiple character wildcard (\*) may be used to represent any number of sub-OIDs but may only be used in the last sub-OID position.
- The last character in the OID must not be a dot character (.).

An OID containing wildcards must resolve to an object of type Integer, Counter, Gauge or Time Ticks.

The following examples demonstrate valid syntax for the Variable field.

```
$historyControlBucketsRequested.*  
$appflowConnHistOutLastPacketArrivalDelay.1.?.80.216.239.33.100.?.*  
$appflowConnHistInvalidTransaction.????164.109.16.115.?.203.63.54.132  
$appflowConnHistInvalidTransaction.????164.109.16.115.?.203.63.54.132
```

## Configuring the data sent in notifications

The variable binding table `genAlarmVarBindTable` enables you to configure the amount of information sent with notifications generated by the `genAlarm` subagent.

### Creating variable bindings

Variable bindings insert data into the notifications that the subagent sends when a threshold violation occurs.

Each row in `genAlarmVarBindTable` specifies the OID of one or more variables to be included in a notifications. This OID may include wildcards, however wildcards may only be specified for index fields.

**Note:** Only the asterisk (\*) wildcard may be used at the end of the OID for a variable binding. If the OID specified by the control variable object `genAlarmControlVariable` does not contain a wildcard, the variable binding cannot contain a wildcard either.

The wildcard (\*) at the end of a variable binding OID is replaced by the portion of the control variable OID string starting from the first wildcard. For example, if the variable binding OID is `.1.5.8.*` and the control variable OID is `.1.3.6.1.?.?.5.?.?.4.*`, the portion of the control variable's OID added to the variable binding is `?.?.5.?.?.4.*` and the actual OID of the variable binding is then `.1.5.8.?.?.5.?.?.4.*`.

### Referencing variable bindings

You can refer to variable bindings from within description objects, enabling you to create informative messages for threshold violation events.

The `genAlarmControlRisingDescription`, `genAlarmControlFallingDescription`, and `genAlarmControlMatchDescription` objects enable you to reference the value of variables bound to the notification generated when an alarm is triggered. To refer to a variable's value use the expression `$i`, where `i` is an index into the variable binding list associated with the notification. When the event is generated, these expressions are substituted with the value of the specified variable.

The value of `i` must be in the range  $[1, n]$ , where  $n$  represents the total number of variable bindings associated with the notification, which is determined as the number of variable bindings included in the notification type plus the number of variables indicated by the alarm's `genAlarmControlVarBindSize` object.

For example, the notification type `genAlarmRisingAlarm` provides 10 standard bindings, so the total number of variable bindings is  $10 + \text{genAlarmControlVarBindSize}$ . See "Notification types" on page 126 for details about the standard variable bindings provided in the notification type associated with each alarm mode.

**Note:** To specify the `$` character in description string objects, use `$$`.

## SNMPv1 compatibility

The genAlarm MIB module uses the Counter64 data type, so it is not compatible with SNMPv1. You should only use the genAlarm subagent with SNMPv2 and later.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The genAlarm commands create rows in the genAlarm control table (genAlarmControlTable).

The general syntax of these commands is:

```
genalarm property=value  
genalarm create property=value ...  
genalarm reset
```

Table 69 lists the properties supported in these commands.

*Table 69. Configuration command parameters - genalarm*

Property	Type	Description	Sets MIB object
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl
falldescr	string	The description sent with the falling alarm notification.	FallingDescription
fallduration	int	The amount of time (in seconds) that the sample value must be below the falling alarm threshold before a falling alarm is generated.	FallingDuration
fallevent	int	The index of the event generated by a falling alarm.	FallingEventIndex
fallseverity	enum	The severity level of the falling alarm event:  critical  info  severe  warning	FallingSeverity
fallthresh	int	The falling alarm threshold.	FallingThreshold
interval	int	The sample interval (in seconds).	SampleInterval
matchdescr	string	The description sent with the match alarm notification.	MatchDescription

Table 69. Configuration command parameters - genalarm (continued)

Property	Type	Description	Sets MIB object
matchduration	int	The amount of time (in seconds) that the sample value must equal the match value before a match alarm is generated.	MatchDuration
matchevent	int	The index of the event generated by a match alarm.	MatchEvent
matchseverity	enum	The severity level of the match alarm event:  critical  info  severe  warning	MatchSeverity
matchvalue	int	The match alarm value.	MatchValue
mode	enum	The alarm generation mode:  fallingContinuous  hysteresis  match  push  risingContinuous  singleEdge	AlarmMode
risedescrip	string	The description sent with the rising alarm notification.	RisingDescription
riseduration	int	The amount of time (in seconds) that the sample value must exceed the rising alarm threshold before a rising alarm is generated.	RisingDuration
riseevent	int	The index of the event generated by a rising alarm.	RisingEventIndex
riseseverity	enum	The severity level of the rising alarm event:  critical  info  severe  warning	RisingSeverity
risethresh	int	The rising alarm threshold.	RisingThreshold
startup	enum	The alarm behavior at startup:  falling  rising  risingOrFalling	StartupAlarm

Table 69. Configuration command parameters - genalarm (continued)

Property	Type	Description	Sets MIB object
type	enum	The sampling method:  absolute  delta	SampleType
var	string	The monitored variable.	Variable
vardescr	string	A description of the monitored variable.	VariableDescription

## Variable bindings

The genalarmvb commands create rows in the configuration variable table and associate them with the next row created in the configuration table using the genalarm command.

The general syntax of these commands is

```
genalarmvb property=value
genalarmvb store property=value ...
genalarmvb reset
```

Table 70 lists the properties supported in these commands.

Table 70. Configuration command parameters - genalarmvb

Property	Type	Description	Sets MIB object
oid	string	<p>The object ID of the variable to be included in the notifications sent when an alarm is generated by the corresponding control row.</p> <p>To pick out part of an instance index, add [N] to the end of genAlarmVarBindOID. For example,</p> <p>If genAlarmControlVariable = .1.2.3.4.* matches: .1.2.3.4.5.6.7</p> <p>then:</p> <p>genAlarmVarBindOID = .9.8.7.* gets .9.8.7.5.6.7</p> <p>genAlarmVarBindOID = .9.8.7.*[0] gets .9.8.7.5</p> <p>genAlarmVarBindOID = .9.8.7.*[1] gets .9.8.7.6</p> <p>genAlarmVarBindOID = .9.8.7.*[2] gets .9.8.7.7</p> <p>genAlarmVarBindOID = .9.8.7.*[4] (any index out of range) gets .9.8.7.5.6.7</p>	genAlarmVarBindOID



---

## Examples

These following examples demonstrate how to use the `genalarm` subagent to perform specific tasks.

### Monitoring CPU usage

Monitor the value of the host machine's CPU usage metric (stored in the object `srSystemCPUUsage`) every 10 seconds and if it exceeds 80% for five minutes generate an alarm with the message CPU usage has been above *rising threshold* for five minutes (currently *value*). If the CPU usage then falls below 80% again for one minute, generate an 'all clear' notification with the message CPU usage has returned to normal (less than *falling threshold*).

The subagent configuration commands for creating this alarm are:

```
subagent load genalarm
```

```
event type=snmp-trap
event community=public
event description="CPU usage threshold violation"
event create
violationevent=$?
```

```
event description="CPU usage threshold normal"
event create
normalevent=$?
```

```
genalarm reset
genalarm var="$srSystemCPUUsage.0"
genalarm vardescr="CPU Usage"
genalarm interval=10
genalarm type=absolute
genalarm mode=singleEdge
genalarm risethresh=80
genalarm fallthresh=80
genalarm riseduration=300
genalarm fallduration=60
genalarm riseevent=$violationevent
genalarm fallevent=$normalevent
genalarm risedescr="CPU usage has been above $$7% for five minutes (currently $$6%)"
genalarm falldescr="CPU usage has returned to normal (less than $$7%)"
genalarm create
```

### Monitoring disk usage

Monitor the amount of space used on all fixed disks on the host machine every 30 seconds and if the amount used on any disk exceeds 90%, generate an alarm with the message *percent used* of storage on volume *disk name* is in use.

If the disk usage then falls below 85% again, generate an 'all clear' notification with the message Free storage level on volume *disk name* has returned to normal (less than *falling threshold*).

Bind the corresponding three objects `hrStorageDescr`, `hrStorageType` and `srStorageUsedMegabytes` for each fixed disk to the notifications sent in response to these alarm conditions.

The subagent configuration commands for creating this alarm are:

```

subagent load genalarm

event type=snmp-trap
event community=public
event description="Disk usage alarm"
event create
violationevent=$?

event description="Disk usage normal"
event create
normalevent=$?

genalarm reset
genalarm var="$srStorageLogicalUsedPercent.4.?"
genalarm vardescr="Disk usage"
genalarm interval=30
genalarm type=absolute
genalarm mode=hysteresis
genalarm risethresh=90
genalarm fallthresh=85
genalarm riseduration=0
genalarm fallduration=0
genalarm riseevent=$violationevent
genalarm fallevent=$normalevent
genalarm risedescr="$$6% of storage on volume $$11 is in use."
genalarm falldescr="Free storage level on volume $$11 has returned to normal
  (less than $$7%)."
```

```

genalarmvb reset
genalarmvb store oid="$hrStorageDescr.*"
genalarmvb store oid="$hrStorageType.*"
genalarmvb store oid="$srStorageUsedMegabytes.*"
genalarm create
```

## Monitoring gateway behavior

Monitor the host machine at one minute intervals and generate an alarm if the host commences operating as an IP gateway. The host is determined to be operating as a gateway when the value of the ipForwarding object is 1.

The subagent configuration commands for creating this alarm are:

```

subagent load genalarm

event type=snmp-trap
event community=public
event description="Status change"
event create
statusevent=$?
genalarm vardescr="IP Forwarding"
genalarm var="$ipForwarding.0"
genalarm interval=60
genalarm type=absolute
genalarm mode=match
genalarm matchvalue=1
genalarm matchevent=$statusevent
genalarm matchdescr="IP forwarding is enabled."
genalarm create
```

---

## MIB module

The genAlarm MIB is a subtree of networkharmoni (1977).

The genAlarm subtree is shown in Figure 26.

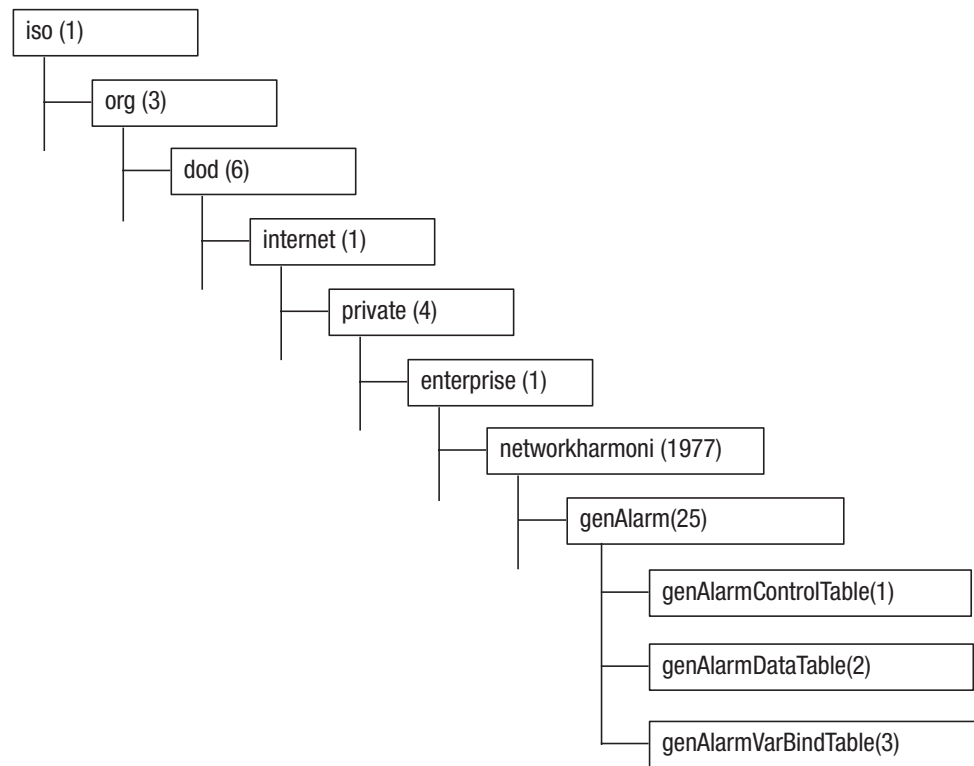


Figure 26. OID tree diagram of the genAlarm MIB module

This section provides a summary of the objects defined in the genAlarm MIB module. For detailed information on all objects in the module, see the `genalarm-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB tables

The genAlarm MIB provides a standard SNMP interface with control row semantics.

The MIB module contains:

- A control table, `genAlarmControlTable`
- A data table, `genAlarmDataTable`
- A variable binding table, `genAlarmVarBindTable`

## Control table

The control table (`genAlarmControlTable`) defines threshold monitors and alarms.

Each row defines an alarm and specifies the object or group of objects to be monitored, the alarm threshold for those objects and the events to be generated should the alarm be triggered. Table 71 lists the row objects in `genAlarmControlTable`.

Table 71. *genAlarm control table (genAlarmControlTable)*

Row object	Description
AlarmCount	The total number of rows containing a variable matching that specified by <code>genAlarmControlVariable</code> whose value generated an alarm during the previous sample period.
AlarmMode	<p><code>hysteresis(1)</code> - The alarm generation is identical to the RMON alarm and event generation</p> <p><code>risingContinuous(2)</code> - A rising alarm is generated in each sample interval in which the sample value is greater than or equal to the rising threshold. A single falling alarm is generated each time the sample value falls below the falling threshold.</p> <p><code>singleEdge(3)</code> - An alarm is generated in each sample interval in which the sample value exceeds either threshold.</p> <p><code>fallingContinuous(4)</code> - A falling alarm is generated in each sample interval in which the sample value is less than or equal to the falling threshold. A single rising alarm is generated each time the sample value rises above the rising threshold.</p> <p><code>matchMode(5)</code> - An alarm is generated in each sample interval in which the sample value is equal to the match value.</p> <p><code>pushMode(6)</code> - An alarm is generated in each sample interval.</p>
DataControl	<p>Sets the data collection status of the control row:</p> <p><code>on(1)</code> - Enables data collection</p> <p><code>off(2)</code> - Suspends data collection</p>
FallingDescription	Specifies a description sent in the notification generated for a falling alarm.
FallingDuration	Sets the duration (in seconds) for which the sample value must be less than or equal to the falling threshold before a falling alarm is generated. If the value of this object is 0, an event will be generated as soon as the current sample value is less than or equal to the falling threshold.
FallingEventIndex	Holds the index of the RMON event that is generated when the falling alarm is triggered. If the value of this object does not correspond to an entry in the RMON <code>eventTable</code> , or if its value is zero, no event will be generated.
FallingSeverity	Sets the severity code for the falling alarm.

Table 71. *genAlarm control table (genAlarmControlTable) (continued)*

Row object	Description
FallingThreshold	<p>Sets the falling threshold for the sampled value. When the value of the current sample is less than or equal to this threshold and the value of the previous sample was greater than the threshold, a falling event is generated.</p> <p>If the value of <code>genAlarmControlStartupAlarm</code> is equal to <code>fallingAlarm(2)</code> or <code>risingOrFallingAlarm(3)</code>, a single event is also generated if the value of the first sample taken after row activation is less than or equal to this threshold.</p> <p>After a falling event is generated, another such event will not be generated until the sampled value rises above this threshold and reaches the rising threshold.</p>
MatchDescription	Specifies a description sent in the notification generated for a match event.
MatchDuration	<p>Sets the duration (in seconds) for which the sample value must equal the value of <code>genAlarmControlMatchValue</code> before a match alarm is generated.</p> <p>If the value of this object is 0, an event will be generated immediately if the sample value equals the match value.</p>
MatchEventIndex	Holds the index of the RMON event that is generated when the match alarm is triggered. If the value of this object does not correspond to an entry in the RMON <code>eventTable</code> , or if its value is zero, no event will be generated.
MatchSeverity	Sets the severity code for the match alarm.
MatchValue	Sets a trigger value for match event. If the sample value is equal to this value, the match event indicated by <code>genAlarmControlMatchEventIndex</code> is generated.
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
RisingDescription	Specifies a description sent in the notification generated for a rising alarm.
RisingDuration	Sets the duration (in seconds) for which the sample value must be greater than or equal to the rising threshold before a rising alarm is generated. If the value of this object is 0, an event will be generated as soon as the current sample value is greater than or equal to the rising threshold.
RisingEventIndex	Holds the index of the RMON event that is generated when the rising alarm is triggered. If the value of this object does not correspond to an entry in the RMON <code>eventTable</code> , or if its value is zero, no event will be generated.
RisingSeverity	Sets the severity code for the rising alarm.

Table 71. *genAlarm control table (genAlarmControlTable) (continued)*

Row object	Description
RisingThreshold	<p>Sets the rising threshold for the sampled value. When the value of the current sample is greater than or equal to this threshold and the value of the previous sample was less than the threshold, a rising event is generated.</p> <p>If the value of <code>genAlarmControlStartupAlarm</code> is equal to <code>risingAlarm(1)</code> or <code>risingOrFallingAlarm(3)</code>, a single event is also generated if the value of the first sample taken after row activation is greater than or equal to this threshold.</p> <p>After a rising event is generated, another such event will not be generated until the sampled value falls below this threshold and reaches the falling threshold.</p>
SampleInterval	Sets the interval (in seconds) at which the monitored object's value is sampled and compared with the alarm thresholds.
SampleType	<p>Specifies how the sample value is derived from the monitored object's value for comparison against the alarm thresholds:</p> <p><code>absoluteValue(1)</code> - The sample value is the value of the monitored object.</p> <p><code>deltaValue(2)</code> - The sample value is determined as the difference between the monitored object's value at the last sample and that at the current sample.</p>
StartupAlarm	<p>Controls alarm behavior when the control row is first activated.</p> <p>If the first sample taken after row activation is greater than or equal to the rising threshold and this object has the value <code>risingAlarm(1)</code> or <code>risingOrFallingAlarm(3)</code>, then a single rising alarm is generated.</p> <p>If the first sample taken after row activation is less than or equal to the falling threshold and this object has the value <code>fallingAlarm(2)</code> or <code>risingOrFallingAlarm(3)</code>, then a single falling alarm is generated.</p>
Status	The status of the alarm entry. If the value of this object is not equal to <code>active(1)</code> , all associated rows in <code>genAlarmDataTable</code> are deleted by the agent.
VarBindSize	Sets the number of variables bound to the trap PDU sent when an alarm is generated. The variable bindings themselves are defined in <code>varBindTable</code> .
Variable	The OID of the object to be monitored. This OID may contain wildcards in index fields. Only variables that resolve to an ASN.1 primitive type of integer (INTEGER, Counter, Gauge, or TimeTicks) may be monitored.
VariableDescription	A description of the object being monitored.

## Data table

The data table (`genAlarmDataTable`) contains information about the monitored object and activation history of the alarms defined in the control table.

For each row in the control table, the data table contains one row for each object monitored. Data table rows are automatically generated for each alarm defined in the control table and are updated at every sample interval. Table 72 lists the row objects in `genAlarmDataTable`.

Table 72. Data table (`genAlarmDataTable`)

Row object	Description
Count	The total number of times the alarm has been triggered.
Instance	The OID of the monitored object.
LastViolationTime	The value of <code>sysUpTime</code> when the sample value last crossed a threshold.
Value	<p>Stores the value of the monitored object during the most recent sample.</p> <p>If the <code>genAlarmControlSampleType</code> object in the corresponding control row is set to <code>absoluteValue(1)</code>, the value of this object is the same as the value of the monitored object.</p> <p>If the <code>genAlarmControlSampleType</code> object is set to <code>deltaValue(2)</code>, the value of this object is the difference between the value of the monitored object during the current sample and that from the previous sample.</p> <p>The subagent compares the value of this object to the rising and falling thresholds, <code>genAlarmControlRisingThreshold</code> and <code>genAlarmControlFallingThreshold</code>, to evaluate alarm conditions.</p> <p>The value of the monitored object is updated at the end of a sampling period and remains available until the next period is complete.</p>
Value64	The 64-bit representation of <code>genAlarmDataValue</code> .

## Variable binding table

The variable binding table (`genAlarmVarBindTable`) specifies the OIDs of objects to be included in the notifications generated by alarms defined in the control table.

The number of variable bindings associated with a control row is determined by the `genAlarmControlVarBindSize` object. Table 73 lists the row objects in `genAlarmVarBindTable`.

Table 73. Variable binding table (`genAlarmVarBindTable`)

Row object	Description
Index	Uniquely identifies an entry in the variable binding table.
OID	The object identifier of the variable included in the notification. The OID may contain the * wildcard in the instance portion. The wildcards may only be specified for fields of type INDEX. If the object contains the ! character, the object will resolve to the <code>genAlarmDataValue64</code> variable that is appropriate for the alarm.

## Notification types

The genAlarm MIB defines notification types for events generated by the subagent.

The genAlarm notification types are shown in Figure 27 and listed in Table 74.

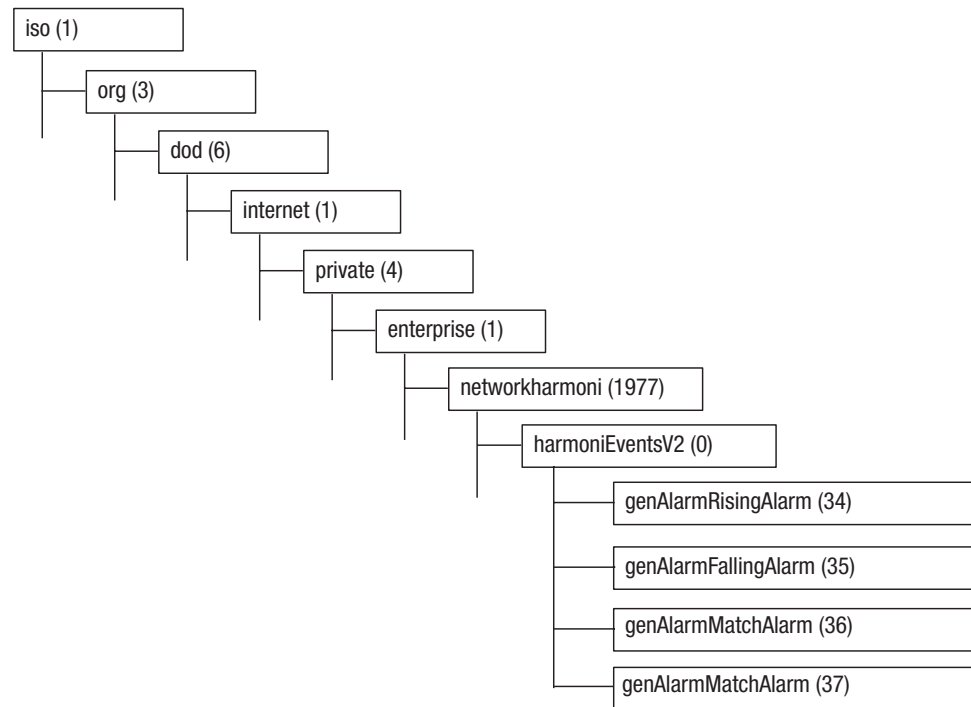


Figure 27. OID tree diagram for generic alarm notification types

Table 74. genAlarm notification types

Notification type	Description
genAlarmFallingAlarm	<p>Notification type for falling alarms.</p> <p>Variable bindings:</p> <p>genAlarmControlVariable</p> <p>genAlarmControlVariableDescription</p> <p>genAlarmDataInstance</p> <p>genAlarmControlSampleType</p> <p>genAlarmControlAlarmMode</p> <p>genAlarmDataValue</p> <p>genAlarmControlFallingThreshold</p> <p>genAlarmControlFallingDuration</p> <p>genAlarmControlFallingSeverity</p> <p>genAlarmControlFallingDescription</p>



Table 74. *genAlarm* notification types (continued)

Notification type	Description
genAlarmMatchAlarm	<p>Notification type for match alarms.</p> <p>Variable bindings:</p> <p>genAlarmControlVariable</p> <p>genAlarmControlVariableDescription</p> <p>genAlarmDataInstance</p> <p>genAlarmControlSampleType</p> <p>genAlarmControlAlarmMode</p> <p>genAlarmDataValue</p> <p>genAlarmControlMatchValue</p> <p>genAlarmControlMatchDuration</p> <p>genAlarmControlMatchSeverity</p> <p>genAlarmControlMatchDescription</p>
genAlarmPushAlarm	<p>Notification type for push alarms.</p> <p>Variable bindings:</p> <p>genAlarmControlVariable</p> <p>genAlarmControlVariableDescription</p> <p>genAlarmDataInstance</p> <p>genAlarmControlSampleType</p> <p>genAlarmControlAlarmMode</p> <p>genAlarmDataValue</p> <p>genAlarmControlMatchValue</p> <p>genAlarmControlMatchDuration</p> <p>genAlarmControlMatchSeverity</p> <p>genAlarmControlMatchDescription</p>

Table 74. *genAlarm* notification types (continued)

Notification type	Description
genAlarmRisingAlarm	<p>Notification type for rising alarms.</p> <p>Variable bindings:</p> <p>genAlarmControlVariable</p> <p>genAlarmControlVariableDescription</p> <p>genAlarmDataInstance</p> <p>genAlarmControlSampleType</p> <p>genAlarmControlAlarmMode</p> <p>genAlarmDataValue</p> <p>genAlarmControlRisingThreshold</p> <p>genAlarmControlRisingDuration</p> <p>genAlarmControlRisingSeverity</p> <p>genAlarmControlRisingDescription</p>

---

## Chapter 12. Host resources

Netcool/SSM can monitor all of the variables that are part of the Host Resource MIB (RFC1514). These include CPU, memory and disk usage and as well as installed software and its usage on the host system.

The hostres subagent and MIB module monitor attributes common to most Internet hosts including, for example, both personal computers and systems that run variants of UNIX. The host MIB module implements the RFC 1514/2790 standards.

---

### Component files

The hostres subagent and MIB module are comprised of a set of component files.

Table 75 lists the hostres subagent and MIB module component files and their locations.

*Table 75. hostres component files*

File	Location	Description
hostres.dll (Windows) libhostres.so/.sl (UNIX)	bin	Binary implementation of the hostres subagent.
hostres-mib.mib	mibs	hostres MIB definition document.
hostres.oid	config/oid	hostres subagent object identifier file.

---

### Guidelines

Use the hostres subagent and MIB module to collect system configuration and performance information from the host machine.

To load the subagent, use the command:

```
subagent load hostres
```

### Remote mount points

By default, the hosres subagent does not resolve the hrFSRemoteMountPoint object, which stores the name and address of servers from which file systems are mounted.

To enable resolution of this object, set the MonitorRemoteFS inivar to on.

### Host Resources object support

Host Resources objects are located in the subtree under the OID .1.3.6.1.2.1.25.

Certain objects are not available on some platforms. In some cases an object is available on a platform but derives from two of the standard objects to form a modified object with different semantics. Table 76 on page 130 provides a summary of the objects available on each platform.

Table 76. Host Resources supported objects

Sub-OID	MIB object	Win	Linux	Solaris	AIX	HP-UX
hrSystem Objects		%	X	%	%	%
1.1	hrSystemUptime	X	X	X	X	X
1.2	hrSystemDate	X	X	X	X	X
1.3	hrSystemInitialLoadDevice	X	X	X	X	X
1.4	hrSystemInitialLoadParameters	X	X			
1.5	hrSystemNumUsers	X	X	X	X	X
1.6	hrSystemProcesses	X	X	X	X	X
1.7	hrSystemMaxProcesses		X	X	X	X
hrStorage Objects		%	%	%	%	%
2.2	hrMemorySize	X	X	X	X	X
2.3.1.1	hrStorageIndex	X	X	X	X	X
2.3.1.2	hrStorageType	X	X	X	X	X
2.3.1.3	hrStorageDescr	X	X	X	X	X
2.3.1.4	hrStorageAllocationUnits	X	X	X	X	X
2.3.1.5	hrStorageSize	X	X	X	X	X
2.3.1.6	hrStorageUsed	X	X	X	X	X
2.3.1.7	hrStorageAllocationFailures					%
hrDevice Objects		%	%	%	%	%
3.2.1.1	hrDeviceIndex	X	X	X	X	X
3.2.1.2	hrDeviceType	X	X	X	X	X
3.2.1.3	hrDeviceDescr	X	X	X	X	X
3.2.1.4	hrDeviceID					
3.2.1.5	hrDeviceStatus	X	X	X	X	X
3.2.1.6	hrDeviceErrors					
3.3.1.1	hrProcessorFrwID					
3.3.1.2	hrProcessorLoad	X	X	X	X	X
3.4.1.1	hrNetworkIfIndex	X	X	X	X	X
3.5.1.1	hrPrinterStatus	X			X	
3.5.1.2	hrPrinterDetectedErrorState				X	
3.6.1.1	hrDiskStorageAccess	X	X	X	X	X
3.6.1.2	hrDiskStorageMedia	X	X	X	X	X
3.6.1.3	hrDiskStorageRemoveble	X	X	X	X	X
3.6.1.4	hrDiskStorageCapacity	X	X	X	X	X
3.7.1.1	hrPartitionIndex	X	X	X	X	X
3.7.1.2	hrPartitionLabel	X	X	X	X	X
3.7.1.3	hrPartitionID	X	X	X	X	X
3.7.1.4	hrPartitionSize	X	X	X	X	X
3.7.1.5	hrPartitionFSIndex	X	X	X	X	X
3.8.1.1	hrFSIndex	X	X	X	X	X
3.8.1.2	hrFSMountPoint	X	X	X	X	X

Table 76. Host Resources supported objects (continued)

Sub-OID	MIB object	Win	Linux	Solaris	AIX	HP-UX
3.8.1.3	hrFSRemoteMountPoint	X	X	X	X	X
3.8.1.4	hrFSType	X	X	X	X	X
3.8.1.5	hrFSAccess	X	X	X	X	X
3.8.1.6	hrFSBootable	X	X	X	X	X
3.8.1.7	hrFSStorageIndex	X	X	X	X	X
3.8.1.8	hrFSLastFullBackupDate					
3.8.1.9	hrFSLastPartialBackupDate					
hrSWRun Objects		%	%	%	%	%
4.1	hrSWOSIndex	X	X	X	X	X
4.2.1.1	hrSWRunIndex	X	X	X	X	X
4.2.1.2	hrSWRunName	X	X	X	X	X
4.2.1.3	hrSWRunID					
4.2.1.4	hrSWRunPath	X	X	%	%	%
4.2.1.5	hrSWRunParameters	X	X	X	X	X
4.2.1.6	hrSWRunType	X	X	X	X	X
4.2.1.7	hrSWRunStatus	X	X	X	X	X
hrSWRunPerf Objects		X	X	X	X	X
5.1.1.1	hrSWRunPerfCPU	X	X	X	X	X
5.1.1.2	hrSWRunPerfMem	X	X	X	X	X
hrSWInstalled Objects		%	%	%	%	%
6.1	hrSWInstalledLastChange	X	X	X	X	X
6.2	hrSWInstalledLastUpdateTime	X	X	X	X	X
6.3.1.1	hrSWInstalledIndex	X	X	X	X	X
6.3.1.2	hrSWInstalledName	X	X	X	X	X
6.3.1.3	hrSWInstalledID					
6.3.1.4	hrSWInstalledType		X		X	X
6.3.1.5	hrSWInstalledDate	X	X	X	X	X
6.3.1.5	hrSWInstalledDate	X	X	X	X	X
X indicates full support; % indicates partial support						

## hostres inivars

The hostres subagent provides inivars for configuring its operation.

The inivars are listed in Table 77.

Table 77. Host Resources subagent inivars

Inivar	Type	Description
HostresCallDiInit (Solaris only)	enum	Controls the execution of di_init. on - hrDiskStorage uses the system call di_init. off - hrDiskStorage does not call di_init.  Default: on

Table 77. Host Resources subagent inivars (continued)

Inivar	Type	Description
HostresProbeDisks (Solaris only)	enum	Controls collection of the DeviceStatus for disks. on - Enables collection of DeviceStatus for disks. off - Disables collection of DeviceStatus for disks. Default: on
HostresProbeEFI (Solaris only)	enum	Controls collection of EFI Partition Entries. on - EFI partition entries are monitored. off - EFI partition entries are not monitored. Default: on
MonitorRemoteFS	enum	Controls the resolution of the hrFSRemoteMountPoint object: off - The object is not resolved. on - The object is resolved.
MonitorPrinters	enum	Controls whether printers in hostres are monitored: off - Printers are not monitored on - Printers are monitored Default: off
SolarisDiskValue (Solaris only)	string	When hostres cannot automatically map device names, this inivar maps the strings found in kstat to the matching device in the libdevinfo tree. The format is:  SolarisDiskValue=kstat_device_class=devinfo_device_class [,kstat_device_class=devinfo_device_class,kstat_device_class=devinfo_device_class,...]  To find the values to map to, find the class name of the disks. 1. Look at hrDeviceDescr to get the kstat value. For example, in this description which contains the serial number of the device <b>sd0</b> : HITACHI H106060SDSUN600G, the <i>kstat device class</i> is SD. 2. Search the output of prtconf for the same serial number to get the devinfo value. For example,  <b>disk, instance #0</b> (driver not attached) Hardware properties: name='inquiry-revision-id' type=string items=1 value='A2B0' name='inquiry-product-id' type=string items=1 value='H106060SDSUN600G' name='inquiry-vendor-id' type=string items=1 value='HITACHI'  gives you a <i>devinfo device class</i> of disk.  In this example, SolarisDiskValue=sd=disk

For general information about inivars, see the *Netcool/SSM Administration Guide*.

## Device mapping (Linux only)

The hostres subagent provides a facility for mapping Linux block device numbers to hrStorageTypes values.

Device number mappings are defined in the file hrdevicetypes.csv, located in the Netcool/SSM config directory. You may modify this file to add support for additional Linux devices.

The format of each entry in hrdevicetypes.csv is:

*Linux block device major number, hrStorageTypes value [//comment]*

Whenever the hostres subagent is loaded it reads the contents of this file. If you make changes to the file, you must unload and reload the subagent for those changes to take effect.

### Example: Mapping the 129th to 144th SCSI devices

To add support for the 129th to 144th SCSI disk devices, identify the block number of this range of devices, 128, select the appropriate hrStorageTypes value, hrStorageFixedDisk (4), and add an entry to hrdevicetypes.csv.

The format of this entry is:

128, 4 // SCSI disk devices (128-143)

**Tip:** The official registry of Linux device numbers is located at <http://www.lanana.org/docs/device-list/>.

---

## MIB module

The host MIB is a subtree of mib-2(1).

The host subtree is shown in Figure 28 on page 134.

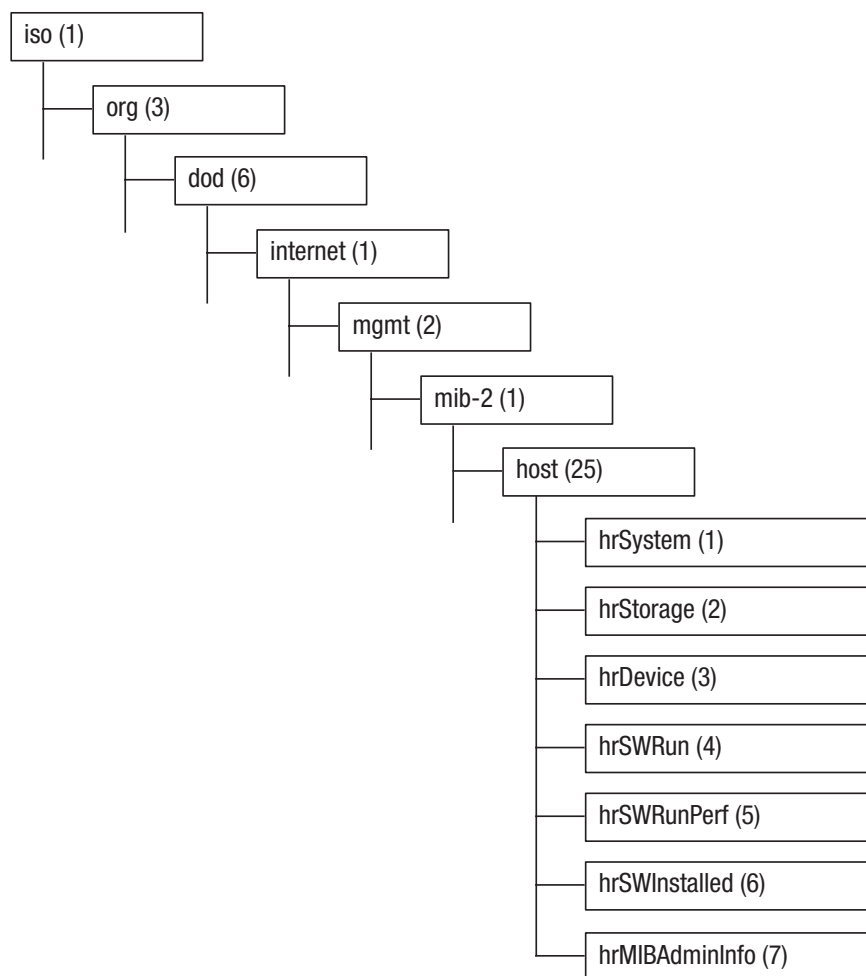


Figure 28. OID tree diagram of the host MIB module

This section provides a summary of the objects defined in the host MIB module. For detailed information on all objects in the module, see the `hostres-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB objects

The host MIB defines a uniform set of objects useful for the management of host computers. It defines objects that are common across many computer system architectures.

### Host resources system group

The Host Resources System group (`hrSystem`) provides host system details.



### **Host resources storage group**

The Host Resources Storage group (hrStorage) gives details of the logical storage areas.

### **Host resources device group**

The Host Resources Device group (hrDevice) is useful for identifying the devices on a system and diagnosing any problems with them. The hrDeviceTable contains general information about devices. Some devices also have their own table containing specific information relating to them.

### **Running software group**

The Host Resources Running Software group (hrSWRun) contains information about software that is currently running on the system. The hrSWRunTable contains an entry for each distinct piece of software that is either running on the system, or loaded into physical or virtual memory in preparation for running. This includes the host's operating system, device drivers, and applications.

### **Running software performance group**

The Host Resources Running Software Performance group (hrSWRunPerf) relates to the Running Software Group. Performance information is stored in the hrSWRunPerfTable, which contains an entry corresponding to each entry in the hrSWRunTable.

### **Installed software group**

The Host Resources Installed Software group (hrSWInstalled) contains information about software that is installed on the local host. The hrSWInstalledTable contains an entry for each piece of software installed in long-term storage. This table is useful in identifying installed software on a host, which is useful in diagnosing hardware/software incompatibility and version mismatches.

### **MIB administration**

Host Resources MIB Administration information (hrMIBAdminInfo) provides documentation only.



---

## Chapter 13. IP configuration

The ipconfig subagent and the associated ipconfig MIB module collect IP configuration information for individual network interfaces or entire networks.

---

### Overview

The ipconfig subagent and MIB module are comprised of a set of component files.

Table 78 lists the ipconfig subagent and MIB module component files and their installed locations.

*Table 78. ipconfig component files*

File	Location	Description
ipconfig.dll (Windows) libipconfig.so/.sl (UNIX)	bin	Binary implementation of the ipconfig subagent.
ipconfig-mib.mib	mibs	ipconfig MIB definition document.
ipconfig.oid	config/oid	ipconfig subagent object identifier file.

---

### Guidelines

Use the ipconfig subagent to monitor IP configuration information.

#### About this task

To load the subagent, use the command:  
subagent load ipconfig

---

### MIB module

The ipconfig MIB is a subtree of networkharmoni(1977).

The ipconfig subtree is shown in Figure 29 on page 138.

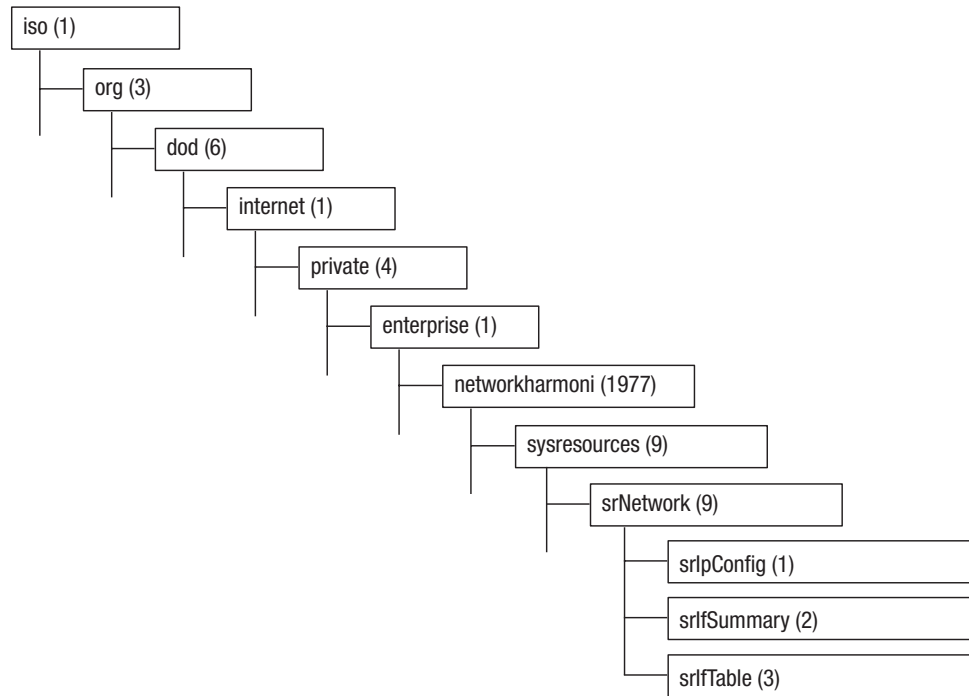


Figure 29. OID tree diagram of the ipconfig MIB module

This section provides a summary of the objects defined in the ipconfig MIB module. For detailed information on all objects in the module, see the ipconfig-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB objects

The ipconfig MIB provides network data for selected host machines and selected network interfaces.

### IP configuration group

The IP configuration group, srIpConfig, provides information about the IP configuration for the selected host.

Table 79 lists the objects in this group.

Table 79. Configuration group

Object	Description
DefaultGateway	The default gateway for this interface.
DnsDomain	The primary DNS domain suffix for this host.
DNSServer	The current, primary DNS server.
HostName	The primary name of this host.

Table 79. Configuration group (continued)

Object	Description
NodeType	Indicates the DHCP type of node:  none(0)  broadcast(1)  peertopeer(2)  mixed(3)  hybrid(4)
RouteEnabled	Indicates whether IP Routing is enabled on this host.
RouteNumber	A count of the number of IP routes configured. The number of rows in the MIB2 ipRouteTable.
TcpListenNumber	The number of TCP listening ports, which is determined as the number of rows in the MIB-2 tcpConnTable with state 'listen'.
UdpPortNumber	The number of UDP ports., which is determined as the number rows in the MIB-2 udpTable.
WinsEnabled	Indicates whether the WINS Proxy is enabled for this host.

## Interface summary group

The interface summary group, `srIfSummary`, provides a set of objects summarizing the state of the network interfaces.

Table 80 lists the objects in this group.

Table 80. Interface summary group objects

Object	Description
DownInterfaces	The number of network interfaces that are down.
InUtilization	Network utilization by inbound traffic to this host multiplied by 100. This utilization applies to all interfaces.
OutUtilization	Network utilization by outbound traffic from this host multiplied by 100. This utilization applies to all interfaces.
SummaryUpInterfaces	The number of network interfaces that are up.
Utilization	The sum of <code>srIfSummaryHostInUtilization</code> and <code>srIfSummaryHostOutUtilization</code> .

## Interface table

The interface table, `srIfTable`, provides additional network interface configuration information.

Each row in the table provides information about a network interface. Table 81 lists the objects in `srIfTable`.

Table 81. Interface table (srIfTable)

Row object	Description
Description	A description of this interface.
DHCPEnabled	Indicates whether this interface uses DHCP.

Table 81. Interface table (srlfTable) (continued)

Row object	Description
DHCPServer	The DHCP server of the interface if present.
HostInUtilization	Network utilization by inbound traffic to this host multiplied by 100.
HostOutUtilization	Network utilization by outbound traffic from this host multiplied by 100.
IPAddr	The primary IP address assigned to this interface.
LeaseExpired	The date when the DHCP lease will expire.
LeaseObtained	The date when the DHCP lease was obtained.
NetMask	The network Mask for this interface.
PhysAddr	The physical address of this interface.

## Notification types

The ipconfig MIB implements the srIpConfigIpChanged notification type.

This type is associated with events generated when the IP address of an interface changes. Its location is indicated in Figure 30.

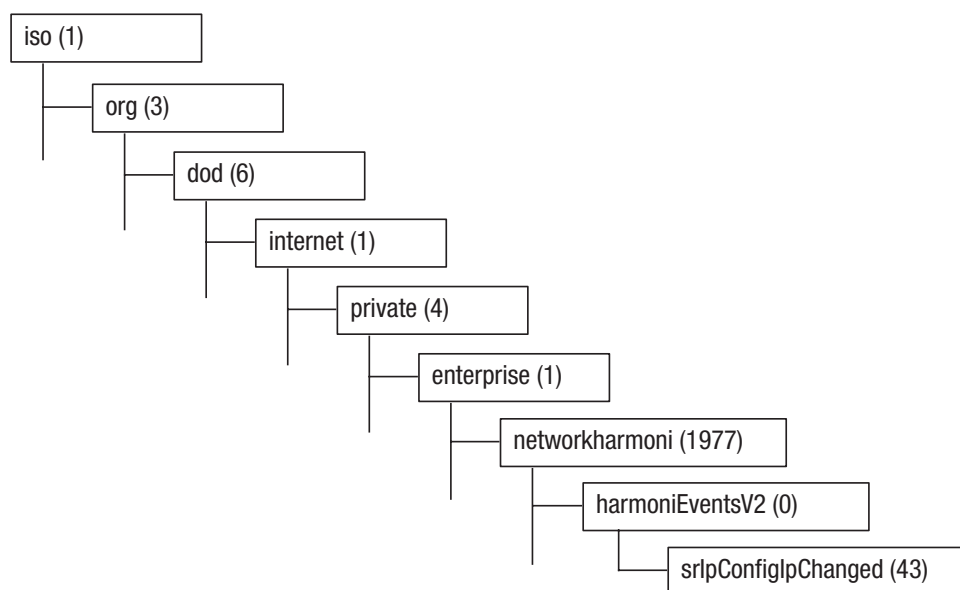


Figure 30. OID tree diagram for ipconfig notification type

The srIpConfigIpChanged notification type contains the following variable binding:

- agentIdUniqueId

---

## Chapter 14. LogMonX

The logmonx subagent and the associated logMonX MIB module monitor log files for specific expressions and generate an event and notification upon finding those expressions.

Using logmonx you can monitor multiple log files whose entries span multiple lines. You can also use the subagent to monitor Windows system, application and event logs.

---

### Component files

The logmonx subagent and MIB module are comprised of a set of component files.

Table 82 lists the logmonx subagent and MIB module component files and their installed locations.

*Table 82. logmonx component files*

File	Location	Description
logmonx.dll (Windows) liblogmonx.so/.sl (UNIX)	bin	Binary implementation of the logmonx subagent.
logmonx-mib.mib	mibs	logmonx MIB definition document.
logmonx.oid	config/oid	logmonx subagent object identifier file.

---

### Guidelines

Use the control table to identify the log files that you wish to monitor.

#### Before you begin

Load the subagent using the command:

```
subagent load logmonx
```

#### Procedure

The general procedure for using logmonx is:

1. Create a control row by setting logMonXControlStatus to CreateAndWait(5).
2. Specify the log files to be monitored (logMonXControlLogFile).
3. Specify a filter expression (logMonXControlFilter) that identifies the log file message that you wish to monitor.  
The default expression is \*. See "Filter expressions" on page 143 for more details.
4. Set the log file entry start and end delimiters (logMonXControlBeginDelimiter and logMonXControlEndDelimiter) to the characters sequences that indicate the start and end of a log file entry.  
See "Log file entry delimiters" on page 143 for more details about delimiters.
5. Set logMonXControlStatus to active(1).  
The agent commences monitoring the log file.

## Specifying the monitored log file

The `logMonXControlLogFile` object specifies the log files to be monitored. This object supports a number of syntax conventions for specifying the target log file.

### Using environment variables

`logMonXControlLogFile` supports environment variable expansion in filenames and paths.

#### About this task

The format for specifying environment variables is `${envvar}`, where `envvar` is the name of the required environment variable. For example, to monitor the Netcool/SLAM server log file, you might use the environment variable `$SLAM_HOME` by setting `logMonXControlLogFile` to `${SLAM_HOME}/log/SLAM_server.log`.

### Monitoring one log file

To monitor a single log file, set the value of `logMonXControlLogFile` to the full path of a log file.

#### Example

For example, on UNIX systems the expression `/var/log/messages` indicates the messages log file located in the `/var/log/` directory.

### Monitoring multiple log files in one directory

To monitor multiple log files in a single directory, set the value of `logMonXControlLogFile` to a glob pattern representing all the files in a specific directory.

#### Example

For example, on UNIX systems the expression `/var/log/*` indicates all log files in the directory `/var/log/`.

### Monitoring multiple log files in multiple directories

To monitor multiple log files located in different directories, set the value of `logMonXControlLogFile` to a set of glob patterns and filenames, each separated by a pipe character (`-`).

#### About this task

The glob expression must not contain any whitespace characters adjacent to pipe characters. For example, on UNIX systems the expression `/var/log*-/etc/newlogfile.txt` indicates all files in the directory `/var/log/` as well as the file `/etc/newlogfile.txt`.

The range of glob patterns available depends on the operating system of the machine on which the log files are located. Common glob characters include the asterisk (`*`) and question mark (`?`) characters. On UNIX systems a wide range of patterns is available, see the operating system's man pages for more details.



## Monitoring Windows event log files

To monitor a Windows event log file, set the value of `logMonXControlLogFile` to *\*logname*, where *logname* denotes a Windows event log name such as SYSTEM, SECURITY or APPLICATION.

## Handling rotated log files

If the monitored log file is rotated to another file when it reaches a certain size, you can configure the subagent so that monitoring is not interrupted when the log is rotated.

### About this task

To configure the subagent to handle rotated logs, set the `logMonXControlRolledFile` object to the absolute path of the file to which the monitored log file is rotated.

If you are monitoring multiple log files using a single control row, you can specify multiple rotated logs using the prefix `!`. For example, the value `!.0` specifies that the names of the rotated logs are the same as that of the monitored logs except that they have the extension `.0`.

## Log file entry delimiters

The `logMonXControlBeginDelimiter` and `logMonXControlEndDelimiter` enable you to define delimiters for the beginning and end of log file entries that span multiple lines.

Table 83 lists basic log file entry delimiters.

*Table 83. Combinations for logMonXControlBeginDelimiter and logMonXControlEndDelimiter*

BeginDelimiter	EndDelimiter	Purpose
<code>^.+</code>	<code>\r\n\$</code>	Specifies a single line on Windows platforms.
<code>^.+</code>	<code>\n\$</code>	Specifies a single line on UNIX platforms.

`logmonx` interprets the `logMonXControlBeginDelimiter` and `logMonXControlEndDelimiter` objects as regular expressions. For more information about using regular expressions, see Appendix D, “Regular expressions,” on page 615.

**Note:** You do not need to specify delimiters when monitoring Windows system event files.

## Filter expressions

`logmonx` uses the regular expression specified in the object `logMonXControlFilter` to perform substring matches on entries in the log files being monitored.

For detailed information about using regular expressions, see Appendix D, “Regular expressions,” on page 615.

## Reverse filters

The `logMonXControlReverseFilter` object defines an additional match criteria that operates in combination with `logMonXControlFilter`.

This additional criteria acts in the following ways:

- When the value of `logMonXControlMode` is `include(1)`, it acts as an exclusion qualifier for the filter expression. For a filter match to occur in this mode, any log file entry matching `logMonXControlFilter` must not match the expression defined in `logMonXControlReverseFilter`.
- When the value of `logMonXControlMode` is `exclude(2)`, it acts as an inclusion qualifier for the filter expression. defined in `logMonXControlFilter`. For a filter match to occur in this mode, any log file entry not matching `logMonXControlFilter` must match the expression defined in `logMonXControlReverseFilter`.

If `logMonXControlReverseFilter` does not contain an expression, it has no effect on filter expression matching and is *not* interpreted as an empty regular expression.

## Matching multi-byte characters

When defining regular expressions to match multi-byte characters, enclose each multi-byte character in parentheses `()`.

## Log file entry size limitations

When defining the log file entry delimiters, ensure that they resolve to log file entries that are as small as possible.

The size of log files entries is important for a number of reasons:

- When `logmonx` detects a match in a log file entry, it stores the entire entry in the `logMonXHistoryLine` object. It is important to keep this in mind when you specify the log file entry delimiters because they directly affect the size of log file entries and the amount of data stored in each history table row.

For example, if the delimiters resolve to log file entries that are 1000 bytes in length, each instance of `logMonXHistoryLine` will contain 1000 bytes. A large number of log file matches would quickly degrade the agent's performance.

- `logmonx` buffers a maximum of 32 lines of information for each log file entry. If the delimiters resolve to entries that span more than 32 lines, `logmonx` will store only the last 32 lines of any log file match in `logMonXHistoryLine`. Any information in previous lines of a log file entry will be lost.

For example, if the delimiters result in log file entries that span 34 lines, the first 2 lines of any matching entry will be lost when the entry is stored in the history table, possibly resulting in the loss of important log file information.

- The maximum size of the `logMonXHistoryLine` object is 4096 bytes, which, depending on the agent's configuration, may exceed the maximum length allowed for trap PDUs. One of the variables bound to the `logMonXFileMatch` trap is the `logMonXHistoryLine` object. If a log file entry results in a trap PDU size greater than that permitted by the agent, the trap is not sent.

**Note:** While it is possible to avoid trap PDU limitations by increasing the size of the trap PDU, doing so may have serious side effects on the agent's operation and cause compatibility problems within your management environment.

## Storing state information

Normally, logmonx starts monitoring a log file from the end of the file. However, if the agent becomes unavailable for some reason, such as agent reboot, this can lead to new entries in log files going undetected if they were added to the log file while the agent was unavailable.

To eliminate this possibility, you can configure logmonx to store information about its internal state in a separate file each time it shuts down. By storing its state information, the logmonx subagent is able to continue monitoring log files from the point which it was at prior to shutdown.

To enable this function, add the following inivar definition to the Netcool/SSM `init.cfg` file:

```
LogMonXStateFile=filename
```

where `filename` denotes the file in which logmonx stores its internal state information.

**Attention:** The subagent overwrites the file specified by the `logMonXStateFile` inivar. Never set the value of this inivar to the name of any Netcool/SSM core configuration file.

**Note:** Control rows that are created after the agent has booted always commence log file monitoring from the end of a file. The `LogMonXStateFile` inivar does not affect this behavior.

### Specifying a start position

You can specify a position in the log file from which the subagent will commence monitoring.

### About this task

This enables you to either skip over old entries when configuring a monitor for the first time, or to revisit older entries that may have occurred if the subagent was inactive for some reason and a state file was not in use.

To specify a start position for monitoring, use the `logMonXStatsCurrentPosition` object.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The logmonx configuration commands create rows in the control table (logMonXControlTable).

The general syntax is these commands is:

```
logmonx property=value
logmonx create property=value ...
logmonx reset
```

Table 84 lists the properties supported in these commands.

Table 84. Configuration command parameters - logmonx

Property	Type	Description	Sets MIB object
begindelimiter	string	The delimiter for the start of a multi-line log file entry.	BeginDelimiter
brackets	string	<p>Defines optional brackets which are inserted into logMonXHistoryLine around the matching substring. If brackets is set, the first matching substring of each line is bracketed. This is two strings separated by a comma, where the first is the opening bracket and the second is the closing bracket:</p> <pre>( ) { {,} } &gt;&gt;&gt;,&lt;&lt;&lt; BEGIN-&gt;,&lt;-END</pre> <p>For example:</p> <pre>filter: "the only" brackets: "{ { { {,} } } }" log file: "this will be the only line in this log file" logmon reports: "this will be { { { {the only} } } } line in this log file"</pre> <p>The default is blank and no bracketing is applied.</p>	Brackets
datacontrol	enum	<p>Data control:</p> <pre>on - Enables data collection. off- Suspends data collection</pre>	DataControl
defaultlevel	enum	<p>The default severity level assigned to each matching entry:</p> <pre>error information unknown warning</pre>	DefaultLevel
description	string	A description of the control row.	Description
displaylimit	int	The maximum length of stored log file entries.	DisplayLimit
enddelimiter	string	The delimiter for the end of a multi-line log file entry.	EndDelimiter
event	int	The index of the event generated when a matching log file entry is found.	Event
eventstatus	enum	<p>Event flow control:</p> <pre>alwaysready fired ready</pre>	EventStatus

Table 84. Configuration command parameters - logmonx (continued)

Property	Type	Description	Sets MIB object
filter	string	A regular expression specifying the text to be searched for in the log file.	Filter
logfile	string	The filename of the log file to be monitored.	LogFile
mode	enum	Sets the event generation behavior: exclude include	Mode
owner	string	The owner of the control row.	Owner
reversefilter	string	A regular expression specifying a qualifier for the filter expression.	ReverseFilter
rolledfile	string	The absolute path to the rolled file of the log file being monitored.	RolledFile
updateinterval	int	The interval (in seconds) at which the log file is monitored.	UpdateInterval
window size	int	The maximum number of rows maintained in the history table for this control row.	WindowSize

## Examples

These examples demonstrate how to use the logmonx subagent perform simple log file monitoring tasks.

### Monitoring system warnings

Monitor all log files located in the directory /var/log/ on a UNIX system, using the word warning as a filter expression. Store the 60 most recent matches.

The subagent configuration commands for creating this log file monitor are:

```
subagent load logmonx

logmonx reset
logmonx logfile=/var/log/*
logmonx filter=warning
logmonx begindelimiter=^.+
logmonx enddelimiter=\n$
logmonx window size=60
logmonx create
```

### Finding specific error messages

Monitor the log file /opt/oracle/OraHome1/network/log/sqlnet.log at one-minute intervals, for entries of the format:

```
*****
...
nt OS err code:
```

For example, the log file monitor would match the following entry:

```
*****
Fatal NI connect error 12547, connecting to:
(LOCAL=NO)
```

```
VERSION INFORMATION:
TNS for Solaris: Version 9.0.1.0.0 - Production
```

```

Oracle Bequeath NT Protocol Adapter for Solaris: Version 9.0.1.0.0 - Product
TCP/IP NT Protocol Adapter for Solaris: Version 9.0.1.0.0 - Production
Time: 18-JUN-2002 10:44:58
Tracing not turned on.
Tns error struct:
  nr err code: 0
  ns main err code: 12547
  TNS-12547: TNS:lost contact
  ns secondary err code: 0
  nt main err code: 0
  nt secondary err code: 0
  nt OS err code: 0

```

The subagent configuration commands for creating this log file monitor are:

```

subagent load rmonc
event reset
event type=snmp-trap
event description="Change in log /opt/oracle/OraHome1/network/log/sqlnet.log"
event create
lmevent=$?

```

```

subagent load logmonx
logmonx reset
logmonx logfile=/opt/oracle/OraHome1/network/log/sqlnet.log
logmonx filter=.*
logmonx begindelimiter="\{8,}"
logmonx enddelimiter="nt OS err code:"
logmonx windowsize=60
logmonx updateinterval=60
logmonx event=$lmevent
logmonx eventstatus=eventReady
logmonx defaultlevel=information
logmonx create

```

## Monitoring Windows log files

Monitor a Windows system log file for any changes:

```

subagent load logmonx
logmonx reset
logmonx logfile=*SYSTEM
logmonx filter=.*
logmonx windowsize=60
logmonx create

```

## Monitoring Web proxy usage

Detect unapproved use of a Web proxy by monitoring the access log file for IP addresses that are not permitted to use the Web proxy:

```

subagent load logmonx
logmonx reset
logmonx logfile=var/log/squid/access.log
logmonx filter=217\.104\.227\.*-195\.175\.0\.*
logmonx begindelimiter=^.+
logmonx enddelimiter=\n$
logmonx mode=exclude
logmonx windowsize=60
logmonx create

```

## Excluding phrases

Monitor the log file /var/adm and generate an event if any entry in the file contains the string failure but not automount:

```

subagent load rmonc
event reset
event type=snmp-trap
event description="Failure detected in log file /var/adm"
event create
adm_log_event=$?

subagent load logmonx
logmonx reset
logmonx logfile=/var/adm
logmonx begindelimiter=^.+
logmonx filter="failure"
logmonx reversefilter="automount"
logmonx enddelimiter=\n$
logmonx mode=include
logmonx windowsize=20
logmonx event=$adm_log_event
logmonx eventstatus=eventReady
logmonx defaultlevel=error
logmonx create

```

## MIB module

The logMonX MIB is a subtree of networkharmoni (1977).

The logMonX subtree is shown in Figure 31.

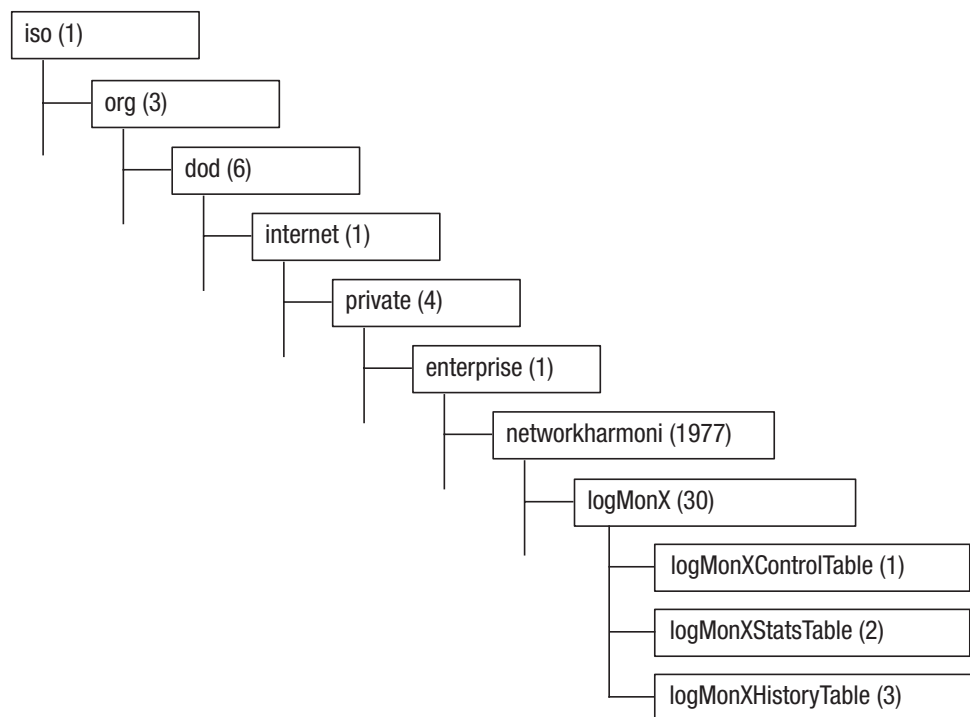


Figure 31. OID tree diagram of the logMonX MIB module

This section provides a summary of the objects defined in the logMonX MIB module. For detailed information on all objects in the module, see the logmonx-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB objects

The logMonX MIB module provides a standard SNMP interface.

The MIB module contains:

- A control table for defining log file monitors.
- A statistics table, which summarizes the results of log file monitoring.
- A history table, which stores log file entries detected during log file monitoring.

### Control table

The control table (logMonXControlTable) provides control rows for setting up log file monitoring.

Each control row defines a monitor for one or more log files, which are checked at regular intervals for a specific filter expression. If the expression is found in a log file, the log file entry that contains the expression is written to the history table. The control table enables you to monitor multiple log files with just one control row. Table 85 describes the objects provided in logMonXControlTable.

Table 85. Control table (logMonXControlTable)

Row object	Description
BeginDelimiter	A regular expression defining a delimiter for the beginning of a multi-line log file entry. For details about regular expressions, see Appendix D, "Regular expressions," on page 615.  Default value: ^.+.
CreateTime	The value of sysUpTime when the control row was last activated.
DataControl	Controls the monitoring status of the control row: on(1) - Monitoring is enabled off(2) - Monitoring is disabled
DefaultLevel	Specifies a default severity level that is assigned to each history row created for this control row. This value is only assigned if the agent is unable to determine a severity level from the log file entry itself and would otherwise assign it the value unknown(1).  Default value: unknown(1).
Description	A description of the control row (that is, the purpose of the log file monitor).
DisplayLimit	The maximum length of entries in the logMonXHistoryLine object.  Default value: 512.
EndDelimiter	A regular expression defining a delimiter for the end of a multi-line log file entry. For details about regular expressions, see Appendix D, "Regular expressions," on page 615.  Default value: \r?\n\$.
Event	The event fired if an entry matching the logMonXControlFilter expression is found in the monitored log files. Set this variable to a valid RMON event index. A value of 0 indicates that no event is fired.
EventStatus	RMON-standard event control. See Appendix B, "RMON1 MIB group," on page 533 for details.
EventTime	The value of sysUpTime when the event was last fired.



Table 85. Control table (logMonXControlTable) (continued)

Row object	Description
Filter	<p>A regular expression (also called the filter expression or match expression) specifying the information to be searched for in the log files being monitored. Whenever the subagent finds a log file entry matching this expression, it writes that log file entry to the history table, logMonXHistoryTable. To match all entries in a log file, use the expression <code>.*</code>. For details about regular expressions, Appendix D, “Regular expressions,” on page 615.</p> <p>Default value: <code>.*</code>.</p>
LogFile	<p>Contains the path or path expression of the log files to be monitored. For more instructions on specifying a log filename, see “Specifying the monitored log file” on page 142.</p>
Mode	<p>Sets the event generation behavior:</p> <p>include(1) - An event is generated if the new log file entry matches the logMonXControlFilter expression and does not match the logMonXControlReverseFilter expression.</p> <p>exclude(2) - An event is generated if the new log file entry does not match the logMonXControlFilter expression and matches the logMonXControlReverseFilter expression.</p> <p>Default value: include(1).</p>
Owner	<p>The name of that entity that configured the control row.</p>
ReverseFilter	<p>Defines an additional match criteria. When the value of logMonXControlMode is include(1), this object defines an exclusion qualifier for the expression defined in logMonXControlFilter. When the value of logMonXControlMode is exclude(2), this object defines an inclusion qualifier for the expression defined in logMonXControlFilter.</p> <p>If logMonXControlReverseFilter does not contain an expression, it has no effect on the match criteria.</p>
RolledFile	<p>Specifies the absolute path to the rolled file of the log file being monitored. Monitor this file when logMonXControlLogFile is rolled. If the logMonXControlLogFile specifies multiple files, specify multiple rolled log files by prefixing the name with an exclamation mark (!). For example, <code>!.0</code> specifies that the rolled file name is the same as the log file except that it has a <code>.0</code> extension.</p>
UpdateInterval	<p>Sets the interval (in seconds) at which the monitored log files are checked.</p> <p>Default value: 60.</p>
WindowSize	<p>Sets the maximum number of rows maintained in the history table logMonXHistoryTable for this control row.</p> <p>Default value: 0.</p>

## Statistics table

The statistics table (logMonXStatsTable) contains summaries of the results generated by log file monitors. A single control row can monitor multiple log files, so the statistics table contains a separate row for each log file monitored. Statistics table rows map to control table rows through the logMonXControlIndex object. Table 86 describes the objects provided in logMonXStatsTable.

Table 86. Stats table (logMonXStatsTable)

Row object	Description
AbsPath	The absolute path and filename of the monitored log file.
CurrentPosition	<p>Sets the position in the log file at which the subagent begins monitoring during the next sample. For the value <math>n</math>, the starting position is determined as follows:</p> <p><math>n = 0</math> Seek to the end of the log file.</p> <p><math>n &lt; 0</math> Begin processing from the last <math>n</math> bytes of the log file (or the last <math>n</math> entries when monitoring Windows event logs). For example, a value <math>n</math> of -1024 reads the last 1024 bytes or entries of the log file.</p> <p><math>n &gt; 0</math> Begin processing from the <math>n</math>th byte (or the <math>n</math>th entry for Windows event logs) from the start of the log file.</p> <p><b>Note:</b> If <math>n</math> is greater than the current size of the log file, monitoring begins from the end of the file. If the monitored log file is a Windows event log and <math>n</math> is a large negative value, monitoring begins from the start of the log.</p> <p>At the end of each sample interval, this object contains the current file position.</p>
FileSize	The size of the log file (in bytes) of the log file when the most recent matching log file entry was found.
LastError	The most recent error message reported by the agent while accessing the monitored log file.
LastMatch	The value of sysUpTime when the most recent matching log file entry was found.
LastModification	The time at which the specified file was last modified.
LastUpdate	The value of sysUpTime when the log file was most recently checked by the subagent.
Matches	The number of entries in the monitored log file that match the expression defined in logMonXControlFilter. This value represents the number of matches found since monitoring started.
StatIndex	Used in the history table to map history table rows to the corresponding statistics table row.

## History table

The history table (`logMonXHistoryTable`) stores log file entries that match the filter expressions defined in control rows.

History table rows map to control table rows via the `logMonXControlIndex` variable. Summary information about the history rows generated for each log file is stored in a statistics table row, referenced by the `logMonXStatsStatIndex` object. The number of history rows allocated to a single log file is defined in the corresponding control row's `logMonXControlWindowSize` variable. If the current number of rows stored for a log file reaches this value, the oldest history row is removed each time a new row is created. Table 87 describes the objects provided in `logMonXHistoryTable`.

*Table 87. History table (`logMonXHistoryTable`)*

Row object	Description
Level	The severity level of the log file entry, if it can be determined. Generally, the subagent can only determine severity levels from Windows application and system log files.
Line	The log file entry containing the matching filter expression.
Time	The date and time when the history row was added.

## Notification types

The `logMonX` MIB defines notification types for events generated by the subagent.

These notification types are shown in Figure 32 on page 154 and listed in Table 88 on page 154.

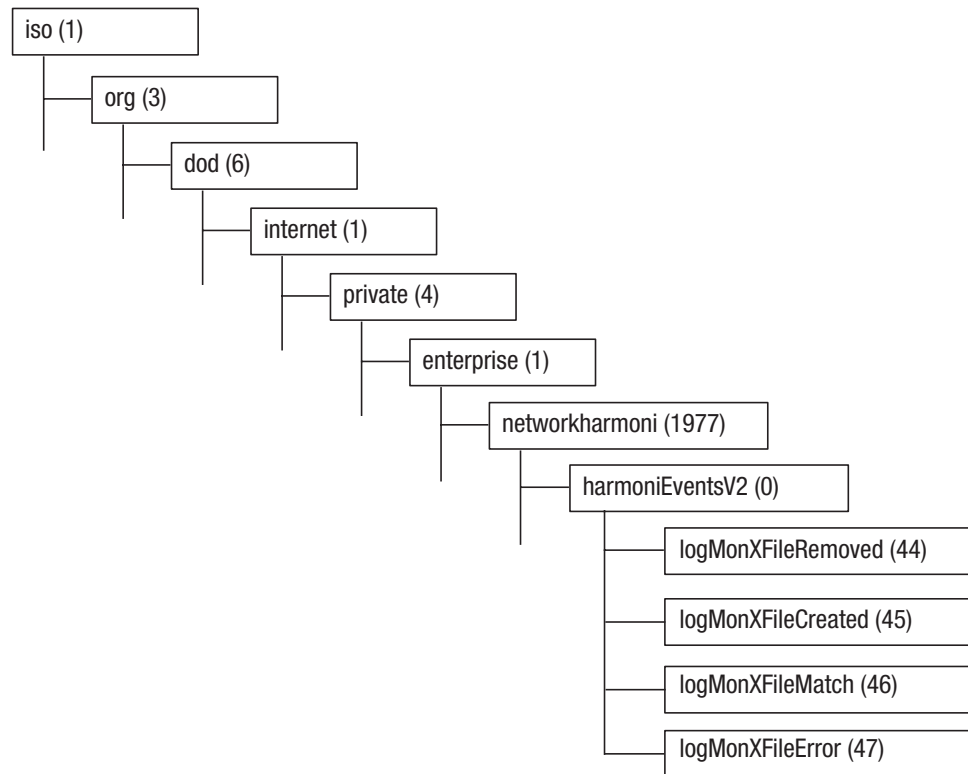


Figure 32. OID tree diagram for logMonX notification types

Table 88. logMonX notification types

Notification types	Description
logMonXFileCreated	Generated when a file being monitored is created.  Variable bindings: logMonXStatsLogFile logMonXControlDescription
logMonXFileError	Generated when an attempt by the subagent to read a monitored log file fails.  Variable bindings: logMonXStatsLastError logMonXControlDescription
logMonXFileMatch	Generated when an entry in a monitored log file matches the expression defined in logMonXControlFilter.  Variable bindings: logMonXControlLogFile logMonXHistoryLine logMonXHistoryLevel logMonXControlDescription

Table 88. *logMonX* notification types (continued)

Notification types	Description
logMonXFileRemoved	<p>Generated when a log file being monitored can no longer be opened.</p> <p>Variable bindings:</p> <p>logMonXStatsLogFile</p> <p>logMonXControlDescription</p>



---

## Chapter 15. NT performance monitor

The ntperfmon subagent views data from performance tables on Windows hosts, which enables it to monitor tasks and processes. These elements include ACS/RSVP Service, browser, cache, distribution transaction, authentication, network, printer, disk, process, RAS, and system metrics.

The ntperfmon subagent and the associated ntPerfMon MIB module access the performance metrics provided by the Windows performance monitoring utility perfmon on machines running Windows platforms.

**Note:** The types of object, counter, and instance that can be monitored depend on the host's operating system, hardware and software.

---

### Overview

The ntperfmon subagent and the ntPerfMon MIB module are comprised of a set of component files.

Table 89 lists the ntperfmon subagent and MIB module component files and their installed locations.

*Table 89. ntperfmon component files*

File	Location	Description
ntperfmon.dll	bin	Binary implementation of the ntperfmon subagent.
ntperfmon-mib.mib	mibs	ntperfmon MIB definition document.
ntperfmon.oid	config/oid	ntperfmon subagent object identifier file.

---

### Guidelines

ntperfmon provides access to metrics available in the Windows perfmon utility.

To load the subagent, use the command:

```
subagent load ntperfmon
```

### Monitoring performance metrics

Use the control and counter tables to specify the Windows perfmon performance objects, counters and instances that you wish to monitor.

#### Procedure

To monitor performance metrics:

1. Open the Windows perfmon utility and identify the names of the performance object, the counters and the instances that you wish to monitor.
2. Create a control row in ntPerfMonControlTable by setting the ntPerfMonControlStatus object to createAndWait(5).
3. Set ntPerfMonControlObject to the name of the Windows perfmon performance object that you wish to monitor.

**Note:** When entering the names of Windows perfmon objects, counters and instances, use exactly the same name as that shown in the Windows perfmon utility.

4. Identify the counters that you wish to monitor:
  - a. To monitor all the counters in the performance group, set `ntPerfMonControlCounterMode` to `automatic(2)`.  
The subagent automatically creates a row in the counter table for every counter available in the performance object.
  - b. To monitor a selection of counters in the performance group, set `ntPerfMonControlCounterMode` to `manual(1)`, set `ntPerfMonControlCounters` to the number of counters that you wish to monitor. Create a row in the counter table for each counter that you wish to monitor, and set `ntPerfMonCounterName` to the name of the counter as displayed in the Windows perfmon utility.

**Tip:** You can apply scaling to the collected counter data using the `ntPerfMonCounterScale` object.

5. If the counters have more than one instance, you can narrow the selection using the `ntPerfMonControlInstance` and `ntPerfMonControlMode` objects to include or exclude particular instances of the counter.
6. Set the `ntPerfMonControlStatus` object to `active(1)`.

The selected metrics are listed in the data table `ntPerfMonDataTable`.

## Identifying performance metrics in Windows perfmon

In the Windows perfmon utility, performance metrics are grouped by performance objects. Each performance object contains a number of counters, each of which holds a performance metric.

If a performance object pertains to multiple devices or entities on the host machine, each of its counters has more than one instance, one for each device or entity. Figure 33 shows how to identify performance objects, counters, and instances in the perfmon Add Counters dialog.

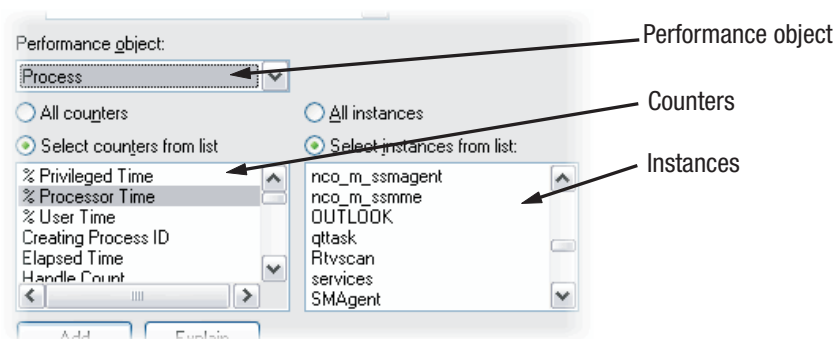


Figure 33. Identifying perfmon performance objects, counters and instances

For example, on a host machine with more than one physical disk, the counters that provide metrics for physical disk performance have multiple instances, one for each disk.



## Data table indexing

Rows in the data table are indexed by a combination of ntPerfMonControlInstance, ntPerfMonCounterIndex, and ntPerfMonDataInstance objects. However, ntPerfMonDataInstance objects contain Windows perfmon instance names, which are string values, which cannot be used as indexes. You must convert such strings to sub-OIDs.

To create a sub-OID from the string in an ntPerfMonDataInstance object, use a length-encoded ASCII string; that is, insert the length of the string, then convert each character in the string into its decimal ASCII value. For example to refer to an ntPerfMonDataValue object in a data table row whose ntPerfMonDataInstance object contains the string explorer, use the following OID:

```
$ntPerfMonDataValue.controlIndex.counterIndex.8.101.120.112.108.111.114.101.114
```

## SNMPv1 compatibility

The ntPerfMon MIB module uses the Counter64 data type, so it is not compatible with SNMPv1. Only use the ntperfmon subagent with SNMPv2 and later.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The perfmon commands create rows in the control table (ntPerfMonControlTable).

The general syntax of these commands is:

```
perfmon property=value  
perfmon create property=value ...  
perfmon reset
```

Table 90 lists the properties supported in these commands.

*Table 90. Configuration command parameters - perfmon*

Property	Type	Description	Sets MIB object
ctrmode	int	Specifies how the counter table is populated:  automatic  manual	CounterMode
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl
inst	enum	Specifies the instance to be monitored.	Instance

Table 90. Configuration command parameters - perfmon (continued)

Property	Type	Description	Sets MIB object
instmode	enum	Sets the function of the inst property:  exclude  include	InstanceMode
interval	int	Sets the sampling interval (in seconds).	SampleInterval
object	string	Specifies the perfmon counter to be monitored.	Object

## Counter table

The perfmonctr commands create rows in the counter table (ntPerfMonCounterTable) and assign them to the next ntPerfMon control row created using the perfmon create command.

The general syntax of these commands is:

```
perfmonctr property=value
perfmonctr store property=value ...
perfmonctr reset
```

Table 91 lists the properties supported in these commands.

Table 91. Configuration command parameters - perfmonctr

Property	Type	Description	Sets MIB object
name	string	The name of the monitored counter.	Name
scale	int	The scaling factor applied to the counter value:  $n < -1$ : scale using $\text{counterval} * n$  $-1 \leq n \leq 1$ : no scale  $n > 1$ : scale using $\text{counterval} / n$	Scale

## Examples

These examples demonstrate how to use the ntperfmon subagent to perform simple monitoring tasks.

### Monitor all processes

Monitor all the processes running on the host machine:

```
subagent load ntperfmon
perfmon reset
perfmon object=Process
perfmon ctrmode=automatic
perfmon create
```

### Monitor physical disks

Use the ntperfmon subagent to monitor the physical disks on a host machine running Windows, gathering information about the average time taken for disk reads and writes as well as the data transfer rate and the number of data transfers

per second from the four Windows perfmon counters Avg. Disk sec/Read, Avg. Disk sec/Write, Disk Bytes/sec, and Disk Transfers/sec:

```
subagent load ntpfmon
perfmon reset
perfmon object=PhysicalDisk
perfmon inst=_Total
perfmon instmode=exclude
perfmon ctrmode>manual
perfmonctr reset
perfmonctr store name="Avg. Disk sec/Read"
perfmonctr store name="Avg. Disk sec/Write"
perfmonctr store name="Disk Bytes/sec"
perfmonctr store name="Disk Transfers/sec"
perfmon create
```

**Tip:** The control row can either include or exclude counter instances. More than one instance of a counter is available when more than one device exists to which the counter applies. In such cases, the Total instance provides a total for all devices. When monitoring physical disks, it is likely that there will at least be a hard drive and a floppy drive. This example monitors the statistics for the individual disks, so the Total instance is not required and is excluded.

## Monitor an application

Monitor the Windows perfmon counter named Private Bytes for the process explorer.exe running on the local machine. Using the genalarm subagent, generate an event when the value of the counter exceeds an upper threshold value of 2MB:

```
uThreshold=2000000
lThreshold=1250000

subagent load ntpfmon
subagent load genalarm

perfmon reset
perfmon object="process"
perfmon inst=explorer
perfmon instmode=include
perfmon ctrmode>manual
perfmon interval=60
perfmonctr reset
perfmonctr store name="Private Bytes"
perfmon create
ctrlIdx=?

event type=snmp-trap
event community=public
event description="explorer Private Bytes alarm"
event create
violationevent=?

event description="explorer Private Bytes normal"
event create
normalevent=?

genalarm reset
genalarm var="$ntPerfMonDataValue.$ctrlIdx.1.8.101.120.112.108.111.114.101.114"
genalarm vardescr="explorer Private Bytes"
genalarm interval=60
genalarm type=absolute
genalarm mode=hysteresis
genalarm risethresh=$uThreshold
genalarm fallthresh=$lThreshold
genalarm riseduration=0
```

```

genalarm fallduration=0
genalarm riseevent=$violationevent
genalarm fallevent=$normalevent
genalarm risedescr="explorer Private Bytes usage at $$6."
genalarm falldescr="explorer Private Bytes usage returned to acceptable (l.t. $$7)."
genalarm create

```

## MIB module

The ntPerfMon MIB is a subtree of networkharmoni (1977).

The ntPerfMon subtree is shown in Figure 34.

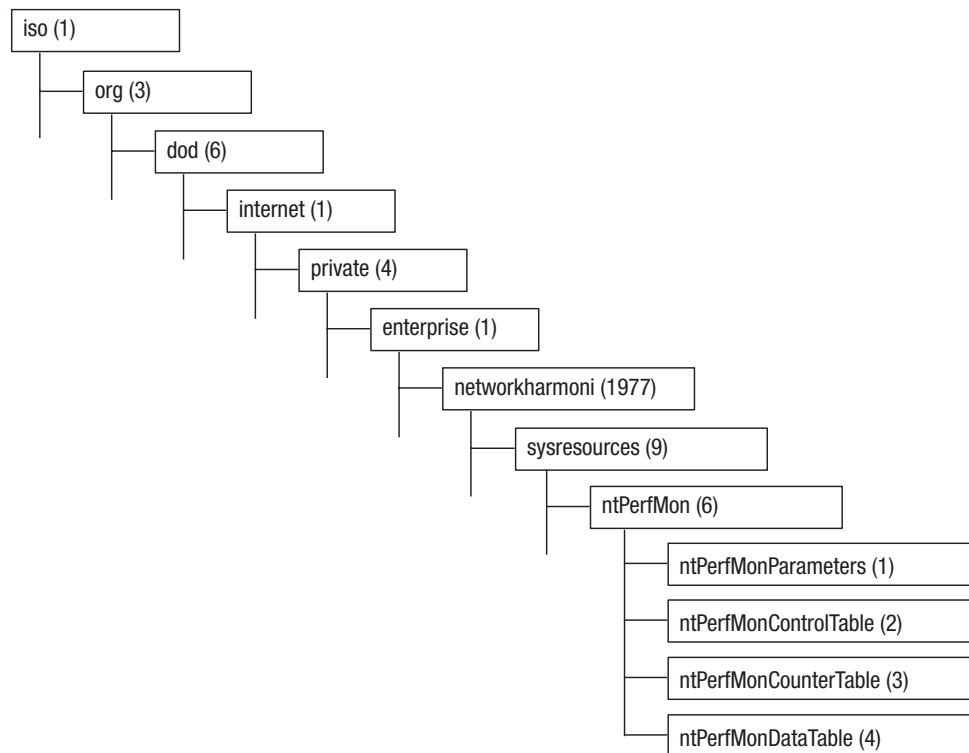


Figure 34. OID tree diagram of the NT performance monitor MIB module

This section provides a summary of the objects defined in the ntPerfMon MIB module. For detailed information on all objects in the module, see the ntpfmon-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB objects

The ntPerfMon MIB provides performance data from host machines running Windows operating systems.

The MIB module contains:

- A performance parameter group, ntPerfMonParameters
- A control table, ntPerfMonControlTable
- A counter table, ntPerfMonCounterTable
- A data table, ntPerfMonDataTable

## ntperfmon parameter group

The performance parameter group (ntPerfMonParameters) contains the object ntPerfMonThreadLimit, which sets the number of threads used for data collection. A value of zero allocates a thread limit of 1 for each control row.

**Note:** Once ntPerfMonThreadLimit is set, you can only decrease its value by stopping the agent and restarting it. The exception to this is to set the value to zero, which allocates a thread limit of 1 for each control row.

## Control table

The control table (ntPerfMonControlTable) specifies perfmon counters to be monitored.

Each control row specifies one or more of the Windows perfmon objects to be monitored. Table 92 lists the row objects in ntPerfMonControlTable.

Table 92. ntPerfMon control table (ntPerfMonControlTable)

Row object	Description
CounterMode	Determines how ntPerfMonCounterTable is populated with counter names for the object specified by ntPerfMonControlObject:  manual(1) - The table must be manually populated  automatic(2) - Counter names are automatically added to the table
Counters	The number of rows in ntPerfMonCounterTable allocated to this control row.
CreateTime	The time at which this control row was activated.
DataControl	Controls data collection:  on(1) - Enables data collection  off(2) - Disables data collection, however any data already in the data table is preserved
Instance	Specifies the instance name of the performance monitoring object from which data is sampled. The interpretation of this name is determined by the ntPerfMonControlInstanceMode object.
InstanceMode	Determines how the name specified by ntPerfMonControlInstance is interpreted. The interpretations are:  include(1) - The name of the instances from which data is sampled. Use the name * to indicate all instances.  exclude(2) - The name of the instances from which data is not sampled  includeRegex(3) - A regular expression for the names of instances from which data is sampled  excludeRegex(4) - A regular expression for the instances from which data is not sampled
Object	Specifies the performance monitoring object from which data for the counters in the ntPerfMonCounterTable is sampled.
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
SampleInterval	The interval (in seconds) at which data for the rows in ntPerfMonCounterTable are sampled.

Table 92. *ntPerfMon control table (ntPerfMonControlTable)* (continued)

Row object	Description
Status	The status of the control row. Setting the status to active(1) commences sampling of the associated counters. When the status of this object changes from active(1) to any other value, all data table rows associated with the control row are cleared.

## Counter table

The counter table (ntPerfMonCounterTable) lists the monitored perfmon counters and their status.

Each row in the table represents one counter. Table 93 lists the row objects in ntPerfMonCounterTable.

Table 93. *ntPerfMon counter table (ntPerfMonCounterTable)*

Row object	Description
Index	Uniquely identifies this row.
Name	The name of the monitored counter. The counter must be a counter belonging to the object specified by the corresponding control row's ntPerfMonControlObject.
Scale	The current scaling factor that is applied to this counter. The value of the counter is divided by the value of this object to produce a scaled value.  Values 1 and -1 apply no scaling.  Negative values apply a multiplication factor to the 32-bit value.  Default value: 1
Status	This object indicates whether or not the corresponding ntPerfMonCounterName has been successfully queried:  pending(1)  invalid(2) - The value of any corresponding entry in the ntPerfMonDataTable will be set to zero  valid(3) - The counter is valid and all corresponding entries in ntPerfMonDataTable contain valid data  negative(4)

## Data table

The data table (ntPerfMonDataTable) stores the data sampled from counter instances specified by control table rows.

Rows in the data table are indexed by a combination of the ntPerfMonControlIndex and ntPerfMonCounterIndex objects, which associate objects with counters. Table 94 lists the objects in ntPerfMonDataTable.

Table 94. *ntPerfMon data table (ntPerfMonDataTable)*

Row object	Description
Instance	Uniquely identifies each row in the data table. The value contained by this object is derived from the Windows perfmon instance.
Value	The current value of this counter. If the actual counter value is greater than 32 bits, then this object will be the value of the lowest 32 bits.

Table 94. *ntPerfMon* data table (*ntPerfMonDataTable*) (continued)

Row object	Description
Value64	The 64-bit representation of the value of this counter.





---

## Chapter 16. NT services

The ntscm subagent and the associated ntServices MIB module provide functions for monitoring and controlling Windows services, similar to those provided by the administrative tools delivered on Windows platforms.

The subagent enables you to start or stop a service or driver in response to changes in its activity.

---

### Component files

The ntscm subagent and MIB module are comprised of a set of component files.

Table 95 lists the ntscm subagent and MIB module component files and their installed locations.

*Table 95. NTServices component files*

File	Location	Description
ntscm.dll	bin	Binary implementation of the ntscm subagent.
ntscm-mib.mib	mibs	ntscm MIB definition document.
ntscm.oid	config/oid	ntscm subagent object identifier file.

---

### Guidelines

Use the subagent to filter NT services and devices of interest, and monitor or control those items.

To load the subagent, use the command:

```
subagent load ntscm
```

- Use the ntServiceFilter group to specify the type of items you wish to gather data on: either drivers, services or both.
- Use the ntServiceTable to monitor the status of drivers and services on the host machine.
- Use the ntServiceControlTable to detect when one of these items stops and, if required, to restart it.

### Inivars

The ntscm subagent provides one inivar for configuring its operation.

The inivar is listed in Table 96.

*Table 96. NT services subagent inivars*

Inivar	Type	Description
ntscmEnum	enum	Enables automatic control row creation for services:  true  false

For general information about inivars, see the *Netcool/SSM Administration Guide*.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### Control table

The ntscm commands create rows in the control table (ntServiceControlTable).

The general syntax of these commands is:

```
ntscm property=value
ntscm create [property=value ...]
ntscm reset
```

Table 97 lists the properties supported in these commands.

*Table 97. Configuration command parameters - ntscm*

Property	Type	Description	Sets MIB object
lag	unsigned	Sets the interval (in seconds) between detection that the monitored service has stopped and any attempt to restart it.	RestartLag
limit	unsigned	Sets the maximum number of attempts to restart the monitored service.	RestartLimit
name	string	The name of the service to monitor.	Name
restart	enum	Determines whether the monitored service is restarted if it stops:  false  true	RestartFlag
sample	int	The sample interval (in seconds).	SampleInterval

---

## Examples

This example demonstrates how to use the ntscm subagent.

### Monitoring a DHCP service

Monitor the DHCP service on the host machine and ensure that if it stops, it will be restarted after 5 seconds. If the service cannot be restarted after 10 attempts then abandon the service.

The configuration commands required to perform this operation are:

```
ntscm reset
ntscm name=Dhcp
ntscm sample=60
```

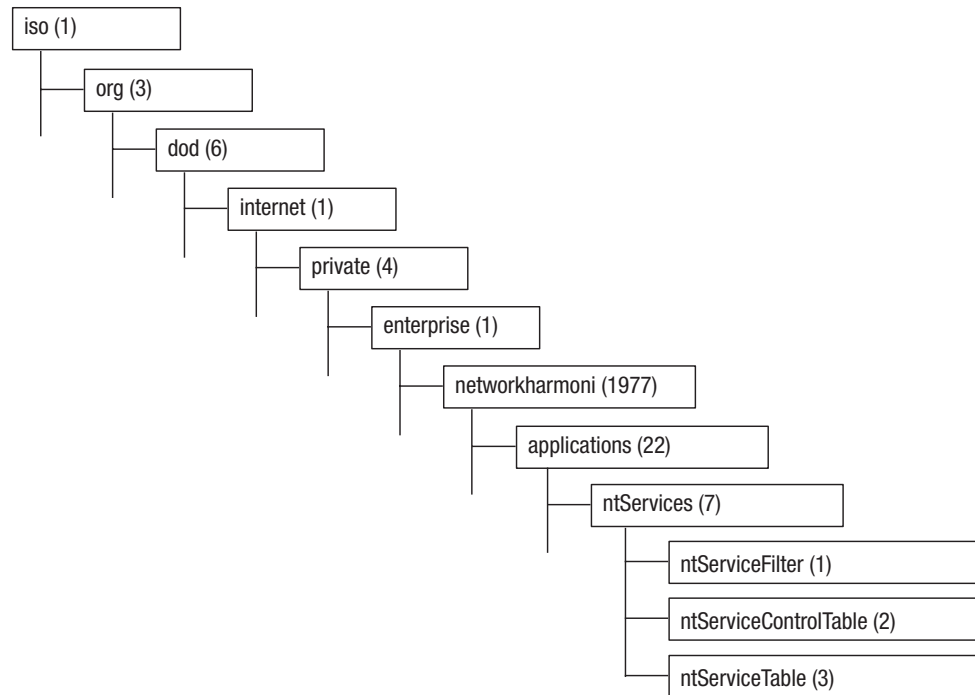
```
ntscm restart=true
ntscm lag=5
ntscm limit=10
ntscm create
```

---

## MIB module

The ntServices MIB is a subtree of networkharmoni (1977).

The ntServices subtree is shown in Figure 35.



*Figure 35. OID tree diagram of the ntServices MIB module*

This section provides a summary of the objects defined in the ntServices MIB module. For detailed information on all objects in the module, see the ntscm-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB objects

The ntServices MIB defines a set of objects useful in detecting the services and drivers running on Windows machines, and for monitoring and controlling the execution of those items.

## Filter group

The NT services filter group (ntServiceFilter) provides options for filtering the information gathered about services and drivers located on the host machine.

Table 98 describes these filters.

*Table 98. NT services filter group (ntServiceFilter)*

Filter	Description
ShowDrivers	Enables the storage of kernel and file system drivers in ntServiceTable:  yes(1) - Enables storage of these items  no(2) - Disables storage
ShowServices	Enables the storage of services in ntServiceTable:  yes(1) - Enables storage of these items  no(2) - Disables storage

## Control table

The NT services control table (ntServiceControlTable) contains configuration information for specifying the services to be monitored and the actions to be taken if a service stops.

Table 99 describes the objects provided.

*Table 99. NT services control table (ntServiceControlTable)*

Row object	Description
CreateTime	The value of sysUpTime.0 when this control table entry was last activated. This object is read-only.
Description	Stores a description of the service being monitored. This object is read-only.
DisplayName	Contains the display (human readable) name of the service. This object is read-only.
Name	Holds the name of the service to be monitored.
RestartFlag	Indicates whether action should be taken when the monitored service stops.
RestartLag	Sets the interval (in seconds) between detection that the monitored service has stopped and when this entry should attempt to restart the service.
RestartLimit	Sets the number of times this entry should restart the monitored service before abandoning further attempts to restart it.
Restarts	Indicates the number of times the monitored service has been restarted by this control entry.
RunControl	Controls the activation of the service: start(1) - Activates the service stop(2) - Deactivates the service

Table 99. NT services control table (ntServiceControlTable) (continued)

Row object	Description
RunStatus	<p>Stores the running status of the service being monitored:</p> <ul style="list-style-type: none"> <li>unknown(1)</li> <li>disabled(2)</li> <li>starting(3)</li> <li>running(4)</li> <li>pausing(5)</li> <li>paused(6)</li> <li>continuing(7)</li> <li>stopping(8)</li> <li>stopped(9)</li> </ul> <p>This object is read-only.</p>
SampleInterval	Sets the interval (in seconds) at which the status of the monitored service is checked. If the data in any of the read-only fields in this control row change, they will be updated at this interval.
StartupType	<p>Stores the monitored service's startup type:</p> <ul style="list-style-type: none"> <li>unknown(1)</li> <li>boot(2)</li> <li>system(3)</li> <li>automatic(4)</li> <li>manual(5)</li> <li>disabled(6)</li> <li>automaticDelayed(7)</li> </ul> <p>This object is read-only.</p>
User	The account under which the monitored service is run. This object is read-only.

## Service table

The NT services table (ntServiceTable) lists status information about the services and drivers located on the host machine. It also enables you to start or stop services and drivers, and to change the startup type of those items. Use the filter group ntServiceFilter to select the type of items listed in this table.

Table 100 describes the information contained in ntServiceTable.

Table 100. NT services table (ntServiceTable)

Row object	Description
BinaryPath	The fully qualified path of the service's binary file. This object is read-only.

Table 100. NT services table (ntServiceTable) (continued)

Row object	Description
Control	<p>Sets the state of the monitored service. The effect of setting this variable depends on the CurrentState of the service:</p> <ul style="list-style-type: none"> <li>start(1) - If the service is in the stopped(7) state, the service will be started</li> <li>stop(2) - If the service is in the running(4) state, the service will be stopped</li> <li>pause(3) - If the service is in the running(4) state, the service will be paused</li> <li>resume(4) - If the service is in the paused(3) state, the service will be started</li> <li>restart(5) - If the service is in the running(4) state, the service will be restarted. If the service is in the stopped(7) state, it will be started</li> </ul>
CurrentState	<p>Indicates the current state of the service:</p> <ul style="list-style-type: none"> <li>continuePending(1) - Service continuation is pending</li> <li>pausePending(2) - Service pause is pending</li> <li>paused(3) - Service is currently paused</li> <li>running(4) - Service is currently running</li> <li>startPending(5) - Service start is pending</li> <li>stopPending(6) - Service stop is pending</li> <li>stopped(7) - Service is currently stopped</li> </ul> <p>This object is read-only.</p>
Dependencies	<p>A comma-delimited list of services that must be started before this service. An entry prefixed with a plus (+) symbol indicates a service group.</p> <p>This object is read-only.</p>
Description	<p>The string describing the purpose of the service. This object is read-only.</p>
DisplayName	<p>The string used by service control programs to identify a service. This object is read-only.</p>
ErrorControl	<p>Determines the action taken when the monitored service encounters an error:</p> <ul style="list-style-type: none"> <li>ignore(1) - The startup (boot) program logs the error but continues the startup operation</li> <li>normal(2) - The startup program logs the error and displays a message box pop-up but continues the startup operation</li> <li>severe(3) - The startup program logs the error. If the last known good configuration is being used, the startup operation continues. Otherwise, the system is restarted using the last known good configuration</li> <li>critical(4) - The startup program logs the error, if possible. If the last known good configuration is being started, the startup operation fails. Otherwise, the system is restarted with the last known good configuration.</li> </ul> <p>This object is read-only.</p>
Name	<p>The name of the service. This object is read-only.</p>
Parameters	<p>A list of the parameters passed to the service executable on startup. This object is read-only.</p>

Table 100. NT services table (ntServiceTable) (continued)

Row object	Description
StartName	<p>If the value of Type is ownProcess(3) or sharedProcess(4), this string contains the account name (in the form <i>DomainName\Username</i>) which the service process will be logged on as when it runs.</p> <p>If the service type is fileSystemDriver(1) or kernelDriver(2), this string contains the name of the driver object used to load this driver.</p> <p>This object is read-only.</p>
StartType	<p>Determines when the service is started:</p> <ul style="list-style-type: none"> <li>automatic(1) - The service is started automatically by the service control manager during system startup</li> <li>onBoot(2) - The service is started by the system loader. This is only valid for driver services</li> <li>onDemand(3) - The service is started when a process calls the StartService function</li> <li>disabled(4) - The service cannot be started</li> <li>system(5) - The service is started by the IoInitSystem function. This is only valid for driver services</li> <li>automaticDelayed(6) - Supports delayed start services</li> </ul>
Type	<p>Identifies the type of service:</p> <ul style="list-style-type: none"> <li>fileSystemDriver(1) - A file system driver service</li> <li>kernelDriver(2) - A driver service</li> <li>ownProcess(3) - A service that runs in its own process</li> <li>sharedProcess(4) - A service that shares a process with other services</li> <li>interactive(5) - A service that can interact with the desktop</li> </ul> <p>This object is read-only.</p>





---

## Chapter 17. Process

The process subagent provides facilities to monitor, start, stop and restart applications and processes running on a server or other network device. It can monitor multiple attributes of a running process, including status, memory usage, size and resident set size, (RSS), process time, threads, and disk and network I/O.

The subagent identifies all processes running on the host machine and enables you to create threshold monitors for process attributes, such as CPU or memory usage, and generate an event or execute a command when a threshold is violated. By monitoring the response time, average throughput (bytes per second), as well as how much memory, CPU and network traffic each process or application uses, you can gather critical information about quality of service (QoS) and capacity planning.

---

### Component files

The process subagent and MIB module are comprised of a set of component files.

Table 101 lists the process subagent and MIB module component files and their locations.

*Table 101. process component files*

File	Location	Description
process.dll (Windows) libprocess.so/.sl (UNIX)	bin	Binary implementation of the process subagent.
process-mib.mib	mibs	process MIB definition document.
process.oid	config/oid	process subagent object identifier file.

---

### Guidelines

The process subagent operates independent of the host machine's platform.

To load the subagent, use the command:

```
subagent load process
```

### General procedure

To monitor a process, identify the process that you wish to monitor, together with the attributes of interest, then create a control row to collect data about those items.

#### Procedure

To configure process monitoring:

1. Using `psRunningTable`, locate the process that you wish to monitor.
2. Create a control row in `psControlTable` by setting the `psControlStatus` object to `CreateAndWait(5)`.
3. Using the objects `psControlFilterType` and `psControlFilterPattern`, identify the process or processes that you wish to monitor.

4. Set the `psControlSampleInterval` and `psControlSampleType` objects to values appropriate to the type of attribute that you wish to monitor.
5. Set the `psControlTestAttribute` to the process attribute that you wish to monitor.

The process attributes available are those provided in `psRunningTable`.

6. Using the `psControlTestThreshold` and `psControlTestOperator` objects set a threshold for the monitored process attribute and the type of comparison operation used when testing the attribute's value against the threshold.

When this threshold condition is violated, the control row generates a process exception, which is then logged in `psExceptionTable`.

7. If required, use the `psControlActionCommand` object to specify the command executed and the `psControlActionEventIndex` object to specify the event generated if the threshold condition is violated.

8. Set the `psControlStatus` object to `active(1)`.

The process exception table logs the process exceptions generated by the process control row.

## Allocating storage in the process exception table

The process exception table stores information about each process exception that occurs (that is, when the monitored process attribute violates the defined threshold).

The table stores a rolling history of exceptions associated with each control row. The `psControlBucketsRequested` object sets the maximum number of exception table rows requested for each control row and the `psControlBucketsGranted` object indicates the number of rows actually allocated. When the maximum number of rows allocated to a control row is reached, the subagent removes the oldest row from the table each time it adds a new row.

If the value of `psControlBucketsRequested` is 0, the exception table stores only those exceptions generated during the most recent sample interval. With each new sample, any existing rows are removed from the table, and the number of rows in the table allocated to the control row varies according to the number of exceptions generated in the sample interval.

## Specifying actions

To define the action performed whenever a process attribute violates the threshold value, set the `psControlActionCommand` object to an operating system command.

### About this task

The subagent executes the action command at the end of any sample interval in which a threshold violation occurs. If multiple processes produce exceptions during the same sample interval, the subagent executes the command multiple times, once for each process.

## Killing processes

To kill a process, set the command string to #.

### About this task

The kill command may be followed by another command, using the general format:

*#command*

where # is the kill command prefix and *command* represents the command to be executed subsequently. The subagent kills the process before executing the command.

**Note:** Do not insert a space between the # character and the command unless the command itself begins with a space.

### Special characters

The subagent supports insertion of values, represented by special character sequences, into the command string.

The following character sequences are provided:

- !p inserts the process ID
- !n inserts the short process name
- !a inserts arguments without the command
- !! inserts a ! character

## Using regular expressions to identify processes

You can use regular expressions to identify the process or processes that you wish to monitor.

If the value of the psControlFilterType object is set to name(1) or command(2), the process subagent interprets the psControlFilterPattern object as a regular expression representing either the name or the command string of the processes to be monitored.

When evaluating this expression, the subagent uses string matching on the process's name or command and argument string. To match a substring in the name or command string objects, use the expression *.\*string.\** where *string* denotes the desired substring. For more details on regular expression usage, see Appendix D, "Regular expressions," on page 615.

**Note:** On Windows platforms, express the filter pattern in upper case letters only.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The process commands create rows in the control table (psControlTable).

The general syntax of these commands is:

```
process property=value  
process create [property=value ...]  
process reset
```

Table 102 lists the properties supported in these commands.

Table 102. Configuration command parameters - process

Property	Type	Description	Sets MIB object
actioncmd	string	The command executed when the sample value violates the threshold condition.	ActionCommand
actionevent	int	The index of the event generated when the sample value violates the threshold condition.	ActionEventIndex
actioneventstatus	enum	Event flow control: alwaysready fired ready	ActionEventStatus
actionthresh	int	Sets the minimum time between command execution.	ActionThreshold
attr	enum	Specifies the process attribute to be monitored: age alive averageCpu averageCpu100 averageCpuPerCpu averageCpuPerCpu100 contextSwitches handles hardPageFaults kernelTime pageFaults priority rss size threads totalTime userTime	TestAttribute
buckets	int	Sets the maximum number of rows in the exception table assigned to this control row.	BucketsRequested
datacontrol	enum	Data control: on - Enables data collection off - Suspends data collection	DataControl
description	string	A description of the process monitor row.	Description
filter	string	A regular expression representing the process to be monitored. This expression is interpreted according to the value of the filtertype property.	FilterPattern

Table 102. Configuration command parameters - process (continued)

Property	Type	Description	Sets MIB object
filtertype	enum	Sets the function of the filter property: command commandOldest groupLeaderCommand groupLeaderName name nameOldest noCommand noName pid	FilterType
interval	int	Sets the sample interval (in seconds).	SampleInterval
oper	enum	Determines the operator used in comparing the sample value to the threshold: eq ge gt le lt ne topN	TestOperator
sampletype	enum	Determines how the sample values are derived: delta exact	SampleType
thresh	int	The threshold value against which each sample is tested.	TestThreshold
trapmask	unsigned	Determines which psRunningTable objects are inserted as varbinds into psException notifications. Each bit position corresponds to an entry in the psRunningEntry sequence. Objects are appended in order, starting from psRunningPID which corresponds to the least significant bit in the mask, to the object corresponding to the most significant bit. Table 103 specifies the corresponding bit position for each available object. Do a bitwise OR of the bits to enable the attributes to be appended to the notification.  The default is 0 which means that no additional objects are appended to the notification.	TrapMask

Table 103. TrapMask bit positions for available objects

Object	Bitmask
psRunningPID	0x00000001
psRunningPGID	0x00000002
psRunningPPID	0x00000004
psRunningName	0x00000008
psRunningState	0x00000010
psRunningPriority	0x00000020

Table 103. TrapMask bit positions for available objects (continued)

Object	Bitmask
psRunningSize	0x0000040
psRunningRSS	0x0000080
psRunningPageFaults	0x0000100
psRunningHardPageFaults	0x0000200
psRunningContextSwitches	0x0000400
psRunningThreads	0x0000800
psRunningHandles	0x0001000
psRunningPercentCPU	unsupported
psRunningKernelTime	0x0004000
psRunningUserTime	0x0008000
psRunningStartTime	0x0010000
psRunningUserName	0x0020000
psRunningDevice	0x0040000
psRunningCommandLine	0x0080000
psRunningFilePath	0x0100000
psRunningCWD	0x0200000
psRunningPercentRSS	unsupported
psRunningPercentSize	unsupported
psRunningZoneName	0x1000000

## Examples

These examples demonstrate how to use the process subagent to perform simple process monitoring and control tasks.

### Monitoring memory usage of all processes

Monitor all processes at one minute intervals, logging an exception for any process that is using more than 15 MB (15 000 KB) of memory. Maintain a list of the last 20 exceptions.

The subagent configuration commands required for creating this monitor are as follows:

```
subagent load process
process reset
process description="Monitor processes using more than 15MB"
process filtertype=name
process filter=.*
process interval=60
process sampletype=exact
process attr=size
process oper=gt
process thresh=15000
process buckets=20
process create
```

## Monitoring memory usage growth

Monitor all processes at one minute intervals, logging exceptions for those processes whose memory use grows by more than 50 KB/minute.

The subagent configuration commands required for creating this monitor are as follows:

```
subagent load process
process reset
process description="Monitor processes whose mem use grows at more than 50KB/min"
process filtertype=name
process filter=.*
process interval=60
process samplotype=delta
process attr=size
process oper=gt
process thresh=50
process buckets=20
process create
```

## Killing a process on high memory usage

Kill any process whose memory use grows faster than 2 MB in 10 seconds.

The subagent configuration commands required for creating this monitor are as follows:

```
subagent load process
process reset
process description="Kill processes whose mem use grows at more than 2MB/10 seconds"
process filtertype=name
process filter=.*
process interval=10
process samplotype=delta
process attr=size
process oper=gt
process thresh=2000
process buckets=20
process actioncmd=#
process create
```

## Monitoring and controlling a single process

Monitor the NOTEPAD process at 10 second intervals and ensure that it is stopped when its memory use exceeds 10 MB, releasing memory consumed by memory leaks.

The subagent configuration commands required for creating this monitor are as follows:

```
subagent load process
process reset
process description="Kill the Notepad process if its mem use exceeds 10MB"
process filtertype=name
process filter=NOTEPAD.*
process interval=10
process samplotype=exact
process attr=size
process oper=gt
process thresh=10000
process actioncmd=#
process create
```

## Keeping a process running

Ensure that the NOTEPAD application is always available in memory by restarting it if it stops running.

The subagent configuration commands required for creating this monitor are as follows:

```
subagent load process
process reset
process description="Keep Notepad running"
process filtertype=name
process filter=NOTEPAD.*
process interval=1
process sampletype=exact
process attr=alive
process oper=eq
process thresh=0
process actioncmd=NOTEPAD
process create
```

## Top 10 CPU usage

Monitor the top 10 processes in terms of average CPU usage.

The subagent configuration commands required for creating this monitor are as follows:

```
subagent load process
process reset
process description="Determine the top 10 CPU users"
process filtertype=name
process filter=.*
process interval=10
process sampletype=delta
process attr=averageCpu
process oper=topn
process buckets=10
process create
```

---

## MIB module

The process MIB is a subtree of networkharmoni(1977).

The process subtree is shown in Figure 36 on page 183.



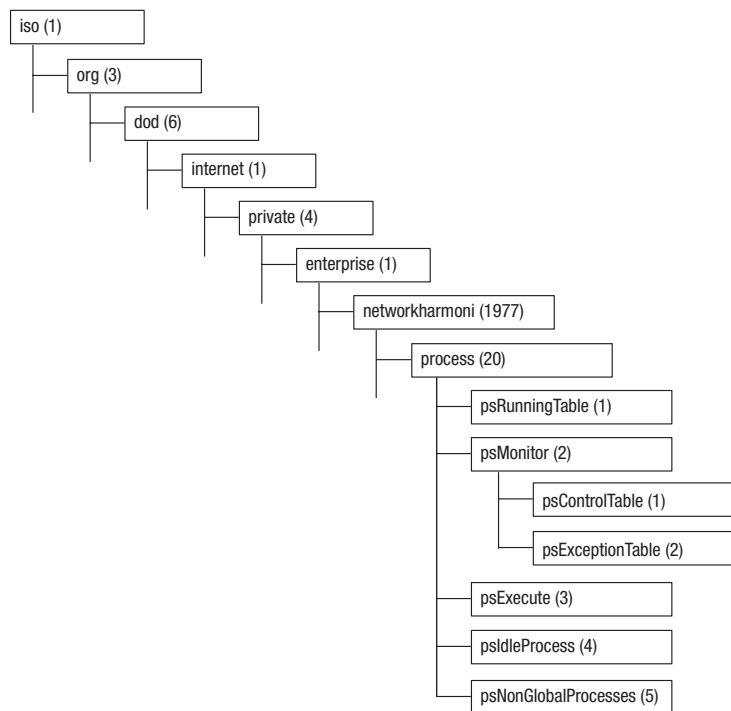


Figure 36. OID tree diagram of the process MIB module

This section provides a summary of the objects defined in the process MIB module. For detailed information on all objects in the module, see the `process-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB tables

The process MIB provides a standard SNMP interface with control row semantics.

The MIB module contains:

- Process running table, `psRunningTable`  
Lists the processes running on a host.
- Process monitor control table, `psControlTable`  
Configures monitoring and control of host processes.
- Process exception table, `psExceptionTable`  
Lists process exceptions caught on a host.

## Process running table

The process running table (psRunningTable) lists all the processes running on a host. Each row in the table represents one process. You cannot configure table rows themselves because they contain information about the processes running; however, you can control the process by setting its psRunningState row object.

Table 104 lists the objects in psRunningTable.

*Table 104. Process running table (psRunningTable)*

Row object	Description
CommandLine	The exact command including parameters used to start this process.
ContextSwitches	The total number of context switches this process has undergone.
CWD	The full path of the process's current working directory, if known. This value is useful in diagnosing problems with removing directories that are busy.
Device	The name of the user interface device associated with this process.
FilePath	The full path of the program file associated with this process.
Handles	The total number of operating system handles, or file descriptors on UNIX systems, that this process currently has open.
HardPageFaults	The total number of hard page faults that this process has generated since starting. Hard page faults are those that require I/O (virtual memory access) to complete.
KernelTime	The total amount of CPU time consumed in kernel calls for this process. On multi-CPU machines this value can represent an amount of time greater than the amount of time over which the process has been running.
Name	The short process name, which is typically the executable file's name without a path or arguments.
PageFaults	The total number of page faults, both hard/major and soft/minor, that this process has generated since starting.
PercentCPU	An approximation of the percentage of CPU time currently consumed by this process. This value is calculated by dividing this process's kernel and user time by the total time available to all CPUs. A value of 100 indicates that this process is occupying all CPUs on the entire machine.
PercentRSS	The percentage of physical memory occupied by the process's resident set of pages. The value of this object is highly volatile and is in no way an indicator of memory requirements. The value can easily vary from 0 to 100 without the process allocating or freeing any memory. The psRunningPercentSize provides a more stable indication of memory use.
PercentSize	The relative percentage of physical memory, not all of which is necessarily resident, allocated to a process. Values greater than 100 are possible and such values indicate that a process is using more physical RAM than is available.
PGID	The process group ID of this process, which is the PID of the process that started the logical group of processes to which this process belongs. On Windows, this value is the WindowsTerminal Services session ID if the process has one otherwise it value will be zero. On other platforms that do not support process groups, this value is simply the PID of the process.
PID	The unique process identifier (PID) of this process.

Table 104. Process running table (psRunningTable) (continued)

Row object	Description
PPID	The PID of the process's parent.
Priority	<p>The UNIX-style static priority value of the process. On non-UNIX systems, this value is a modified base priority value that follows the nice-value convention:</p> <ul style="list-style-type: none"> <li>• The value is between -20 and 20</li> <li>• Lower values represent higher priorities</li> <li>• Negative values are reserved for the system and administrator</li> </ul>
RSS	The amount of memory (in KB) resident in physical or main memory that is allocated to this process.
Size	The total amount of memory (in KB) allocated to this process.
StartTime	The time and date when this process started.
State	<p>The current state of this process:</p> <p>kill(0) - This is not a real state, but setting the object to this value kills the process. First a polite kill is attempted (for example, using SIGTERM on UNIX systems). If the process does not die within 20 seconds, the subagent issues a forceful kill (for example, using SIGKILL on UNIX systems).</p> <p>starting(1) - The process is being created or initialized and is not yet a candidate for any of the other states.</p> <p>suspended(2) - The process has been temporarily prevented from executing. Setting the object to this value suspends a process.</p> <p>sleeping(3) - The process is idle, awaiting an event.</p> <p>running(4) - The process is in the run queue or currently running. Setting the object to this value if the process's current state is suspended(4) causes it to resume execution.</p> <p>zombie(5) - The process has exited, but has not yet been freed. Only the values kill(0), suspended(2) and running(4) may be written to this object.</p>
Threads	The total number of threads that this process currently has.
UserName	The name of the user who owns this process.
UserTime	The total amount of CPU time consumed in user code (non-kernel code) for this process. On multi-CPU machines this value can represent an amount of time greater than the amount of time over which the process has been running.
ZoneName	(Solaris 10 or later) Displays the Name of the zone in which the processes are running. (Read-only)

## Process monitor control table

The process monitor control table (psControlTable) configures process monitors. Each row represents a monitor for a specific attribute of a process or set of processes. The monitor compares the process attribute against a defined threshold and any violations of the threshold, known as process exceptions, are logged in the process exception table.

Table 105 lists the objects in psControlTable.

Table 105. Process monitor control table (psControlTable)

Row object	Description
ActionCommand	Specifies the command executed when the sample value violates the threshold condition. If this object is empty, no command is executed. If multiple processes produce exceptions during the same sample then this command is executed multiple times, once for each process.
ActionEventIndex	The index of the event triggered when the sample value violates the threshold condition. Both this event and the command specified by psControlActionCommand are triggered simultaneously when a process exception occurs.  If the value of this object is zero, no event is triggered.
ActionEventStatus	The RMON-style event throttle for the event indicated by psControlActionEventIndex:  ready(1) - The event can be fired once, after which the value of this object becomes fired(2) and stays there until reset.  fired(2) - After an event with state ready(1) has fired it's status changes to this value.  alwaysReady(3) - The event can be fired any number of times.
ActionThreshold	Provides an action frequency limit that sets the minimum allowed time (in seconds) between action invocations. If the time between two successful tests of the sample value is less than that specified by this object, the action is considered to be triggering too frequently. In this case the value of psControlStatus is set to notInService(2) and the event (if set) is triggered.  If the value of this object is zero, action frequency checking is disabled, guaranteeing that the control row will remain in service regardless of the action frequency.
BucketsGranted	Indicates the number of rows that the subagent can log in psExceptionTable for this control row, up to the maximum number indicated by psControlBucketsRequested. When this limit is reached the oldest entry associated with this control row is deleted each time a new row is added.
BucketsRequested	Sets the maximum number of rows that may be created in psExceptionTable for this control row.  When the value of this object is 0, the row allocation behavior in psExceptionTable changes so that the number of rows varies to match the number of exceptions generated. With each new sample, any existing rows are removed from the table.
CreateTime	Indicates the value of sysUpTime when this entry was last activated.

Table 105. Process monitor control table (psControlTable) (continued)

Row object	Description
DataControl	Controls data collection:  on(1) - Enables data collection  off(2) - Disables data collection
Description	A description of the control row.
FilterPattern	If the value of the psControlFilterType object is name(1) or command(2), the subagent interprets this object as a regular expression representing the entire name or command string of the processes to be monitored. If the value of the psControlFilterType object is pid(4), the subagent interprets this object as a textual representation of the exact process ID of the process to be monitored.
FilterType	Specifies how the psControlFilterPattern is interpreted to identify the process or processes monitored:  name(1) - Matches a process's short name (psRunningName)  command(2) - Matches a process's full command and argument string  groupLeaderName(3) - Matches the short name of all process group leaders (process group leaders are those for which process ID = process group ID)  pid(4) - Matches a specific process ID  noName(5) - Checks that no process names match the filter specified by psControlFilterPattern, and if so, skip attribute checking and start an action.  nameOldest(6) - Matches the oldest process whose short name corresponds to that specified by psControlFilterPattern  noCommand(7) - Similar to noName(5) except that it operates on the process command string rather than the short name.  groupLeaderCommand(8) - Matches a process's command line, but only for process group leaders  commandOldest(9) - Matches the oldest process whose command line corresponds to psControlFilterPattern  The contents of any object resulting from a noName(5) or noCommand(7) match in psExceptionTable or in traps are undefined.
Index	Uniquely identifies the row.
LastFilterCount	Indicates the number of processes in the last sample that matched the filter. This value does not provide any indication of the number of processes that passed the test.
LastUpdate	Indicates the value of sysUpTime when the last sample for this entry was taken.
Owner	Indicates the owner and creator of the control row.
SampleInterval	Sets the sample interval (in seconds). This specifies how frequently the subagent checks the list of running processes, performs filter matching and tests the given process attribute.

Table 105. Process monitor control table (psControlTable) (continued)

Row object	Description
SampleType	<p>Indicates how sample values are derived:</p> <p>exact(1) - The sample value is the exact value of the process attribute.</p> <p>delta(2) - The sample value is the difference between the current value of the process attribute and that during the previous sample.</p>
Status	Controls creation, activation and deletion of the control row.
TestAttribute	<p>Specifies the process attribute to be tested:</p> <p>alive(1) - If the process is alive, the value of this attribute is 1(true). A value of 0(false) indicates that the process existed in the previous sample but has since died.</p> <p>size(2) - The value of psRunningSize.</p> <p>rss(3) - The value of psRunningRSS.</p> <p>priority(4) - The value of psRunningPriority.</p> <p>pageFaults(5) - The value of psRunningPageFaults.</p> <p>hardPageFaults(6) - The value of psRunningHardPageFaults.</p> <p>contextSwitches(7) - The value of psRunningContextSwitches.</p> <p>threads(8) - The value of psRunningThreads.</p> <p>handles(9) - The value of psRunningHandles.</p> <p>totalTime(10) - psRunningKernelTime + psRunningUserTime.</p> <p>kernelTime(11) - The value of psRunningKernelTime.</p> <p>userTime(12) - The value of psRunningUserTime.</p> <p>age(13) - The amount of time (in seconds) that the process has been running.</p> <p>averageCpu(14) - The percentage of total CPU usage on an n-CPU machine, determined as totalTime/(total time available for all CPUs). To monitor average CPU usage over a period of time, set psControlSampleType to delta(2). This value has no meaning when psControlSampleType is set to exact(1) and is not permitted by the subagent.</p> <p>averageCpu100(15) - averageCpu * 100, providing accuracy to two decimal places. The value 100 represents 1% and 10000 represents 100%.</p> <p>averageCpuPerCpu(16) - A percentage of CPU usage per CPU calculated as totalTime/SampleInterval. A value of 100 indicates that the process is occupying one entire CPU. <i>This value may exceed 100 on machines with multiple CPUs able to run multiple threads of the same process in parallel.</i> To monitor average CPU usage over a period of time, set psControlSampleType to delta(2). This value has no meaning when psControlSampleType is set to exact(1) and is not permitted by the subagent.</p>

Table 105. Process monitor control table (*psControlTable*) (continued)

Row object	Description
TestAttribute <i>cont'd</i>	<p>averageCpuPerCpu100(17) - averageCpuPerCpu * 100, providing accuracy to two decimal places. The value 100 represents 1% and 10000 represents 100%.</p> <p>percentSize(18) - As defined in psRunningTable.</p> <p>lastFilterCount(19) - The number of processes in the last sample that matched the filter. This attribute cannot be tested if the psControlTestOperator is defined as topN. In this mode, the corresponding psExceptionPID object has the value 0, and the psExceptionMatch object is empty.</p>
TestOperator	<p>Specifies the logical operator applied to psControlTestAttribute and psControlTestThreshold:</p> <p>eq(1) - Equal</p> <p>ne(2) - Not equal</p> <p>gt(3) - Greater than</p> <p>lt(4) - Less than</p> <p>ge(5) - Greater than or equal to</p> <p>le(6) - Less than or equal to</p> <p>topN(7) - Selects <i>n</i> processes with the highest value of psControlTestAttribute, where <i>n</i> is equal to the value of psControlBucketsRequested.</p> <p>If the value of this object is topN(7), psControlTestThreshold has no effect.</p>
TestThreshold	The numeric value, of type psControlTestAttribute, against which each sample is tested.
TrapMask	<p>Determines which psRunningTable objects are inserted as varbinds into psException notifications. Each bit position corresponds to an entry in the psRunningEntry sequence starting with psRunningPID corresponding to the least significant bit in the mask. Objects are appended in order, starting from psRunningPID to the object corresponding to the most significant bit.</p> <p><b>Note:</b> The psRunningPercentCPU, psRunningPercentRSS and psRunningPercentSize objects cannot be inserted. If any of these bits are set, they are silently ignored.</p> <p>The default value (0) inserts no additional varbinds.</p>

## Process exception table

The process exception table (`psExceptionTable`) logs exceptions generated by the process monitors defined in the control table. Process monitors generate exceptions when the sampled process attribute specified by `psControlTestAttribute` meets the test criteria set by `psControlTestOperator` and `psControlTestThreshold`.

Each row represents one process exception and the number of rows allocated to a process monitor is determined by the control row's `psControlBucketsRequested` object. When the number of rows reaches this limit, the oldest entry associated with the control row is deleted each time a new row is added. Table 106 lists the objects in `psExceptionTable`.

Table 106. Process exception table (`psExceptionTable`)

Row object	Description
Match	The process name or command that matched <code>psControlFilterPattern</code> . If <code>psControlFilterType</code> is <code>pid(4)</code> , this object is empty.
PID	The PID of the process that generated this exception. This index distinguishes multiple exceptions that occur in the same sample.
Time	The value of <code>sysUpTime</code> for the sample in which this exception occurred. If the value of <code>psControlBucketsRequested</code> is 0, the value of this object is also 0.
Value	The sampled value that met the test criteria specified in the control row, causing this exception.

## Notification types

The process MIB implements the `psException` notification type for events generated when a process exception occurs.

Figure 37 shows the OID tree diagram for the `psException` trap.

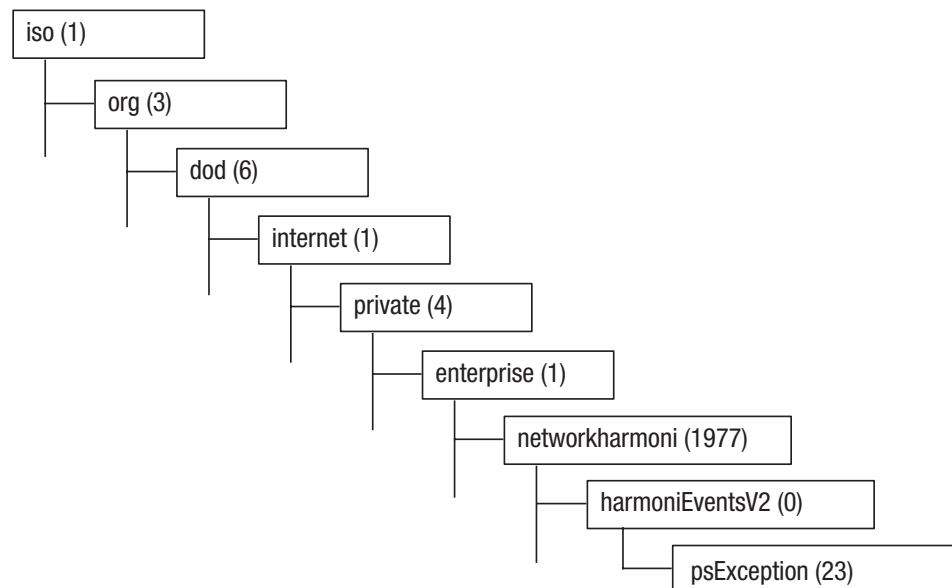


Figure 37. OID tree diagram for `psException` notification type



This notification type contains variable bindings indicating what was being monitored and the process exception that occurred. Table 107 lists the variable bindings.

Table 107. *psException* notification type

Notification type	Description
psException	<p>Generated when a process exception occurs.</p> <p>Variable bindings:</p> <ul style="list-style-type: none"><li>psControlStatus</li><li>psControlOwner</li><li>psControlDescription</li><li>psControlFilterType</li><li>psControlFilterPattern</li><li>psControlSampleInterval</li><li>psControlSampleType</li><li>psControlTestAttribute</li><li>psControlTestOperator</li><li>psControlTestThreshold</li><li>psExceptionMatch</li><li>psExceptionValue</li></ul> <p><b>Tip:</b> Additional varbinds from the psRunningTable may be inserted by setting the appropriate bit flags in psControlTrapMask. See Table 105 on page 186 for more information.</p>



---

## Chapter 18. Process handle

The process handle subagent, `prochandle`, and its MIB module collect data about handles associated with processes running on the host machine.

The subagent collects data about network, pipe, and file handles and makes this data available for monitoring and diagnostic activities such as application discovery and port mapping.

---

### Component files

The `prochandle` and MIB module are comprised of a set of component files.

Table 108 lists the subagent and MIB module component files and their installed locations.

*Table 108. prochandle component files*

File	Location	Description
<code>prochandle.dll</code> (Windows) <code>prochandle.sl/.so</code> (UNIX)	<code>bin</code>	Binary implementation of the <code>prochandle</code> subagent.
<code>prochandle-mib.mib</code>	<code>mibs</code>	MIB definition document.
<code>prochandle.oid</code>	<code>config/oid</code>	<code>prochandle</code> subagent object identifier file.

---

### Guidelines

To monitor process handles on the host machine, create a control row that identifies the process whose handles you wish to monitor.

#### Before you begin

Ensure that the `prochandle` subagent is loaded. To load the subagent, use the command:

```
subagent load prochandle
```

#### Procedure

To create a control row in `phControlTable`:

1. Set `phControlStatus` to `CreateAndWait(5)`.
2. Set `phControlProcess` to a regular expression matching the name of the process whose handles you wish to monitor.  
To monitor all processes, use `.*`.
3. Set `phControlFilter` to indicate the types of handle you wish to monitor.  
To monitor all types of handle on all ports, use the string `ANY:*`.
4. Set `phControlInterval` to the interval (in seconds) at which you wish to gather data.
5. Activate the control row by setting `phControlStatus` to `active(1)`.  
The subagent gathers data about the process handles and inserts it into the data and port tables, `phDataTable` and `phPortTable`, at the interval specified.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### Control table

The prochandle commands create rows in the prochandle control table (phControlTable).

The general syntax of these commands is:

```
prochandle property=value
prochandle create property=value ...
prochandle reset
```

Table 109 lists the properties supported in these commands.

*Table 109. Configuration command parameters - prochandle*

Property	Type	Description	Sets MIB object
dataControl	enum	Data control:  on - Enables data collection.  off- Suspends data collection.	DataControl
description	string	A description of the control row.	Description
filter	string	Specifies the type and port of the process handles to monitor, in the format: <i>type:port[,port] [type:port ...]</i>  The <i>type</i> may be TCP, UDP, PIPE, FILE, or ANY.	Filter
process	string	A regular expression specifying the name of the process to monitor. This expression matches the entire process name.	Process
updateInterval	int	The interval (in seconds) at which process handle data is sampled.	Interval

---

## Examples

These examples show how to use the prochandle subagent to collect data.

### Monitor all processes

Collect data about all types of process handles used by all processes running on the host machine, at 15-minute intervals:

```
subagent load prochandle
prochandle reset
prochandle description="Monitor all processes and handles"
```

```
prochandle process=".*"  
prochandle filter="ANY:*"   
prochandle updateInterval=900  
prochandle create
```

## Discover SNMP agents

Discover the SNMP agent currently running in the host machine:

```
subagent load prochandle  
prochandle reset  
prochandle description="Discover SNMP agent"  
prochandle process=".*"  
prochandle filter="UDP:161"  
prochandle create
```

## Discover Web servers

Discover any Web servers using the standard ports 80, 8080, and 443 on the host machine:

```
subagent load prochandle  
prochandle reset  
prochandle description="Discover Web servers"  
prochandle process=".*"  
prochandle filter="TCP:80,8080,443"  
prochandle create
```

## Find processes that access a file

Find all processes that have the file /var/log/messages open:

```
subagent load prochandle  
prochandle reset  
prochandle description="Access to /var/log/messages"  
prochandle process=".*"  
prochandle filter="FILE:/var/log/messages"  
prochandle create
```

## Find all TCIP/IP endpoints

Find all TCP/IP endpoints on the host machine:

```
subagent load prochandle  
prochandle reset  
prochandle description="TCP/IP endpoints"  
prochandle process=".*"  
prochandle filter="TCP:* UDP:*"   
prochandle create
```

## Find Windows Directory Service processes

Find all Windows Directory Service processes running on the host machine:

```
subagent load prochandle  
prochandle reset  
prochandle description="Find Windows Directory Services"  
prochandle process=".*"  
prochandle filter="TCP:135,445 UDP:135,445"  
prochandle create
```

---

## MIB module

The procHandle MIB is a subtree of networkharmoni (1977).

The procHandle subtree is shown in Figure 38.

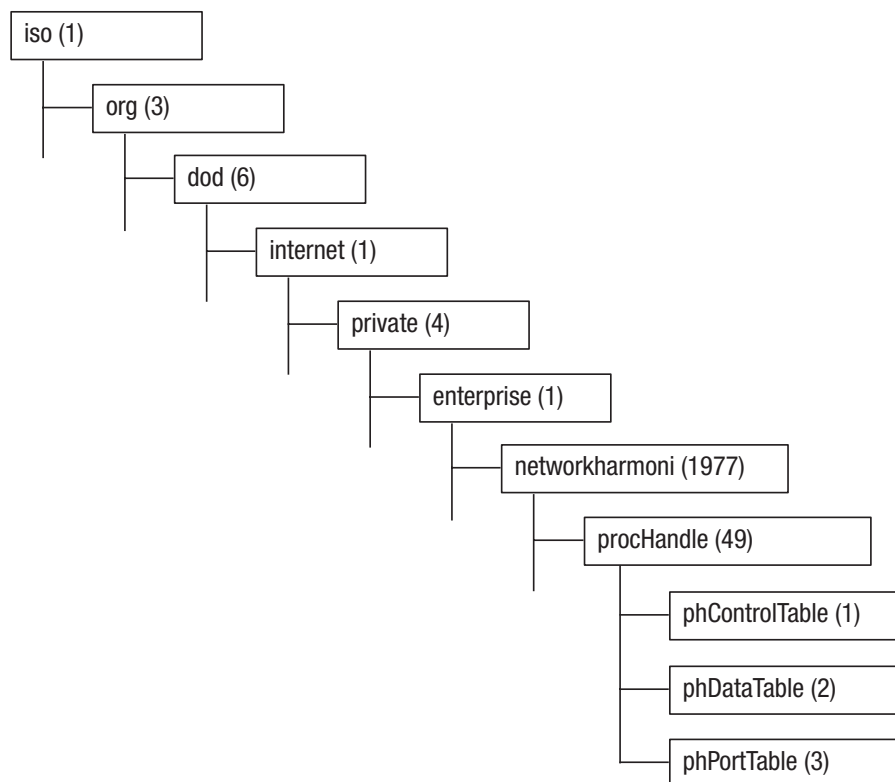


Figure 38. OID tree diagram of the procHandle MIB module

This section provides a summary of the objects defined in the procHandle MIB module. For detailed information on all objects in the module, see the procHandle-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

### Control table

The control table (phControlTable) contains a list of control information for configuring process handle mapping.

Table 110 lists the objects in phControlTable.

Table 110. procHandle control table (phControlTable)

Row object	Description
DataControl	Controls the monitoring status of the row:  on(1) - Data collection is enabled.  off(2) - Data collection is disabled.
Description	A text description of the control row.

Table 110. *procHandle* control table (*phControlTable*) (continued)

Row object	Description
Filter	Specifies the handle type and port number being monitored. This filter string has the format: <code>type:port[,port] [type:port ...]</code> where: <code>type</code> is the type of process handle: TCP, UDP, PIPE, FILE, ANY. <code>port</code> is the port number. * indicates all ports. The filter string must contain both a handle type and a port number to be valid. Default: ANY:*
Index	Uniquely identifies the row.
LastUpdate	Contains the value of <code>sysUpTime</code> when the corresponding data table entries were updated. This is the time at which the update completed, not when it started.
NumberHandles	Indicates the number of handles in the data table associated with the control row.
Owner	Identifies the owner/creator of the control row.
Process	A regular expression for matching process names. Regular expressions match entire strings.
Status	SNMPv2 row status. Controls creation, activation and deletion of the control row.
UpdateInterval	Sets the interval in seconds at which the data table is updated.

## Data table

The data table (*phDataTable*) contains a list of handles for the processes monitored by a *phControlTable* entry. Each row provides data about one handle and is indexed by a combination of *phControlIndex* and *phDataHandleIdx* objects.

Table 111 lists the objects in *phDataTable*.

Table 111. *procHandle* data table (*phDataTable*)

Row object	Description
DescriptorValue	The numerical value of the handle, if appropriate.
HandleIdx	Uniquely identifies the handle.
LocalName	A string representation of the local name of the handle. For files, this is the file name. For sockets, this is a local endpoint.
Pid	The PID of the process to which the handle belongs.
ProcName	The name of the process to which the handle belongs.
RemoteName	A string representation of remote name of the handle. For files this is blank. For sockets this is the remote endpoint.
Type	The type of handle.

## Port table

The port table (phPortTable) contains a list of handles for the processes monitored by a phControlTable entry, ordered by port number. The data in this table is the same as that provided in phDataTable but with a different indexing scheme.

Rows in this table are indexed by a combination of phControlIndex, phPortPortNumber, phPortPid, and phPortDescriptorValue objects. The only major difference is that on some platforms ports that are no longer associated with a process may only be shown once in this table, whereas such a port may appear twice in the PhDataTable. Table 112 lists the objects in phPortTable.

*Table 112. procHandle port table (phPortTable)*

Row object	Description
DescriptorValue	The numerical value of the handle, if appropriate.
LocalName	A string representation of the local name of the handle. For files, this is the file name. For sockets, this is a local endpoint.
Pid	The PID of the process to which this handle belongs.
PortNumber	The port number that this handle represents. If the handle is a file, this object has the value 100000.
ProcName	The name of the process to which the handle belongs.
RemoteName	A string representation of remote name of the handle. For files this is blank. For sockets this is the remote endpoint.
Type	The type of handle.



---

## Chapter 19. Programmable

The programmable subagent can invoke any program script to be executed manually or in response to specific events. Using this feature you can add customized functionality to the agent, allowing it to take action on specific events.

The programmable subagent and MIB module enable you to execute programs such as Perl and Java scripts or binary executables on the host machine. This facility is useful for performing network and enterprise management tasks remotely and it enables you to configure program execution in response to events and alarms.

---

### Component files

The programmable subagent and MIB module are comprised of a set of component files.

Table 113 lists the programmable subagent and MIB module component files and their locations.

*Table 113. programmable component files*

File	Location	Description
programmable.dll (Windows) libprogrammable.so/.sl (UNIX)	bin	Binary implementation of the programmable subagent.
programmable-mib.mib	mibs	programmable MIB definition document.
programmable.oid	config/oid	programmable subagent object identifier file.

---

### General procedure

Use the control table to configure program or command execution.

#### Before you begin

Ensure that the programmable subagent is loaded. To load the subagent use the command:

```
subagent load programmable
```

#### Procedure

To create a control row in programmableControlTable:

1. Set programmableControlStatus to CreateAndWait(5).
2. Set programmableControlApplication to the command or application you wish to execute.

See "Specifying programs" on page 201 for more details.

3. If required, configure event-based control of the program's execution:
  - a. Set programmableControlTurnOnEventIndex to the RMON eventIndex of the event that activates the program.

- b. Set `programmableControlTurnOffEventIndex` to the RMON `eventIndex` of the event that deactivates the program.

See “Event-triggered execution” on page 203 for more details.

4. Set `programmableControlVM` to the required program execution and input/output mode, and configure the input table (`programmableInTable`) and output table (`programmableOutTable`) accordingly.

See “Program execution and I/O modes” on page 202 for details.

5. If required, configure event generation for the program's execution. Set `programmableControlEventIndex`, `programmableControlEventStatus`, and `programmableControlEventCondition` to the desired event behavior.

See “Event generation” on page 203 for details.

6. Activate the control row by setting `programmableControlStatus` to `active(1)`.

The `programmableControlProgramStatus` object indicates the current state of the program. See for “Program status” more information.

## Program status

The `programmableControlProgramStatus` object controls execution of the program. Its value both indicates and controls the state of program execution.

Figure 39 illustrates the relationship between the values of `ProgrammableControlProgramStatus`.

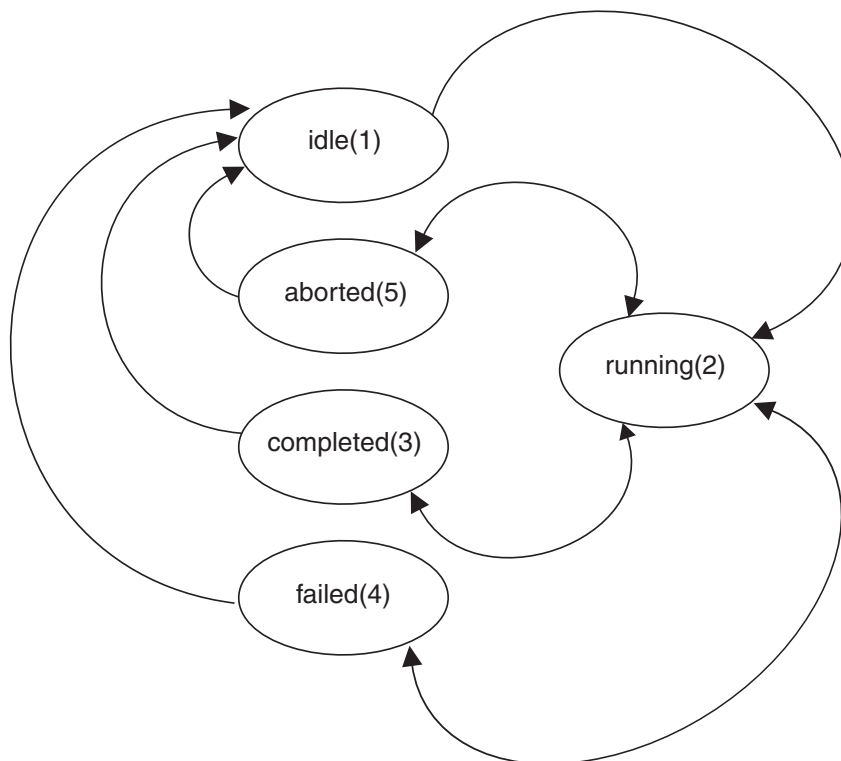


Figure 39. Program status modes

The following restrictions apply to the way that the program status object may be modified:

- The control row must be active
- The current status must be:

- running(2) if the required status is aborted(5)
- completed(3), aborted(5) or failed(4) if the required status is idle(1)

No other operations on programmableControlProgramStatus are permitted.

## Specifying programs

The programmableControlApplication object specifies the program executed. The contents of this object form the command line executed by the subagent. This command may be an operating system command or utility, or a call to an application or interpreter such as Perl or Java.

The subagent verifies that the command specified in programmableControlApplication is an executable file. The command must contain the full path to the executable unless the command is located in the system path. To specify Windows commands, such as copy, that are not executable files use an explicit call to the Windows command interpreter of the format `cmd.exe /c command`.

**Note:** Processes started by the programmable subagent, have the same current working directory as the SSM process.

### Command string tokens

The programmable MIB provides a set of tokens that represent special values in the command string defined in programmableControlApplication. The subagent performs substitutions on these tokens before executing the command.

Table 114 describes the tokens.

Table 114. Command string tokens

Token	Description
!!	Inserts an exclamation character (!) into the command string.
!n	Inserts the value of the <i>n</i> th varbind of the event that triggered the control row into the command string. If there is no matching varbind, an empty string is substituted. The allowed range of <i>n</i> is 1 to 99.
\$AGENTDIR	Represents the name of the Netcool/SSM bin directory. During command execution, the subagent substitutes this token with the path of Netcool/SSM bin directory.
\$CONFIGDIR	Represents the name of the Netcool/SSM config directory. During command execution, the subagent substitutes this token with the path of the Netcool/SSM config directory.
\$INPARAM	Directs input into commands that require an input file instead of using stdin. During command execution, the \$INPARAM token is substituted with the name of a temporary file, the contents of which are created from the programmableInTable rows associated with the control row.
\$OUTPARAM	Redirects output produced by commands that require an output file instead of using stdout. During command execution, the \$OUTPARAM token is replaced with the name of a temporary file. When the command has finished execution, the subagent loads the contents of this file into the programmableOutTable.
%variable_name%	Inserts the environment variable indicated by <i>variable_name</i> into the command string. For example, %VAR% expands the VAR environment variable.
%%	Inserts a single percentage (%) character.

Table 114. Command string tokens (continued)

Token	Description
\	UNIX only. Escapes a character in the command string.

Figure 40 demonstrates the operation of the \$INPARAM and \$OUTPARAM tokens.

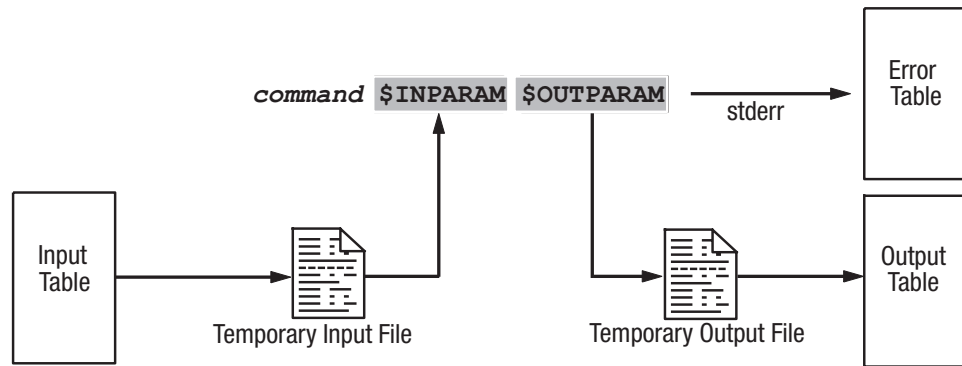


Figure 40. Operation of \$INPARAM and \$OUTPARAM tokens

## Program execution and I/O modes

The programmableControlVM object sets the program execution and input/output mode.

The values of this object have the following effects:

- vmNorm(1) - Takes stdin from programmableInTable. stdout is directed to programmableOutTable. stderr is directed to programmableErrTable.
- vmIFile(2) - Reads the file indicated by programmableControlInFile into the programmableInTable before executing the command specified by programmableControlApplication and before processing \$INPARAM.
- vmIOFile(3) - Specifies both vmIFile(2) and vmOFile(4) modes.
- vmOFile(4) - Writes the contents of the programmableOutTable to the file indicated by programmableControlOutFile after executing the command specified by programmableControlApplication and processing \$OUTPARAM.
- vmConsole(5) - Executes Netcool/SSM configuration commands. Takes stdin from programmableInTable. stdout is directed to programmableOutTable. stderr is directed to programmableErrTable.
- vmConsoleIFile(6) - Executes Netcool/SSM configuration commands. Reads the file indicated by programmableControlInFile into the programmableInTable before executing the command specified by programmableControlApplication.
- vmConsoleIOFile(7) - Specifies both vmConsoleIFile(6) and vmConsoleOFile(8) modes.
- vmConsoleOFile(8) - Writes the contents of the programmableOutTable to the file indicated by programmableControlOutFile after executing the Netcool/SSM configuration command specified by programmableControlApplication.

Figure 41 on page 203 illustrates how the input, output and data tables relate to program execution.

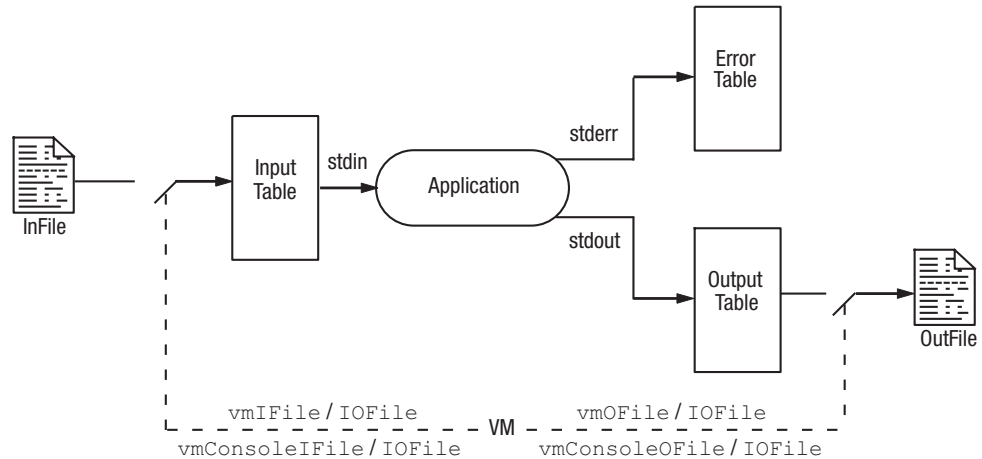


Figure 41. Program execution and I/O modes

**Note:** When executing Netcool/SSM configuration commands using any of the vmConsole modes, the subagent stores a maximum 10000 output lines in the output table, however it stores the *entire* output of the command in the file indicated by programmableControlOutFile.

## Event-triggered execution

Using the programmableControlTurnOnEventIndex and programmableControlTurnOffEventIndex objectst, you can configure the program to automatically activate or deactivate when an event occurs.

- To automatically activate the program when an event occurs, set programmableControlTurnOnEventIndex to the eventIndex of the triggering event.
- To deactivate the program when an event occurs, set the programmableControlTurnOnEventIndex to the eventIndex corresponding event.

## Event generation

Control rows can generate events upon completion of command execution. To configure a control row to generate an event on command execution, create an RMON event row then set programmableControlEventIndex to the eventIndex value of the event.

The programmableControlEventStatus defines how often events are generated. The default value for eventstatus is eventReady(1), which generates only a single event upon completion. When the program completes, the programmableControlEventStatus is changed to eventFired(2), which prevents subsequent completions from generating events. To allow the program to generate an event for every application completion, the programmableControlEventStatus must be initialized to eventAlwaysReady(3).

The programmableControlEventCondition object determines the conditions under which the event if fired: either on successful execution, failed execution or in all cases.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### Control table

The program commands create rows in the programmable control table (programmableControlTable).

The general syntax of these commands is:

```
program property=value
program create [property=value ...]
program reset
```

Table 115 lists the properties supported in these commands.

*Table 115. Configuration command parameters - program*

Property	Type	Description	Sets MIB object
application	string	Specifies the program to be executed.	Application
community	string	Sets the community string associated with the program.	Community
description	string	A description of the control row.	Description
event	int	The index of the event generated when program execution is complete.	EventIndex
eventcondition	enum	Specifies the condition under which the execution complete event is fired: fireonsuccess fireonfailure fireoneither	EventCondition
eventstatus	enum	Event flow control: alwaysready fired ready	EventStatus
infile	string	The file for program input.	InFile

Table 115. Configuration command parameters - program (continued)

Property	Type	Description	Sets MIB object
lineouteventindex	int	This index identifies the event that is fired for each line output (programmableOutOut) generated by the corresponding programmableControlEntry program. If no event is intended for this programmableControlEntry, set to zero.  This event is not affected by the current value of programmableControlEventStatus and is generated as if set to eventAlwaysReady.	LineOutEventIndex
offevent	int	The index of the event that deactivates the control row.	TurnOnEventIndex
onevent	int	The index of the event that activates the control row.	TurnOnEventIndex
outfile	string	The file for program output.	OutFile
status	enum	Controls the status of program execution: aborted idle running	ProgramStatus
vm	enum	Sets the program execution and I/O modes: console console-ifile console-iofile console-ofile ifile iofile norm ofile oevent ioevent	VM

## In table

The programin commands create rows in the input table (programmableInTable) and assign them to the next control row created using the program create command.

The general syntax of these commands is:

```
programin property=value
programin store [property=value ...]
programin reset
```

Table 116 on page 206 lists the properties supported in these commands.

Table 116. Configuration command parameters - programin

Property	Type	Description	Sets MIB object
line	string	Input data for program execution.	In

## Examples

These examples demonstrate how to use the programmable subagent.

### Using input and output tables as stdin and stdout

Create four rows in the input table in the order shown below and sort them using the UNIX sort utility, which takes a set of strings as stdin and returns them to stdout sorted in alphabetical order.

1. zip-drive
2. cpu
3. memory
4. buffer

The configuration commands for implementing this example are:

```
subagent load genalarm
program reset
program description="Input/Output table demo"
program application="/bin/sort"
program vm=norm
# Create the strings to be sorted
programin store line="zip-drive"
programin store line="cpu"
programin store line="memory"
programin store line="buffer"
program create
# Store the control row index for later reference
thisIdx=$?
# Execute the program
snmp set $programmableControlProgramStatus.$thisIdx i 2
# Display the contents of the output table on the console
snmp walk $programmableOutOut.$thisIdx
```

### Event-triggered script execution

Monitor CPU usage on the host machine at 60-second intervals. If the CPU usage exceeds 90%, run the script /opt/netcool/ssm/config/action.csh. Redirect any output produced by the script from stdout to the file /opt/netcool/ssm/log/action.log. For debugging purposes, generate an event of type log-and-trap if the script fails to execute.

CPU usage information is provided by the System Resources srSystemCPUUsage object. Use the genalarm subagent in hysteresis mode with a lower threshold of 5%.

The configuration commands for implementing this monitor are:

```
subagent load sysres
subagent load genalarmsubagent load programmable

_actioncmd=/opt/netcool/ssm/config/action.csh
_actionlog=/opt/netcool/ssm/log/action.log
_cputhresh=90
_cpufallthresh=5
```



```

event description="CPU Usage threshold violation"
event create type=snmp-trap community=public
cpuevent=$?

event description="DEBUG: execution failed on script ../config/action.csh"
event create type=log-and-trap community=public
debugevent=$?

program reset
program description="Executes the action script when CPU usage exceeds 90%"
program application=$_actioncmd
program onevent=$cpuevent
program event=$debugevent
program eventcondition=fireonfailure eventstatus=alwaysready
program vm=ofile
program outfile=$_actionlog
program create

genalarm interval=60
genalarm riseseverity=warning fallseverity=info
genalarm vardescr="CPU Usage"
genalarm var="$srSystemCPUUsage.0"
genalarm type=absolute mode=hysteresis
genalarm risethresh=$_cputhresh fallthresh=$_cpufallthresh
genalarm riseevent=$cpuevent
genalarm risedescr="CPU usage has been above $$7% for the last minute
(currently $$6%)"
genalarm falldescr="CPU usage has returned to a normal level (less than $$7%)"
genalarm create

```

---

## MIB module

The programmable MIB is a subtree of networkharmoni (1977).

The programmable subtree is shown in Figure 42 on page 208.

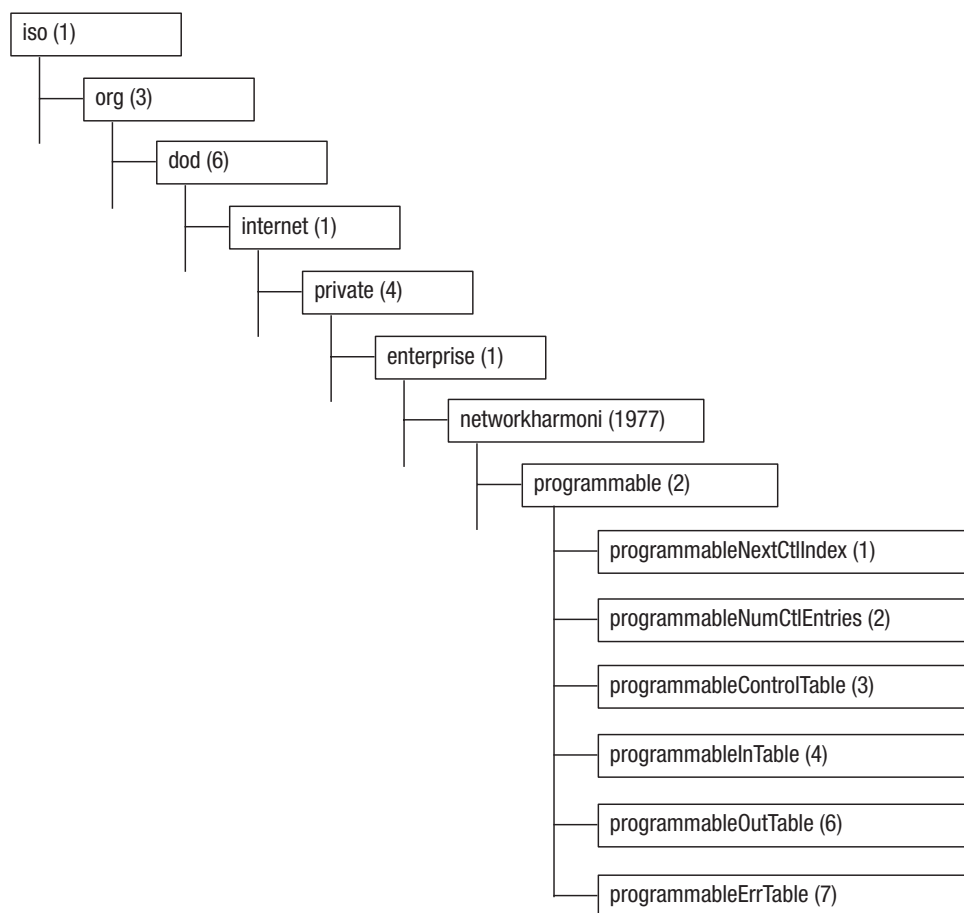


Figure 42. OID tree diagram for programmable

This section provides a summary of the objects defined in the programmable MIB module. For detailed information on all objects in the module, see the `programmable-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB tables

The programmable MIB provides a standard SNMP interface with control row semantics.

The MIB module contains the following tables:

- A control table, `programmableControlTable`, which defines program execution.
- An input table, `programmableInTable`, which stores programs uploaded to Netcool/SSM.
- An output table, `programmableOutTable`, which stores standard output by the programs executed.
- An error table, `programmableErrTable`, which lists error output by programs executed.

## Control instance scalar objects

The programmable MIB includes two scalar objects that provide information about the control table.

Table 117 lists these objects.

*Table 117. Control instance scalar objects*

Object	Description
programmableNextCtrlIndex	Stores the row index of the next available control row. To reserve a control row, read the value of this object and write it back again.
programmableNumCtrlEntries	Indicates the number of rows currently defined in the control table.

## Control table

The programmable control table (programmableControlTable) defines the operation of the program represented by data in the programmableInTable. This includes control information, such as the execution command-line, whether the program is to be passed to virtual machines through standard input and output (stdin/stdout), and event generation/activation.

Table 118 lists the row objects in programmableControlTable.

*Table 118. Programmable control table (programmableControlTable)*

Row object	Description
Application	Specifies the program executed. See “General procedure” on page 199 for more details on using this object.
Community	Sets the community string associated with the program.
DataTableSize	Sets the number of rows in programmableDataTable assigned to this control row.
Description	A description of the control row's purpose.
ErrTableSize	Sets the number of rows in programmableErrTable assigned to this control row.
EventCondition	Specifies the condition under which the event indicated by programmableControlEventIndex is fired: fireOnSuccess(1) - Event fires on successful completion of program execution fireOnFailure(2) - Event fires if program fails to execute fireOnEither(3) - Event fires on either of the above conditions
EventIndex	Identifies the event generated when the program executes. The programmableControlEventCondition object sets the condition under which the event is generated.  If the value of this object does not correspond to an entry in the RMON eventTable or is 0, then no event is generated.

Table 118. Programmable control table (programmableControlTable) (continued)

Row object	Description
EventStatus	<p>Sets the event flow control for the events associated with this row:</p> <p>eventReady(1) - A single event may be generated, after which the value of this object changes to eventFired(2).</p> <p>eventFired(2) - Disables event generation. No events may be generated until the object is modified to eventReady(1) or eventAlwaysReady(3).</p> <p>eventAlwaysReady(3) - Disables the flow control, allowing events to be generated without restriction. Using this setting is not recommended as it can result in high network traffic or other performance problems.</p>
Index	Uniquely identifies the control row.
InFile	Specifies the file that is read into programmableInTable if programmableControlVM has the value vmIFile(2) or vmIOFile(3).
InTableSize	Sets the number of rows in programmableInTable assigned to this control row.
LineOutEventIndex	Index of events fired for each line-out output.
OutFile	Specifies the file to which the contents of programmableOutTable are written if programmableControlVM has the value vmOFile(4) or vmIOFile(3).
OutTableSize	Sets the number of rows in programmableOutTable assigned to this control row.
Owner	Indicates the owner of the control row.
ProgramStatus	<p>Defines the program status:</p> <p>idle(1) - The default program state can be set from failed, completed, aborted.</p> <p>running(2) - The program may be set to running from idle, completed.</p> <p>completed(3) - Is the result of the application finishing irrespective of return code.</p> <p>failed(4) - Indicates that the application failed to run.</p> <p>aborted(5) - May be set from running to terminate the application.</p> <p>The value of this object may be set to idle(1), running(2) or aborted(5).</p>
ReturnCode	The code returned by the program after execution.
Status	The SNMPv2 row status object controlling creation, activation and deletion of the control row.
TurnOffEventIndex	Contains the RMON eventIndex of the event configured to deactivate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or is 0, then deactivation via an event is disabled.
TurnOnEventIndex	Contains the RMON eventIndex of the event configured to activate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or is 0, then activation via an event is disabled.

Table 118. Programmable control table (programmableControlTable) (continued)

Row object	Description
VM	<p>Sets the program execution and input/output modes:</p> <p>vmNorm(1) - Executes the command specified by programmableControlApplication. stdin is taken from programmableInTable. stdout is directed to programmableOutTable. stderr is directed to programmableErrTable.</p> <p>vmIFile(2) - Reads the file indicated by programmableControlInFile into the programmableInTable before executing the command specified by programmableControlApplication and before processing \$INPARAM.</p> <p>vmIOFile(3) - Selects both vmIFile(2) and vmOFile(4) modes.</p> <p>vmOFile(4) - Writes the contents of the programmableOutTable to the file indicated by programmableControlOutFile after executing the command specified by programmableControlApplication and processing \$OUTPARAM.</p> <p>vmConsole(5) - Executes a Netcool/SSM configuration command.</p> <p>vmConsoleIFile(6) - Executes a Netcool/SSM configuration command. Reads the file indicated by programmableControlInFile into the programmableInTable before executing the command.</p> <p>vmConsoleIOFile(7) - Selects both vmConsoleIFile(6) and vmConsoleOFile(8) modes.</p> <p>vmConsoleOFile(8) - Executes a Netcool/SSM configuration command. Writes the contents of the programmableOutTable into the file indicated by programmableOutFile after executing the command.</p> <p>vmIOEvent(9) - Performs both vm-ifile and vm-oevent. Also adds these options to the program vm console command, oevent and ioevent.</p> <p>vmOEvent(10) - All stdout line output generates a LineOutEvent. No output is written to OutTable or ErrTable.</p>

## Input table

The input table (programmableInTable) provides the storage for input to programs. The text stored in rows in this table is passed to the program as stdin.

Table 119 lists the row objects in programmableInTable.

Table 119. Input table (programmableInTable)

Row object	Description
In	The actual text sent to stdin.
Line	The line number of text sent to stdin.

## Out table

The programmableOutTable stores text passed by the program to stdout.

Table 120 lists the row objects in programmableOutTable.

Table 120. Out table (programmableOutTable)

Row object	Description
Line	The line number of text sent to stdout.
Out	The actual text sent to stdout.

## Error table

The programmableErrTable stores text passed by the program to stderr.

Table 121 lists the row objects in programmableErrTable.

Table 121. Error table (programmableErrorTable)

Row object	Description
Err	The actual text sent to stdout.
Line	The line number of text sent to stdout.

## Notification types

The programmable MIB defines notification types for events generated by the subagent.

These types are shown in Figure 54 and listed in Table 168.

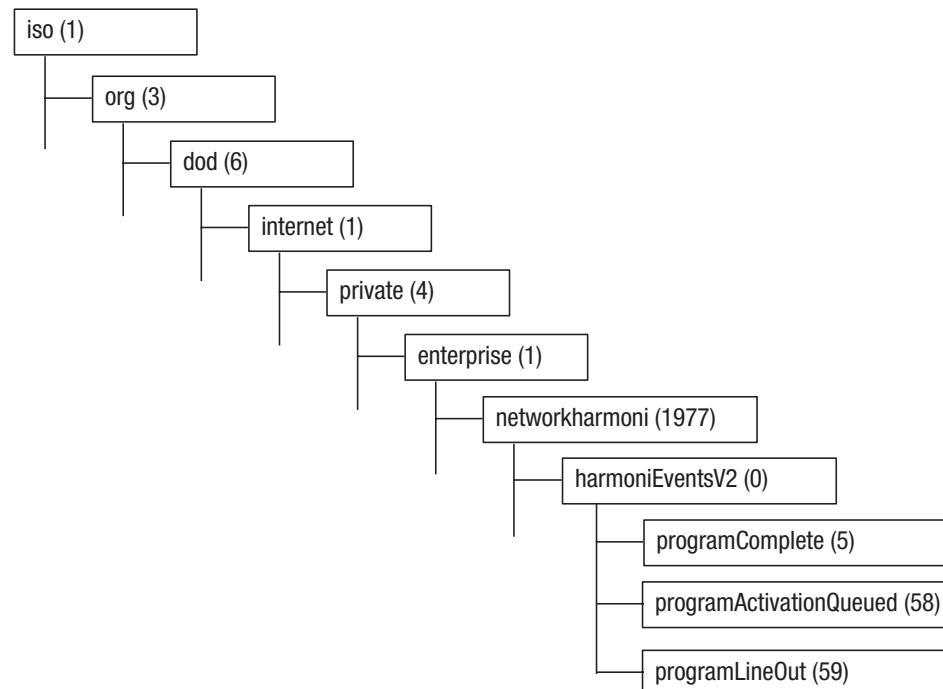


Figure 43. OID tree diagram for programmable notification type

Table 122. Notification types for programmable

Notification type	Description
programComplete	<p>Generated when program execution finishes.</p> <p>Variable bindings:</p> <ul style="list-style-type: none"> <li>programmableControlProgramStatus</li> <li>programmableControlReturnCode</li> <li>programmableControlDescription</li> </ul>
programActivationQueued	<p>Generated when a program is activated multiple times in succession, causing program execution to be queued. This situation occurs when programmableControlTurnOnEventIndex event is fired while the program is currently running.</p> <p>Variable bindings:</p> <ul style="list-style-type: none"> <li>programmableControlTurnOnEventIndex</li> <li>programmableControlDescription</li> </ul>
programLineOut	<p>Generated for each line output from the program during execution</p> <p>This situation occurs only if the programmableControlLineOutEventIndex object is non-zero, and the corresponding eventType in the eventTable is set to either snmp-trap(3) or log-and-trap(4).</p>





---

## Chapter 20. Scratchpad

The scratch subagent and its MIB module provide facilities for storing user defined data.

---

### Component files

The scratch subagent and MIB module are comprised of a set of component files.

Table 123 lists the scratch subagent and MIB module component files and their installed locations.

*Table 123. Scratch component files*

File	Location	Description
scratch.dll (Windows) libscratch.sl/.so (UNIX)	bin	Binary implementation of the scratch subagent.
scratch-mib.mib	mibs	MIB definition document.
scratch.oid	config/oid	scratch subagent object identifier file.

---

### Guidelines

Use the scratch subagent and MIB module as a 'scratchpad' for storing and retrieving data. When you allocate storage in the scratchpad, you can write to and read from it as required using either the Netcool/SSM command console or an external monitoring application. Then, using the arithmetic or genalarm subagents, you can perform threshold monitoring and event generation on data in the scratchpad.

To load the subagent, use the command:  
subagent load scratch

### Allocating storage

To obtain storage space in the scratchpad, perform an SNMP GET operation on the scratchAllocate object.

#### About this task

The value returned by the SNMP GET is the OID of currently unused storage space in the scratchpad. Scratchpad OIDs are allocated beneath the OID scratchPad.2147483647.

**Note:** Scratchpad storage remains allocated until is it erased. Always delete allocated storage when you no longer require it.

#### Allocating storage on the scratchpad

The following configuration commands allocate scratchpad storage and print the OID of the allocated storage to the console:

```
snmp get $scratchAllocate.0
scr=$?
echo $scr
```

## Storing values

To store a value in the scratchpad, perform an SNMP SET operation on an object identifier under the scratchpad branch.

### Storing values in the scratchpad

The following configuration commands allocate scratchpad storage then store a string value in the allocated storage:

```
snmp get $scratchAllocate.0
scr=$?
snmp set $scr s "My scratchpad storage"
```

## Retrieving values

To retrieve a value stored in the scratchpad, perform an SNMP GET operation on the scratchpad storage.

### About this task

**Tip:** Always examine the returned value to ensure that is of the expected type, and has not been accidentally overwritten by a value of a different type.

### Retrieving values from the scratchpad

```
snmp get $scratchAllocate.0
scr=$?
snmp set $scr i "My scratchpad storage"
snmp get $scr
```

## Deleting values

To delete values from the scratchpad, perform an SNMP SET operation on the scratchErase object, setting it to the OID of the scratchpad branch that contains the values you wish to delete.

### About this task

The SNMP SET operation deletes all values in the scratchpad OID tree commencing from the specified OID.

### Deleting values from the scratchpad

The following console session demonstrates how to delete values from the scratchpad. The first erase operation deletes the value branch 1, item 3 (the single value stored at scratchPad.1.3), the second operation deletes values in the OID subtree from branch 1 downwards (the OID subtree commencing at scratchPad.1), and the final erase operation deletes all remaining values from the scratchpad:

```
Agent> snmp walk $scratchPad
.1.3.6.1.4.1.1977.50.1.1 s "branch 1"
.1.3.6.1.4.1.1977.50.1.1.1 s "branch 1, item 1"
.1.3.6.1.4.1.1977.50.1.1.2 s "branch 1, item 2"
.1.3.6.1.4.1.1977.50.1.1.3 s "branch 1, item 3"
.1.3.6.1.4.1.1977.50.1.2 s "branch 2"
.1.3.6.1.4.1.1977.50.1.2.1 s "branch 2, item 1"
.1.3.6.1.4.1.1977.50.1.2.2 s "branch 2, item 2"
Agent> snmp set $scratchErase.0 o $scratchPad.1.3
```

```

Agent> snmp walk $scratchPad
.1.3.6.1.4.1.1977.50.1.1 s "branch 1"
.1.3.6.1.4.1.1977.50.1.1.1 s "branch 1, item 1"
.1.3.6.1.4.1.1977.50.1.1.2 s "branch 1, item 2"
.1.3.6.1.4.1.1977.50.1.2 s "branch 2"
.1.3.6.1.4.1.1977.50.1.2.1 s "branch 2, item 1"
.1.3.6.1.4.1.1977.50.1.2.2 s "branch 2, item 2"
Agent> snmp set $scratchErase.0 o $scratchPad.1
Agent> snmp walk $scratchPad
.1.3.6.1.4.1.1977.50.1.2 s "branch 2"
.1.3.6.1.4.1.1977.50.1.2.1 s "branch 2, item 1"
.1.3.6.1.4.1.1977.50.1.2.2 s "branch 2, item 2"
Agent> snmp set $scratchErase.0 o $scratchPad
Agent> snmp walk $scratchPad

```

## MIB module

The scratch MIB is a subtree of networkharmoni (1977).

The scratch subtree is shown in Figure 44.

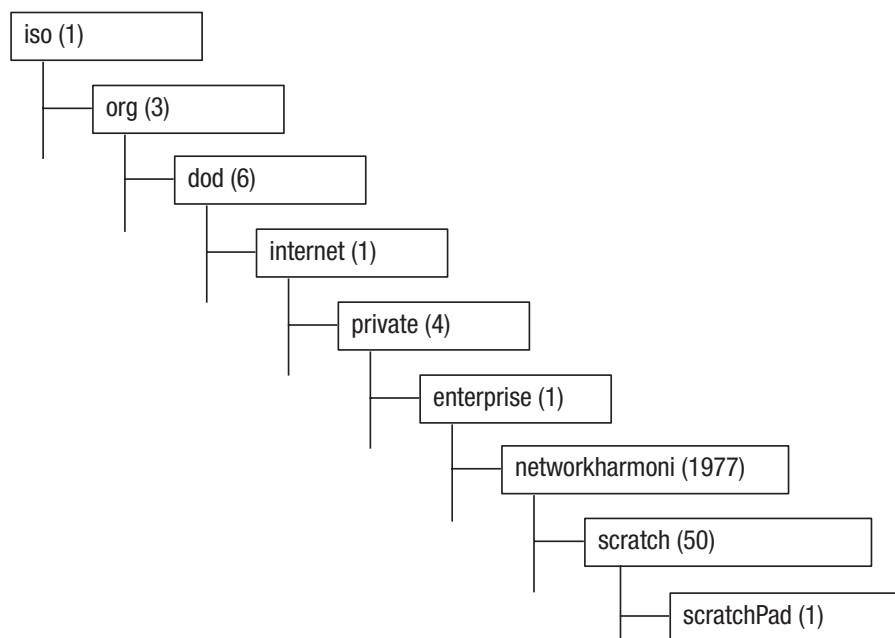


Figure 44. OID tree diagram of the scratch MIB module

This section provides a summary of the objects defined in the scratch MIB module. For detailed information on all objects in the module, see the `scratch-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB objects

The scratch MIB module contains the allocation and erasure objects, as well as the base scratchpad OID.

- `scratchErase`

Controls deletion of storage in the scratchpad. Setting this object to an OID located in the OID tree below `scratchPad` removes all objects stored in the scratchpad below that OID. Setting this object to an OID located outside the `scratchPad` branch results in a `badValue` error.

- `scratchAllocate`

Manages storage allocation in the scratchpad. Performing an SNMP GET operation on this object returns a OID of storage in the scratchpad that is currently unused. OIDs are allocated beneath the branch `scratchPad.2147483647`. Performing an SNMP GETNEXT operation on this object returns the value `0.0`.

The scratch MIB module also defines the `scratchPad` OID, which is the base OID for all scratchpad storage. The OID of any storage allocated in the scratchpad is located below this OID.

---

## Chapter 21. Service level agreement

The sla subagent provides metrics for application performance (response time and availability). The MIB allows threshold values to be set (dynamically) for application performance metrics and can generate an SNMP trap. The SLA MIB measures various metrics about client, server and network response time and can be configured to monitor individual applications, clients or servers.

This type of information is used to monitor compliance to service level agreements for application performance to/from a host.

---

### Component files

The sla subagent and the associated sla MIB module are comprised of a set of component files.

Table 124 lists the sla subagent and MIB module component files and their locations.

*Table 124. sla component files*

File	Location	Description
sla.dll (Windows) libsla.so/.sl (UNIX)	bin	Binary implementation of the sla subagent.
sla-mib.mib	mibs	sla MIB definition document.
sla.oid	config/oid	sla subagent object identifier file.

---

### Guidelines

To use the subagent for monitoring service levels, specify client and server IP addresses of interest, and the ports on which the applications of interests are running. You can also configure monitoring to be activated and deactivated in response to events.

To load the subagent, use the command:

```
subagent load sla
```

### Client/server filtering

The slaControlClientAddr and slaControlServerAddr objects set the monitored client and server addresses.

By default, the client address is initialized to the localhost on which the sla subagent is running, while the server address is initialized to \*, null, or 0.0.0.0, which matches all hosts so that data will be collected for all servers.

To filter out information about connections from other network hosts that are not of interest, set the client and server addresses to specific IP addresses. Using specific addresses filters out unrelated requests and responses from other hosts on the same network.

Apart from the uninitialized server address (0.0.0.0), network addresses that encompass a range of host addresses are not supported. The client address cannot be set to 0.0.0.0. It must contain a specific IP address.

## Protocol filtering

You can configure control rows to monitor a specific protocol or a protocol range.

To do so, set the `slaControlMinPort` and `slaControlMaxPort` objects to port numbers that correspond to the protocol range. The default values of these objects are 1 and 65535 respectively, which represent a range for monitoring all protocols. If you wish to monitor only a single protocol, set both objects to the same port value.

## Event-controlled activation and deactivation

Control rows can be activated and deactivated by events.

Set the `slaControlTurnOnEventIndex` and `slaControlTurnOffEventIndex` objects to the indexes of the RMON events that will activate or deactivate the control row.

## Event generation

The subagent can generate events in reaction to poor response times.

The `slaControlSLAValue` object sets the response time threshold. If the observed response time exceeds the value of this object and the control row is configured with a valid event index, an event is generated.

The `slaControlEventIndex` object identifies the event generated and the `slaControlEventStatus` object provides event flow control.

## SNMPv1 compatibility

The `sla` MIB module uses the Counter64 data type, so it is not compatible with SNMPv1. Only use the `sla` subagent with SNMPv2 and later.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The sla commands create rows in the control table (slaControlTable).

The general syntax of these commands is:

```
sla property=value  
sla create [property=value ...]  
sla reset
```

Table 125 lists the properties supported in these commands.

Table 125. Configuration command parameters - sla

Property	Type	Description	Sets MIB object
buckets	int	The number of summary table rows requested.	BucketsRequested
client	IP	The IP address of the monitored client.	ClientAddr
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl
datasource	OID	The interface monitored by the control row.	DataSource
description	string	A description of the control row.	Description
event	int	The index of the event generated if an SLA violation occurs.	EventIndex
eventstatus	enum	Event flow control:  alwaysready  fired  ready	EventStatus
history	int	The history table size limit.	HistInterval
maxport	int	The maximum port of the monitored protocol.	MaxPort
minport	int	The minimum port of the monitored protocol.	MinPort
offevent	int	The control row deactivation event index.	TurnOffEventIndex
onevent	int	The control row activation event index.	TurnOnEventIndex
pdesc	string	A description of the SLA protocol associated with the monitored port range.	ProtocolDesc
server	IP	The IP address of the monitored server.	ServerAddr
type	enum	Selects the type of response time monitored:  application  network	SLAType
value	int	Sets the response time (in milliseconds) against which the SLA is monitored.	SLAValue

---

## Examples

These examples demonstrate how to use the sla MIB to monitor the quality of a service.

### Monitoring response times

Monitor all HTTP response times originating from the localhost on port 80:

```
subagent load sla
sla reset
sla server=0.0.0.0
sla minport=80
sla maxport=80
sla type=application
sla value=1000
sla datacontrol=on
sla buckets=10
sla create
```

### Monitoring POP3

Monitor the Post Office Protocol (POP3) on port 110 and generate an event if the response time is greater than 12 seconds. Send the trap associated with this event to the management station at address 123.23.123.23.

To set up the SLA control row and the associated trap destination and event, use the following configuration commands:

```
trapdest add 123.23.123.23:162 public "Example"
event reset
event description="POP3 SLA event"
event type=log-and-trap
event community=public
event create
pop3event=$?
```

```
subagent load sla
sla reset
sla server=0.0.0.0
sla minport=110
sla maxport=110
sla type=application
sla value=12000
sla event=$pop3event
sla eventstatus=alwaysready
sla datacontrol=on
sla buckets=288
sla history=300
sla description="POP3 trap"
sla create
```

The sla subagent records an SLA violation when the transaction time exceeds 12 seconds for any POP3 mail. If a transaction is not concluded in 16 seconds then that transaction is ignored by the subagent. A history table records a rolling history of up to 288 buckets. A trap is also logged and sent to the NOC community group for each violation.



---

## MIB module

The sla MIB is a subtree of networkharmoni (1977).

The sla subtree is shown in Figure 45.

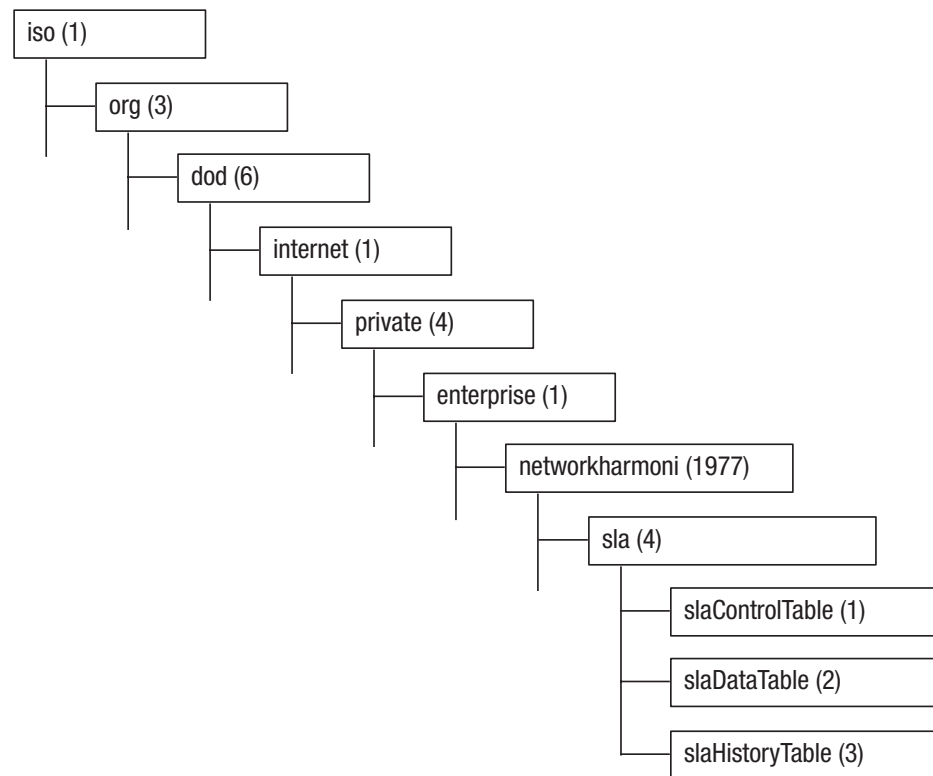


Figure 45. OID tree diagram of the sla MIB module

This section provides a summary of the objects defined in the sla MIB module. For detailed information on all objects in the module, see the sla-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB tables

The sla MIB provides a standard SNMP interface with control row semantics.

The MIB module contains the following tables:

- A control table, slaControlTable
- A data table, slaDataTable
- A history table, slaHistoryTable

## Control table

The SLA control table (slaControlTable) defines monitors for service level agreement response times. Each control row specifies the interface to be monitored, the type of response times and the client and server to be monitored. The results of each monitor are stored in slaDataTable.

Table 126 lists the row objects in slaControlTable.

Table 126. Service level agreement control table (slaControlTable)

Row object	Description
BucketsGranted	Indicates the number of rows in slaHistoryTable granted by the subagent to this control row.
BucketsRequested	Sets the number of rows in slaHistoryTable requested for storing the data associated with this control row.
ClientAddr	Specifies the IP address of the monitored client.
CreateTime	Stores the value of sysUpTime when this entry was last activated.
DataControl	Sets the monitoring status of the control row:  on(1) - Enables data collection  off(2) - Suspends data collection
DataSource	Identifies the source of the data (that is, the interface) monitored by the control row. This source can be any interface on the device hosting the subagent. The value of this object identifies the interface by its ifIndex object. For example, for a control row to monitor data using interface #1, this object must be set to ifIndex.1.
Description	A description of the SLA requests and responses or transactions monitored.
EventIndex	Contains the RMON eventIndex of the event generated if an SLA violation occurs. If this index does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then no event is generated.
EventStatus	Sets the event flow control for the events associated with this row:  eventReady(1) - A single event may be generated, after which the value of this object changes to eventFired(2).  eventFired(2) - Disables event generation. No events may be generated until the object is modified to eventReady(1) or eventAlwaysReady(3).  eventAlwaysReady(3) - Disables the flow control, allowing events to be generated without restriction. Using this setting is not recommended as it can result in high network traffic or other performance problems.
HistInterval	Sets the sampling interval (in seconds). Each time this interval elapses, rows in slaDataTable are moved to slaHistoryTable. If the value of this object is 0, rows in slaDataTable are kept indefinitely.
Index	Uniquely identifies the control row.
MaxPort	Specifies the maximum port of the monitored protocol.
MaxResponseTime	Sets the maximum response time processed by the subagent.
MinPort	Specifies the minimum port of the monitored protocol.

Table 126. Service level agreement control table (slaControlTable) (continued)

Row object	Description
MinResponseTime	Sets the minimum response time processed by the subagent.
Owner	Indicates the owner of the control row.
ProtocolDesc	A description of the SLA protocol associated with the monitored port range.
ServerAddr	Specifies the IP address of the monitored server.
SLAType	Selects the monitored response time:  networkLayer(1) - Monitors network layer response time  applicationLayer(2) - Monitors application layer response time. Some applications and protocols do not support network layer response time monitoring.
SLAValue	Sets the SLA monitored response time (in milliseconds). The control row evaluates response times using this value.
Status	An SNMPv2 row status object controlling creation, activation and deletion of the control row.
TableSize	Sets the number of entries in slaDataTable associated with the control row.
TurnOffEventIndex	Contains the RMON eventIndex of the event configured to deactivate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then deactivation via an event is disabled.
TurnOnEventIndex	Contains the RMON eventIndex of the event configured to activate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then activation via an event is disabled.

## Data table

The SLA data table (slaDataTable) stores statistics generated by each monitor defined in the control table. Each slaDataEntry maintains the last, average, best and worst response times between the client and server measured by one control row.

Data may be sampled from this table and stored in the history table at an interval set by the corresponding control row object slaControlHistInterval.

Table 127 on page 226 lists the row objects in slaDataTable. The names of all objects in the data table have the prefix slaData.

## History table

The SLA history table (slaHistoryTable) contains statistics sampled from slaDataTable. The interval at which statistics are sampled from the data table and the number of history table rows maintained is defined by the corresponding control row.

The row objects in slaHistoryTable are identical to those in slaDataTable and are indicated in Table 127 on page 226.

Table 127. Service level agreement data/history table objects

Row object	Description
AvgClientTime	The average time between receipt of a data packet and transmission of the acknowledgement by a client, determined for a particular client-server pair.
AvgNetworkTime	<p>The average value of Network Time for a particular client-server pair, where the Network Time is measured as:</p> $\text{Network Time} = \text{Transaction Time} - \text{Server Time} - \text{Client Time}$
AvgResponseTime	The average response time (in milliseconds) of the observed request/response pair. A value of -1 indicates that a request/response pair was not observed or has expired.
AvgServerTime	The average time between a server's receipt of the initial request and transmission of the first response, determined for a particular client-server pair.
AvgTransactionTime	The average time between transmission of the request and transmission of the acknowledgement of the response's last packet, determined from all last request/response pairs.
BestClientTime	The least time measured between receipt of a data packet and transmission of the acknowledgement by a client, determined for a particular client-server pair.
BestNetworkTime	<p>The lowest Network Time measured for a particular client-server pair, where the Network Time is measured as:</p> $\text{Network Time} = \text{Transaction Time} - \text{Server Time} - \text{Client Time}$
BestResponseTime	The best response time (in milliseconds) of the observed request/response pair. A value of -1 indicates that a request/response pair was not observed or has expired.
BestServerTime	<p>The least time measured between a server's receipt of the initial request and transmission of the first response, determined for a particular client-server pair. This value is measured as:</p> $\text{Server Time} = \text{Application Response Time} - \text{Network Response Time}$
BestTransactionTime	The time measured between transmission of the request and transmission of the acknowledgement of the response's last packet, determined from the request/response pair with the best response time.
ClientAddr	The IP address of the client.
InOctets	The number of octets observed from server to client.
InPkts	The number of packets observed from server to client.
LastClientTime	The most recently measured time between receipt of a data packet and transmission of the acknowledgement by a client, determined for a particular client-server pair.

Table 127. Service level agreement data/history table objects (continued)

Row object	Description
LastNetworkTime	The most recently measured Network Time for a particular client-server pair, where the Network Time is measured as:  Network Time = Transaction Time - Server Time - Client Time
LastResponseTime	The response time (in milliseconds) of the last observed request/response pair. A value of -1 indicates that a request/response pair was not observed or has expired.
LastServerTime	The most recently measured time between a server's receipt of the initial request and transmission of the first response, determined for a particular client-server pair. This value is measured as:  Server Time = Application Response Time - Network Response Time
LastTransactionTime	The time between transmission of the request and transmission of the acknowledgement of the response's last packet, determined from the last request/response pair observed.
NumberOfTransactions	The number of transactions for the current data row since its instantiation.
OutOctets	The number of octets observed from client to server.
OutPkts	The number of packets observed from client to server.
Responses	The number of responses with a response time less than (8×slaControlSLAValue) received as a reply to an outgoing request.
ResponsesFailed	The number of requests that did not receive a response within (8×slaControlSLAValue).
ResponseTime1SLAto2SLA	The number of responses received in which:  $slaControlSLAValue < response\ time < (2 \times slaControlSLAValue)$ .
ResponseTime2SLAto4SLA	The number of responses received in which:  $(2 \times slaControlSLAValue) < response\ time < (4 \times slaControlSLAValue)$
ResponseTime4SLAto8SLA	The number of responses received in which:  $(4 \times slaControlSLAValue) < response\ time < (8 \times slaControlSLAValue)$
ResponseTimeGreaterThan8SLA	The number of responses received in which:  $(8 \times slaControlSLAValue) < response\ time$
ResponseTimeLessThan1SLA	The number of responses received within the SLA time frame specified by slaControlSLAValue.
Retransmissions	The number of client retransmission requests.
ServerAddr	The IP address of the server.

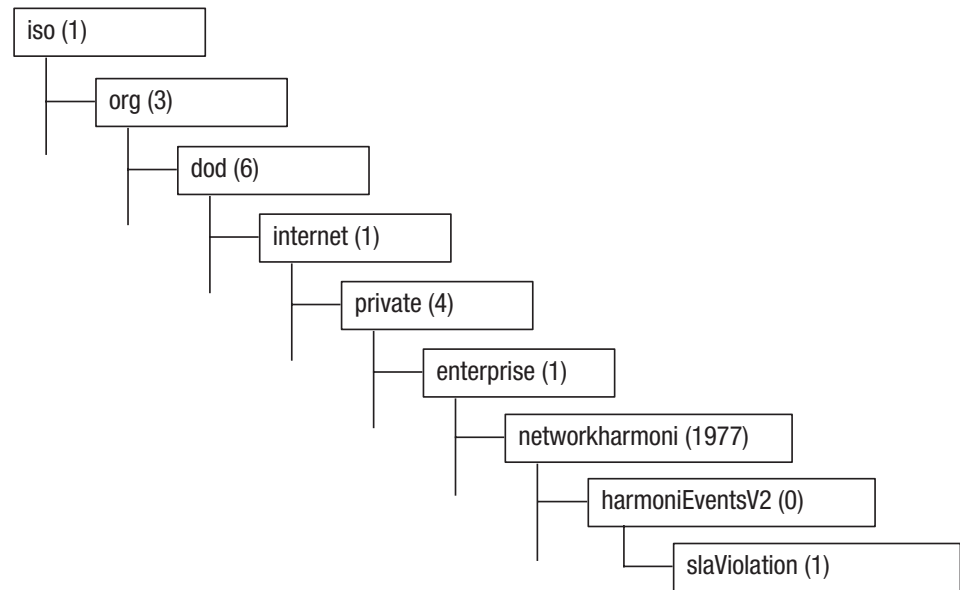
Table 127. Service level agreement data/history table objects (continued)

Row object	Description
SLAVPercentage	The number of response times that were greater than the limit defined by slaControlSLAValue, expressed as a percentage value of all response times measured.
SumOfClientTimes	The sum of all the client times for the current data row since its instantiation.
SumOfNetworkTimes	The sum of all the network times for the current data row since its instantiation.
SumOfResponseTimes	The sum of all the response times for the current data row since its instantiation.
SumOfServerTimes	The sum of all the server times for the current data row since its instantiation.
SumOfTransactionTimes	The sum of all the transaction times for the current data row since its instantiation.
WorstClientTime	The most time measured between receipt of a data packet and transmission of the acknowledgement by a client, determined for a particular client-server pair.
WorstNetworkTime	The highest Network Time measured for a particular client-server pair, where the Network Time is measured as:  Network Time = Transaction Time - Server Time - Client Time
WorstResponseTime	The worst response time (in milliseconds) of observed request/response pair. A value of -1 indicates that a request/response pair was not observed or has expired.
WorstServerTime	The most time measured between a server's receipt of the initial request and transmission of the first response, determined for a particular client-server pair. This value is measured as:  Server Time = Application Response Time - Network Response Time
WorstTransactionTime	The time measured between transmission of the request and transmission of the acknowledgement of the response's last packet, determined from the request/response pair with the worst response time.

## Notification types

The sla MIB implements the slaViolation notification type for events generated by the subagent when an SLA threshold violation occurs.

This notification type contains variable bindings indicating the server and protocol monitored. Figure 46 on page 229 shows the OID tree diagram for this notification type.



*Figure 46. OID tree diagram for slaViolation notification type*

The `slaViolation` notification type contains the following variable bindings:

- `slaDataSLAVPercentage`
- `slaDataServerAddr`
- `slaControlDescription`
- `slaControlProtocolDesc`
- `slaControlSLAType`





---

## Chapter 22. Solaris kstat facility

The kstat subagent and MIB module provide facilities for gathering kernel statistics from host machines running the Solaris operating system.

The kstat subagent uses the Solaris *kstat* infrastructure to access performance data on the host Solaris machine.

---

### Component files

The kstat and MIB module are comprised of a set of component files.

Table 128 lists the kstat component files and their installed locations.

*Table 128. kstat component files*

File	Location	Description
libsolariskstat.so	bin	Binary implementation of the solariskstat subagent.
kstat-mib.mib	mibs	kstat MIB definition document.
solariskstat.oid	config/oid	Object identifier definition file.

**Note:** The solariskstat subagent and kstat MIB module are only available on Solaris platforms.

---

### Guidelines

Use the solariskstat subagent and MIB module to gather the performance statistics required for your monitoring application.

To load the subagent, use the following configuration command:

```
subagent load solariskstat
```

### General monitoring procedure

To monitor kernel statistics using the solariskstat subagent, identify the appropriate statistics, then create a control row to extract those statistics.

#### Identifying kernel statistics

Information about the available kernel statistics is available from the Solaris shell.

#### About this task

To obtain a list of kernel statistics, run the following command from a Solaris shell:

```
/usr/bin/kstat
```

This command displays a complete list of all available kernel statistics. You can refine the selection of statistics displayed, for example, to obtain a list of the available kernel statistics for CPUs, use the following command:

```
/usr/bin/kstat -p -m cpu_stat
```

The command displays all CPU kernel statistics, one statistic per line, in the parseable format:

```
module:instance:name:statistic value
```

For more detailed information about the kstat utility, use the Solaris command `man kstat`.

## Creating control rows

Control table rows identify the monitored kernel statistics.

### Procedure

To create and configure a control row to monitor a kernel statistic:

1. Set `kstatControlStatus` to `createAndWait(5)`.
2. Set the `kstatControlModule`, `kstatControlInstance`, and `kstatControlName` objects to the *module*, *instance* and *name* of the kernel statistic you wish to monitor.  
  
**Tip:** To specify a name that matches any instance number, use the format *name%i*. The `%i` operator automatically appends the instance number to the name. If `kstatControlInstance` has the value `-1`, the `%i` operator matches all instance numbers, and the control row collects statistics for all available instances.
3. Set `kstatSampleInterval` to the desired sample interval (in seconds).
4. Set `kstatSampleMode`:
  - a. To monitor the actual value at each sample, select `immediate(1)`.
  - b. To monitor the change in the statistic's value with each successive sample, select `delta(2)`.
5. For each kernel statistic that you wish to monitor, create a row in the `kstatFieldTable` and set the `kstatFieldName` object to the name of the statistic. See "Specifying kernel statistic field names" for more information about the options available with the `kstatFieldName` object.
6. Activate the control row by setting `kstatControlStatus` to `active(1)`.  
The value of each selected kernel statistic is stored in the data table, one statistic per row, and is updated at the interval set by `kstatSampleInterval`.

### Specifying kernel statistic field names:

The `kstatFieldName` object stores the name of the kernel statistic you wish to monitor.

### Procedure

The value you enter in this object depends on the type of kernel statistic monitored, which is indicated by the `kstatControlType` object:

- When the value of `kstatControlType` is `named(1)`, set the `kstatFieldName` object to name of the kernel statistic. For example, to monitor the kernel statistic `nfs:0:nfs_client:badcalls`, set `kstatFieldName` to `badcalls`.
- When the value of `kstatControlType` is `io(3)`, set the `kstatFieldName` object to either:

- The name of a field in the `kstat_io_t` structure. The following table lists the `kstat_io_t` field names. Definitions of these field names, together with detailed descriptions, are located in the file `/usr/include/sys/kstat.h` on the host machine.

Table 129. *kstat\_io\_t* field names

Field name	Description
<code>nread</code>	The number of bytes read.
<code>nwritten</code>	The number of bytes written.
<code>rcnt</code>	The number of elements in the run state.
<code>reads</code>	The number of read operations.
<code>rlastupdate</code>	The time at which the run queue last changed (in nanoseconds).
<code>rlentime</code>	The product of the cumulative run length and time.
<code>rtime</code>	The cumulative run time.
<code>wcnt</code>	The number of elements in the wait state.
<code>wlastupdate</code>	The time at which the wait queue last changed (in nanoseconds).
<code>wlentime</code>	The product of the cumulative wait length and time.
<code>writes</code>	The number of write operations.
<code>wtime</code>	The cumulative wait time (in nanoseconds).

- The name of a built-in I/O formula. The following table lists the built-in I/O formulas that the `kstat` subagent provides.

Table 130. *I/O formulas*

Formula name	Description
<code>AVERAGEQUEUELEN</code>	Average length of the queue of operations waiting to run, calculated as: $(rlentime + wlentime) / (rlastupdate + wlastupdate)$
<code>AVERAGERUNTIME</code>	Average time for an operation to complete after it has been dequeued from the wait queue, calculated as: $rtime / (reads + writes)$
<code>AVERAGESERVICETIME</code>	Average time to queue and complete an I/O operation (in nanoseconds), calculated as: $(rtime + wtime) / (reads + writes)$
<code>AVERAGEWAITTIME</code>	Average time for which operations are queued before they are run (in nanoseconds), calculated as: $wtime / (reads + writes)$
<code>UTILIZATION</code>	Average device utilization, measured as the percentage of time for which the device was busy servicing I/O requests during the sample interval. $rlentime \times 100 / rlastupdate$
The named values used in each formula are <code>kstat_io_t</code> field names. See Table 130 for definitions of the values.	

## Threshold monitoring and event generation

The solariskstat subagent does not generate events, however you can use the genalarm subagent to monitor thresholds on kstat MIB objects and generate events in response to threshold violations.

### About this task

See “Examples” on page 235 for an implementation of a simple threshold monitoring task.

## SNMPv1 compatibility

The kstat MIB module uses the Counter64 data type, so it is not compatible with SNMPv1. You should only use the solariskstat subagent with SNMPv2 and later.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The kstat commands create rows in the control table (kstatControlTable).

The general syntax of these commands is:

```
kstat property=value
kstat create [property=value ...]
kstat reset
```

Table 131 lists the properties supported in these commands.

Table 131. Configuration command parameters - kstat

Property	Type	Description	Sets MIB object
buckets	int	The number of rows in the data table allocated to store sampled values of the kernel statistic.	Buckets
instance	string	The instance ID of the kernel statistic to monitor. The value -1 is a wildcard to monitor all available instances.	Instance
module	string	The module name of the kernel statistic to monitor.	Module
name	string	The name of the kernel statistic to monitor.	Name
sampleinterval	int	The interval (in seconds) on which to sample the kernel statistic.	SampleInterval

Table 131. Configuration command parameters - kstat (continued)

Property	Type	Description	Sets MIB object
samplemode	enum	The sampling mode for the kernel statistic:  delta - The difference between two successive kernel statistic values.  immediate - The raw value of the kernel statistic.	Real0operand

## Field table

The kstatfield commands create rows in the field table (kstatFieldTable) and assign them to the next control row created using the kstat create command.

The general syntax of these commands is:

```
kstatfield property=value
kstatfield store [property=value ...]
kstatfield reset
```

Table 132 lists the properties supported in these commands.

Table 132. Configuration command parameters - kstatfield

Property	Type	Description	Sets MIB object
name	string	The kernel statistic field name.	Name

## Examples

These examples demonstrate how to use the kstat subagent to perform simple monitoring tasks.

### Monitoring a named kernel statistic

Monitor the number of bad client NFS calls using the named kernel statistic nfs:0:nfs\_client:badcalls, whose value is obtained using the following Solaris kstat command:

```
kstat -m nfs -i 0 -n nfs_client -s badcalls
```

To implement this task, use the following configuration commands:

```
kstat reset
kstatfield reset
kstat module=nfs
kstat instance=0
kstat name=nfs_client
kstatfield store name=badcalls
kstat create
```

### Monitoring multiple instances

Monitor all instances of the NFS reads I/O kernel statistic whose values are obtained using the following Solaris kstat command:

```
kstat -m nfs -s reads
```

To implement this task, use the following configuration commands:

```

kstat reset
kstatfield reset
kstat module=nfs
kstat instance=-1
kstat name=nfs%i
kstatfield store name=reads
kstat create

```

## Threshold monitoring for disk utilization

Monitor of utilization of all SCSI disks at five-second intervals using the built-in UTILIZATION I/O formula, and generate a notification if any value exceeds 98%, with 10% hysteresis on the alarm threshold.

Disk kernel statistics are available using the Solaris kstat command:

```
kstat -m sd
```

To implement this task, use the following configuration commands:

```

subagent load genalarm
subagent load rmonc
subagent load solariskstat

```

```

uThreshold=98
lThreshold=88

```

```

kstat reset
kstatfield reset
kstat module=sd
kstat instance=-1
kstat name=sd%i
kstat sampleinterval=5
kstat samplemode=immediate
kstat buckets=1
kstatfield store name=UTILIZATION
kstat create
ctrlIdx=$?

```

```

event reset
event type=snmp-trap
event community=public
event description="SCSI disk utilization alarm"
event create
violationevent=$?

```

```

event description="SCSI disk utilization normal"
event create
normalevent=$?

```

```

genalarm reset
genalarm var="$kstatDataValue64u.$ctrlIdx.*"
genalarm vardescr="disk utilization"
genalarm interval=5
genalarm type=absolute
genalarm mode=hysteresis
genalarm risethresh=$uThreshold
genalarm fallthresh=$lThreshold
genalarm riseduration=0
genalarm fallduration=0
genalarm riseevent=$violationevent
genalarm fallevent=$normalevent
genalarm risedescr="SCSI disk utilization at $$6."
genalarm falldescr="SCSI disk utilization returned to acceptable (l.t. $$7)."
genalarm create

```

## MIB module

The kstat MIB is a subtree of sysResources(11).

The kstat MIB subtree is shown in Figure 47.

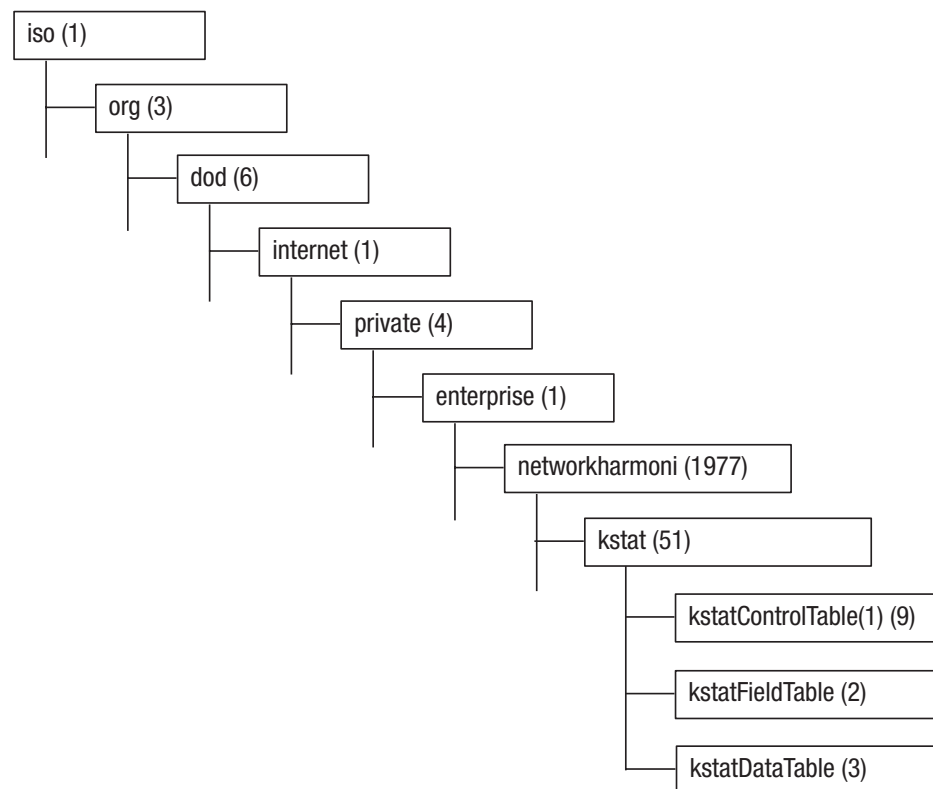


Figure 47. OID tree diagram for kstat MIB module

For detailed information on all the objects provided in this MIB see the MIB document in the mibs subdirectory of the Netcool/SSM installation.

## Control table

The control table (kstatControlTable) contains configuration information specifying kstat statistics to be monitored.

Table 133 lists the objects in kstatControlTable.

Table 133. Control table (kstatControlTable)

Row object	Description
Buckets	Specifies the number of data rows (samples) to retain for each statistic before the oldest sample is removed from the data table.
Class	The kstat class to which the kernel statistic belongs. This field is initialized when the control row is activated.
Index	Uniquely identifies the kstatControlTable row.
Instance	Specifies the instance ID of the kernel statistic to monitor. The value -1 is a wildcard to monitor all available instances.
Module	Specifies the module name of the kernel statistic to monitor.

Table 133. Control table (kstatControlTable) (continued)

Row object	Description
Name	Specifies the name of the kernel statistic to monitor. You can use the format <i>name%i</i> to automatically insert the current instance, or, if kstatControlInstance has the value -1, all available instances.
Owner	Indicates the owner/creator of the row.
SampleInterval	Specifies the interval (in seconds) on which to sample the kernel statistic.
SampleMode	Specifies how to sample the statistics:  immediate(1) - Reports raw data values.  delta(2) - Reports the difference between successive samples.
Status	SNMPv2 row status. Controls creation, activation and deletion of the control row.
Type	The internal type of the kernel statistic being sampled. This value is for information only and is initialized when the control row is activated.  raw(0) - Not supported (no data is collected).  named(1) - The kernel statistic is identified by name.  intr(2) - Not supported (no data is collected).  io(3) - The kernel statistic is the name of a field in the kstat_io_t structure, or a built-in I/O formula.  timer(4) - Not supported (no data is collected).  unknown(999)

## Field table

The field table (kstatFieldTable) contains a list of statistical fields (independent values) to monitor.

Table 134 lists the objects in kstatFieldTable. Each row in this table is indexed by a combination of kstatControlIndex and kstatFieldIndex objects.

Table 134. Field table (kstatFieldTable)

Row object	Description
Index	Uniquely identifies this field row with respect to the corresponding control row.
Name	The well-known name of the monitored field. For named kernel statistics, this is the textual name of the statistic. See “Identifying kernel statistics” on page 231 for more information about this object.



Table 134. Field table (kstatFieldTable) (continued)

Row object	Description
Type	<p>The native data type for the field. This object is initialized when the control row is activated. The data types are:</p> <p>char(0) - Character (<i>this type is not supported</i>).</p> <p>int32(1) - 32-bit signed integer.</p> <p>uint32(2) - 32-bit unsigned integer.</p> <p>int64(3) - 64-bit signed integer (<i>this type is not supported</i>).</p> <p>uint64(4) - 64-bit unsigned integer.</p> <p>unknown(999) - Indicates that the kernel statistic was not found.</p>

## Data table

The data table (kstatDataTable) stores statistics that have been successfully collected.

Table 135 lists the objects in kstatDataTable. Each row in this table is indexed by a combination of kstatControlIndex, kstatFieldIndex, kstatDataInstance, and kstatDataBucket objects.

Table 135. Data table (kstatDataTable)

Row object	Description
Bucket	The bucket index of this sample. Buckets are numbered 1 to kstatControlBuckets in increasing order of age. The newest sample is always bucket number 1.
Instance	The kstat instance ID of the statistic.
Value32	The value of the statistic as a signed 32-bit integer.
Value32u	The value of the statistic as an unsigned 32-bit integer.
Value64u	The value of the statistic as an unsigned 64-bit integer.



---

## Chapter 23. Recently installed software

The srsoftware subagent and the associated srSoftware MIB module provide summary information about software packages installed on Windows hosts.

---

### Component files

The srsoftware subagent and MIB module are comprised of a set of component files.

Table 136 lists the srsoftware subagent and MIB module component files and their installed locations.

*Table 136. srsoftware component files*

File	Location	Description
srsoftware.dll	bin	Binary implementation of the srsoftware subagent.
srsoftware-mib.mib	mibs	srsoftware MIB definition document.
srsoftware.oid	config/oid	srsoftware subagent object identifier file.

---

### Guidelines

When loaded, the srsoftware subagent automatically detects installed software.

To load the subagent, use the command:

```
subagent load srsoftware
```

The recently installed software table lists software packages identified as having been installed during the last  $n$  days, where  $n$  is specified by srSoftwareSummaryRecentThreshold. Set the value of srSoftwareSummaryRecentThreshold as required.

---

### MIB Module

The srSoftware MIB is a subtree of networkharmoni(1977).

The srSoftware subtree is shown in Figure 48 on page 242.

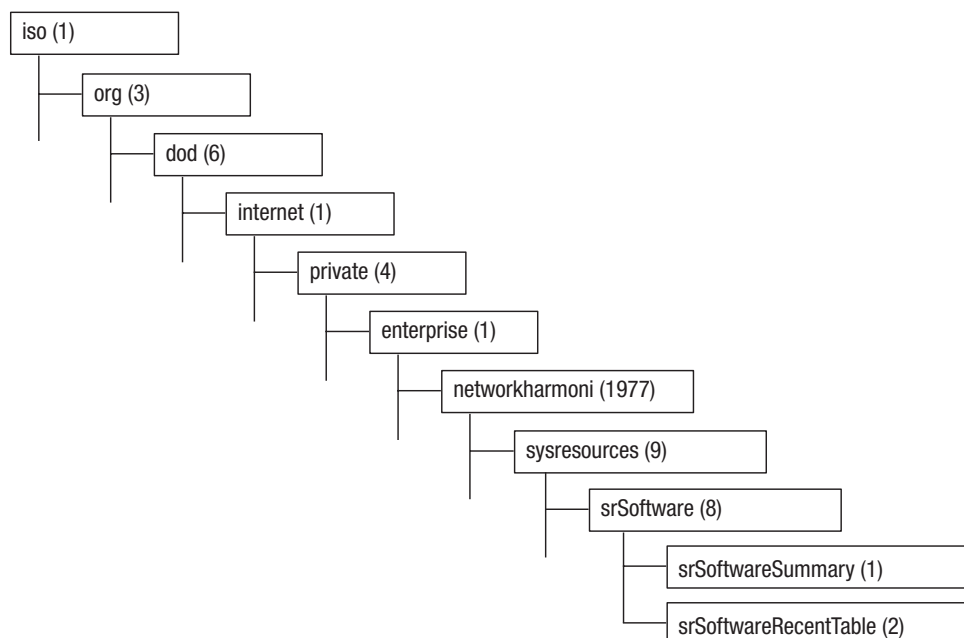


Figure 48. OID tree diagram of the srSoftware MIB module

This section provides a summary of the objects defined in the srSoftware MIB module. For detailed information on all objects in the module, see the srsoftware-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB objects

The srSoftware MIB provides a standard SNMP interface.

The MIB module contains the following items:

- A summary group, srSoftwareSummary, which defines program execution.
- An recently used table, srSoftwareRecentTable, which stores information about recently installed software packages.

### Summary group

The summary group provides information about the number of software packages installed on the local machine.

It consists of the scalar objects listed in Table 137.

Table 137. Software summary group scalar objects

Object	Description
InstalledCount	Indicates the total number of software packages installed on the local machine.
RecentCount	Indicates the number of software packages that have been installed within the last n days, where n is the value of the srSoftwareSummaryRecentThreshold object.
RecentThreshold	Sets the length (in days) of the 'recent' period. The subagent considers any software that was installed within the last n days to be recently installed software, where n is the value of this object. The default value is 7.

## Recently installed software table

The recently installed software table (`srSoftwareRecentTable`) stores information about individual software packages recently installed on the host.

Each row represents one software package. Table 138 lists the row objects in `srSoftwareRecentTable`.

*Table 138. Recently installed software table (`srSoftwareRecentTable`)*

Row object	Description
Date	The installation date of this software package or an approximation of that date.
Index	Uniquely identifies the row.
Name	The name of the installed software package.



---

## Chapter 24. Storage

The srstorage subagent and the associated srStorage MIB module provide additional statistical information to augment the data contained in the hrStorageTable of the hostres MIB.

---

### Component files

The srstorage subagent and the associated srStorage MIB module are comprised of a set of component files.

Table 139 lists the srstorage subagent and MIB module component files, and their installed locations.

*Table 139. srstorage component files*

File	Location	Description
srstorage.dll (Windows) libsrstorage.so/.sl (UNIX)	bin	Binary implementation of the srstorage subagent.
srstorage-mib.mib	mibs	srstorage MIB definition document.
srstorage.oid	config/oid	srstorage subagent object identifier file.

---

### Guidelines

When loaded, the srstorage subagent automatically populated tables in the srstorage MIB module.

To load the subagent, use the command:  
subagent load srstorage

---

### MIB module

The srStorage MIB is a subtree of networkharmoni (1977).

The srStorage subtree is shown in Figure 49 on page 246.

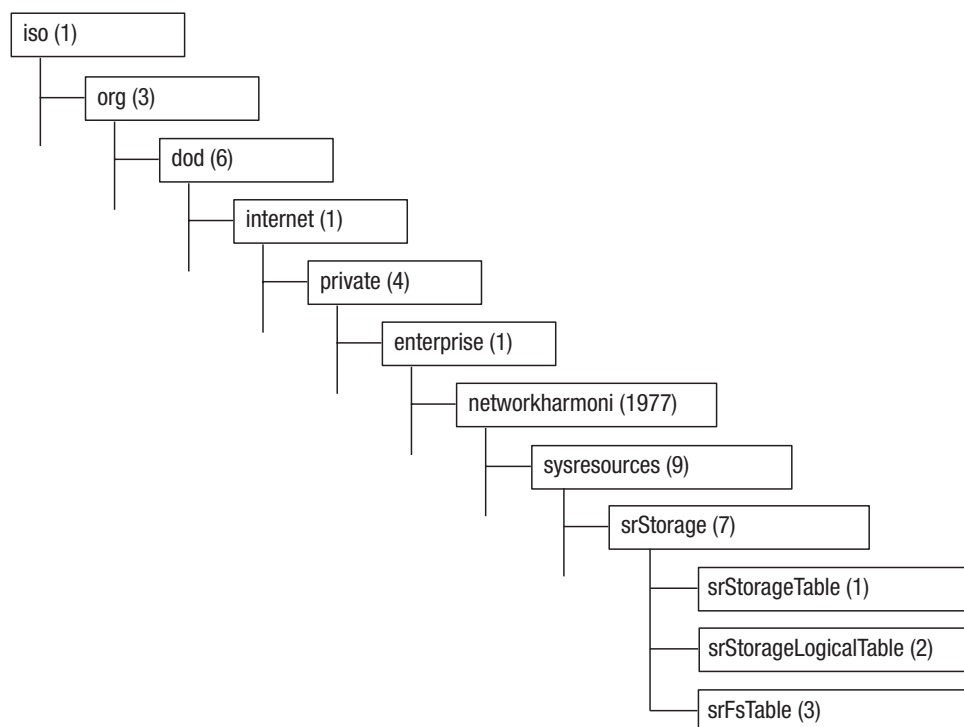


Figure 49. OID tree diagram of the srStorage MIB module

This section provides a summary of the objects defined in the srStorage MIB module. For detailed information on all objects in the module, see the srstorage-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB tables

The srStorage MIB provides a standard SNMP interface.

The MIB module contains the following tables:

- A storage table, srStorageTable
- A logical storage table, srStorageLogicalTable
- An augmented file systems table, srFsTable

### Storage table

The storage table (srStorageTable) contains information from the Host Resources hrStorageTable and provides supplementary data.

Table 140 lists the objects in srStorageTable.

Table 140. Storage table (srStorageTable)

Row object	Description
Megabytes	The amount of space currently in use on the storage area in MB.
Percent	The amount of space currently used on the storage area expressed as a percentage of its capacity.



## Logical storage table

The logical storage table (`srStorageLogicalTable`) provides access to the objects provided in `srStorageTable` by indexing them according to device type rather than `HrStorageIndex`.

Table 141 lists the objects provided in `srStorageLogicalTable`.

*Table 141. Logical storage table (`srStorageLogicalTable`)*

Row object	Description
HrIndex	The <code>hrStorageIndex</code> value for this entry. This index uniquely identifies the storage area in <code>hrStorageTable</code> and <code>srStorageTable</code> .
Type	The logical type of the storage area. These values map directly to the least significant sub-object ID of the corresponding <code>hrStorageType</code> .  other(1) ram(2) swap(3) fixed(4) removable(5) floppy(6) cdrom(7) ramdisk(8) flash(9) network(10) thirdparty(100)
UsedPercent	The amount of space currently used on the storage area expressed as a percentage of its capacity.

## Augmented file systems table

The augmented file systems table (`srFsTable`) supplements the information provided in the Host Resources `hrFsTable`.

Table 142 describes the objects provided in `srFsTable`.

*Table 142. Augmented file systems table (`srFsTable`)*

Row object	Description
FreeFiles	The number of files/inodes available on the file system.
MaxFiles	The maximum number of files/inodes that can be stored on the file system.
UsedFiles	The number of files/inodes present on the file system.
UsedFilesPercent	The number of files/inodes present on the file system, expressed as a percentage of <code>srFsMaxFiles</code> .



---

## Chapter 25. Service discovery

The `svcdisc` subagent provides information on the relationship between clients, servers and the applications they are using. This supports application topology discovery as it can be used to discover the relationship between clients and servers for each different application.

The `svcdisc` subagent and the associated `svcDisc` MIB module provide facilities for discovering and monitoring network applications. They deliver data about the availability of applications on servers as well as application topology information detailing the relationships between applications, application servers and the network elements that connect clients and servers of applications running over networks. You can configure the `svcdisc` subagent to filter out unwanted or infrequently accessed services.

---

### Component files

The `svcdisc` subagent and MIB module are comprised of a set of component files.

Table 143 lists the `svcdisc` subagent and MIB module component files and their locations.

*Table 143. svcdisc component files*

File	Location	Description
<code>svcdisc.dll</code> (Windows) <code>libsvcdisc.so/.sl</code> (UNIX)	<code>bin</code>	Binary implementation of the <code>svcdisc</code> subagent.
<code>svcdisc-mib.mib</code>	<code>mibs</code>	<code>svcdisc</code> MIB definition document.
<code>svcdisc.oid</code>	<code>config/oid</code>	<code>svcdisc</code> subagent object identifier file.

---

### Guidelines

Use the `svcdisc` subagent to gather data about the relationship between clients, servers and the applications.

To load the subagent, use the command:

```
subagent load svcdisc
```

### Discovering services

When discovering services, the subagent registers a service as discovered when the number of client connections to that service within a given period exceeds a threshold value. Once it has discovered a service, the subagent logs each new client connection to the service. If the service becomes inactive for a specified time, the subagent de-registers the service. Once a service has been de-registered, it must be re-registered as a discovered service before reporting resumes.

The following control row objects in `svcDiscControlTable` define the service discovery behavior:

- `svcDiscControlDiscoveryThreshold` - the number of connections within a sample period

- svcDiscControlDiscoveryWindowSize - the sample period
- svcDiscControlDiscoverySample - the number of consecutive sample periods that meet the threshold
- svcDiscControlDiscoveryTimeout - the period of server inactivity observed before a service is de-registered

Before a service is registered as discovered, it must satisfy the following conditions:

- The number of connections must be greater than or equal to the value of svcDiscControlDiscoveryThreshold
- The number of connections must occur within the window period defined by svcDiscControlDiscoveryWindowSize
- The previous two conditions must be satisfied for a minimum number of consecutive window periods defined by svcDiscControlDiscoverySample

## Inivars

The svcdisc subagent provides inivars for configuring its operation.

These inivars are listed in Table 144.

*Table 144. svcdisc subagent inivars*

Inivar	Values	Description
svcdiscClientConfig	string	Specifies the configuration file to be executed whenever the subagent detects that the host machine is acting as client to a new server.  If this variable is not defined, no configuration file is executed.
svcdiscPersistent	true   false	If the value of this variable is true, the data stored by this subagent is persistent between agent instances, ensuring that if the agent is rebooted, client data is not lost.  Default value: false
svcdiscServerConfig	string	Specifies the configuration file to be executed whenever the subagent detects that the host machine is acting as a server to a new client.  If this variable is not defined, no configuration file is executed.

For general information about inivars, see the *Netcool/SSM Administration Guide*.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The `svcdisc` commands create rows in the control table (`svcDiscControlTable`).

The general syntax of these commands is:

```
svcdisc property=value  
svcdisc create [property=value ...]  
svcdisc reset
```

Table 145 lists the properties supported in these commands.

*Table 145. Configuration command parameters - svcdisc*

Property	Type	Description	Sets MIB object
buckets	int	The number of summary table rows requested.	BucketsRequested
client	IP	The IP address of the monitored client.	ClientAddr
clients	enum	Controls client monitoring:  off  on	ClientMonitoring
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl
datasource	OID	The interface monitored by the control row.	DataSource
description	string	A description of the control row.	Description
history	int	The history table size limit.	HistoryInterval
maxport	int	The maximum port of the monitored protocol.	MaxPort
minport	int	The minimum port of the monitored protocol.	MinPort
sample	int	The sample interval.	DiscoverySample
server	IP	The IP address of the monitored server.	ServerAddr
thresh	int	The service discovery threshold.	DiscoveryThreshold
timeout	int	The connection timeout interval.	DiscoveryTimeout
trans	enum	The monitored transport protocol:  all  tcp  udp	TransportProtocol
window	enum	The service discovery window.	DiscoveryWindowSize

---

## Examples

This example demonstrates how to perform simple monitoring tasks using the appusage subagent.

### Discovering frequently used services

Monitor all network services and all clients of those services, but ignore bursts of sporadic activity. Discover only those services that are initially accessed at least 10 times in one hour, and again at least 10 times in the next hour, then monitor each discovered service until it becomes inactive for a period of 24 hours. After 24 hours of inactivity, de-register the service. The 24 hour timeout ensures that services used daily, but only at the same time each day, are not de-registered and monitoring occurs as usual.

The subagent configuration commands for creating this service discovery monitor are as follows:

```
subagent load svcdisc
svcdisc reset
svcdisc description="Example - Monitor all services"
svcdisc thresh=10
svcdisc window=3600
svcdisc sample=2
svcdisc timeout=86400
svcdisc client=0.0.0.0
svcdisc server=0.0.0.0
svcdisc create
```

---

## MIB module

The svcDisc MIB is a subtree of networkharmoni(1977).

The svcDisc subtree is shown in Figure 50 on page 253.

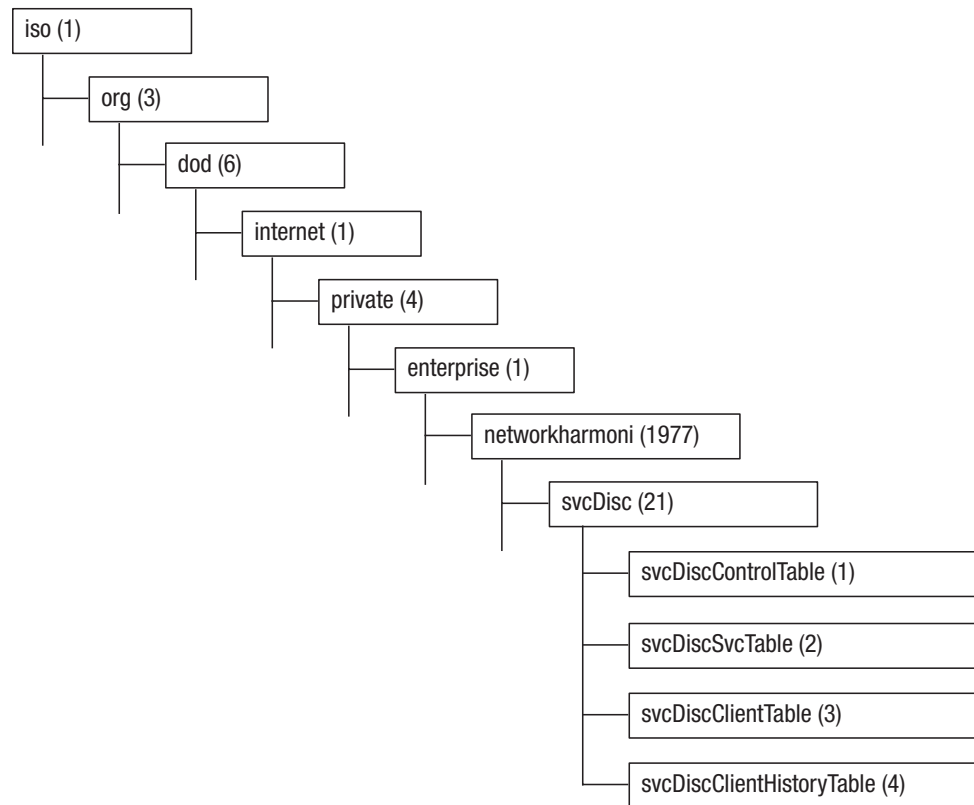


Figure 50. OID tree diagram of the svcDisc MIB module

This section provides a summary of the objects defined in the svcDisc MIB module. For detailed information on all objects in the module, see the `svcdisc-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB tables

The svcDisc MIB provides a standard SNMP interface with control row semantics.

The MIB module contains the following tables:

- A control table, `svcDiscControlTable`
- A service table, `svcDiscSvcTable`
- A client table, `svcDiscClientTable`
- A client history table, `svcDiscClientHistoryTable`

### Control table

The control table (`svcDiscControlTable`) contains control information for configuring service discovery and monitoring.

Table 146 lists the row objects in `svcDiscControlTable`.

Table 146. Control table (`svcDiscControlTable`)

Row object	Description
BucketsGranted	Indicates the number of rows in <code>svcDiscClientHistoryTable</code> granted by the subagent to this control row.

Table 146. Control table (svcDiscControlTable) (continued)

Row object	Description
BucketsRequested	Sets the number of rows in svcDiscClientHistoryTable requested for storing the data associated with this control row.
ClientAddr	Specifies the IP address of the monitored client. The default value is 0.0.0.0, which monitors all clients.
ClientMonitoring	Controls client monitoring:  on(1) - Enables client monitoring  off(2) - Disables client monitoring
DataControl	Sets the monitoring status of the control row: on(1) - Enables data collection.off(2) - Suspends data collection.
DataSource	Selects the data source to be monitored. It is specified as the full OID to the ifIndex object in MIB-2 with the index to the desired interface appended as a sub-OID, for example: .1.3.6.1.2.1.2.2.1.1.ifIndex value
Description	A user-defined description of the control row.
DiscoverySample	Sets the minimum number of consecutive windows, whose length is defined by svcDiscControlDiscoveryWindowSize,during which the number of connections to a service must exceed the value specified by svcDiscControlDiscoveryThreshold before the subagent registers the service as a discovered service and collects data about it.
DiscoveryThreshold	Sets the minimum number of connections to a service that the subagent must observe within the window defined by svcDiscControlDiscoveryWindowSize before it registers the service as a discovered service and commences collecting data about it.
DiscoveryTimeout	Defines the period (in seconds) during which the subagent must observe no connections to a service before it de-registers that service and stops collecting data about it.
DiscoveryWindowSize	Sets the interval or 'window' (in seconds) over which the subagent must observe the number of connections specified by svcDiscControlDiscoveryThreshold before it registers the service as a discovered service and commences collecting data about it.
HistoryInterval	Specifies the interval (in seconds) at which data is transferred from svcDiscClientTable to svcDiscClientHistoryTable.
Index	Uniquely identifies this row in the control table.
MaxPort	Sets the maximum port number monitored.
MinPort	Sets the minimum port number monitored.
Owner	Indicates the owner of the control row.
ServerAddr	Specifies the IP address of the monitored server. The default value is 0.0.0.0, which monitors all servers.
Status	An SNMPv2 row status object controlling creation, activation and deletion of the control row.



Table 146. Control table (svcDiscControlTable) (continued)

Row object	Description
TransportProtocol	Specifies the type of traffic to be monitored:  all(1) - Both TCP and UDP  tcp(2) - TCP only  udp(3) - UDP only

## Service table

The service table (svcDiscSvcTable) lists details about the services discovered by each control row in svcDiscControlTable.

Table 147 describes the row objects in svcDiscSvcTable.

Table 147. Service table (svcDiscSvcTable)

Row object	Description
Application	The name of the application that provides the service discovered. If no name is determined, the value is unknown.
ServerAddr	IP address of the server providing the discovered service.
ServerPort	Server port number used to provide the discovered service.
TimeInactive	The amount of time (in seconds) that the service has been inactive.
TransportProtocol	Transport protocol used by the discovered service:  all(1) - Both TCP and UDP  tcp(2) - TCP only  udp(3) - UDP only

## Client table

The client table (svcDiscClientTable) stores data about the clients of each service discovered. This data is only stored if the svcDiscControlClientMonitoring object in the corresponding control row has the value on(1).

Table 148 describes the objects in svcDiscClientTable.

Table 148. Client table (svcDiscClientTable)

Row object	Description
Application	Index from svcDiscSvcTimeInactive.
ClientAddr	The IP address of the client accessing the service.
ClientName	The IP name of the client host.
ServerAddr	Index from svcDiscSvcServerAddr
ServerName	The IP name of the server host.
ServerPort	Index from svcDiscSvcServerPort
TransportProtocol	Index from svcDiscSvcTransportProtocol

### **Client history table**

The client history table (svcDiscClientHistoryTable) table contains historical data transferred from the client table, svcDiscClientTable.

The subagent transfers data from svcDiscClientTable to svcDiscClientHistoryTable at regular intervals, as specified by the svcDiscControlHistoryInterval object in the corresponding control row. The number of history table rows allocated to a control row is set by the row's svcDiscControlBucketsRequested object.

---

## Chapter 26. Server security

The svrsecurity subagent provides host-based intrusion detection by looking for any indication of a security breach. It can detect and recognize excessive login attempts, denial-of-service attacks and port-scanning attempts.

This subagent helps IT organizations to protect their system and network infrastructure against unauthorized users. It recognizes when unauthorized users attempt to access systems using an Internet service such as FTP, telnet and rlogin, helping to prevent intruders from invading business applications.

The svrsecurity subagent and the associated svrSecurity MIB provide facilities for reporting activity about well known security attacks on the server by passively monitoring network traffic to and from the server for known attack signatures. When a potential attack is detected, the subagent logs details about the attack and, if required, generates an event to indicate the attack.

---

### Component files

The svrsecurity subagent and MIB module are comprised of a set of component files.

Table 149 lists the svrsecurity subagent and MIB module component files and their installed locations.

*Table 149. svrsecurity component files*

File	Location	Description
svrsecurity.dll (Windows) libsvrsecurity.so/.sl (UNIX)	bin	Binary implementation of the svrsecurity subagent.
svrsecurity-mib.mib	mibs	svrsecurity MIB definition document.
svrsecurity.oid	config/oid	svrsecurity subagent object identifier file.
svrsecurity.cfg	config	Sample configuration file.

### Attacks

The subagent is capable of detecting common types of attack.

The types of attack that the subagent can monitor are:

- SYN flood

A SYN flood occurs when the number of incomplete connections increases suddenly. This is known as a denial of service attack and is a common form of attack.

- Ping flooding

A ping flood is an attempt to saturate a network with packets (ICMP) in order to stop legitimate traffic. This is known as a denial of service attack and is a common form of attack.

- Failed login attempts

Consecutive login failures may indicate attempts at unauthorized access.

- Port scanning

Port scanning is a common technique used to detect the services available on a server and is an indicator of a potential attack.

---

## Guidelines

To monitor security attacks on a server, create a control table row and one or more parameter table rows, one parameter row for each attack type that you wish to detect. In each parameter row specify the type of detector and the parameters for that detector.

### Before you begin

Ensure that the `svrsecurity` subagent is loaded. To load the subagent, use the command:

```
subagent load svrsecurity
```

### Procedure

The general procedure for configuring security monitoring is as follows:

1. Create a control row in the control table by setting the `svrSecurityControlStatus` to `CreateandWait(5)`.
2. Set `svrSecurityControlParamTableSize` to the number of desired detector types. For example, if you wish to detect all four attack types, set the table size to 4.
3. In each parameter row, set `svrSecurityParamDetectorIndex` to the detector table index of the attack type that you wish to monitor; then, set the other parameters as appropriate to the detector type or use the default parameters.
4. Activate the control row by setting the `svrSecurityControlStatus` to `active(1)`. Details about any attack detected are stored in the log table.

### What to do next

If you are monitoring failed login attempts, in addition to setting up control rows, ensure that the failed login attempts are written to the message log. For example, on Linux systems, set `/etc/xinetd.conf` to `HOST USERID` and `ATTEMPT`:

```
log_on_failure = HOST USERID ATTEMPT
```

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The `svrsecurity` commands create rows in the control table (`svrSecurityControlTable`).

The general syntax of these commands is:

```
svrsecurity property=value  
svrsecurity create [property=value ...]  
svrsecurity reset
```

Table 150 lists the properties supported in these commands.

*Table 150. Configuration command parameters - svrsecurity*

Property	Type	Description	Sets MIB object
buckets	int	The number of summary table rows requested.	BucketsRequested
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl
datasource	OID	Sets the interface monitored by the control row.	DataSource
description	string	A description of the control row.	Description
event	int	The index of the event generated if an attack is detected.	EventIndex
eventstatus	enum	Event flow control:  alwaysready  fired  ready	EventStatus
offevent	int	The control row deactivation event index.	TurnOffEventIndex
onevent	int	The control row activation event index.	TurnOnEventIndex

## Parameter table

The `svrsecurityParam` commands create rows in the parameter table (`svrSecurityParamTable`) and assign them to the next server security control row created using the `svrsecurity create` command.

The general syntax of these commands is:

```
svrsecurityParam property=value  
svrsecurityParam store [property=value ...]  
svrsecurityParam reset
```

Table 151 lists the properties supported in these commands.

*Table 151. Configuration command parameters - svrsecurityParam*

Property	Type	Description	Sets MIB object
detectorindex	int	Detector index.	DetectorIndex
paramA	string	Generic string parameter A.	ParameterA
paramB	string	Generic string parameter B.	ParameterB

Table 151. Configuration command parameters - *svrsecurityParam* (continued)

Property	Type	Description	Sets MIB object
param1	int	Generic string parameter 1.	Parameter1
param2	int	Generic string parameter 2.	Parameter2
param3	int	Generic string parameter 3.	Parameter3

## Examples

These examples demonstrate how to perform simple monitoring tasks using the *svrsecurity* subagent.

### Detecting ping flood attacks

Monitor interface #3 of the host server for ping flood attacks, where an attack is defined to be more than 1000 pings per minute, with a floor value of 50. Generate an event if such an attack is detected. Log a history of the last 10 attacks.

The subagent configuration commands required for creating this monitor are:

```
subagent load rmonc

event reset
event type=snmp-trap
event community=public
event description="Ping flood event"
event create
pingFloodEvent=?

subagent load svrsecurity
svrsecurity reset
svrsecurity datasource=$ifIndex.3
svrsecurity description="Ping flood security monitor"
svrsecurity buckets=10
svrsecurity event=$pingFloodEvent
svrsecurity eventstatus=ready
svrsecurityParam reset
svrsecurityParam detectorindex=2
svrsecurityParam param1=60
svrsecurityParam param2=1000
svrsecurityParam param3=50
svrsecurityParam store
svrsecurity create
```

### Detecting failed SSH login attempts

Monitor the local SSH daemon for failed login attempts. Generate an event if a failed login is detected, and log a history of the last 10 attacks.

The subagent configuration commands required for creating this monitor are:

```
subagent load rmonc

event reset
event type=snmp-trap
event community=public
event description="SSH login failure"
event create
loginEvent=?

subagent load svrsecurity
svrsecurity reset
```

```

svrsecurity description="SSH login security monitor"
svrsecurity buckets=10
svrsecurity event=$loginEvent
svrsecurity eventstatus=ready
svrsecurityParam reset
svrsecurityParam detectorindex=3
svrsecurityParam paramA="/var/log/secure"
svrsecurityParam store
svrsecurity create

```

## MIB module

The svrSecurity MIB module is a subtree of networkharmoni (1977).

The svrSecurity subtree is shown in Figure 51.

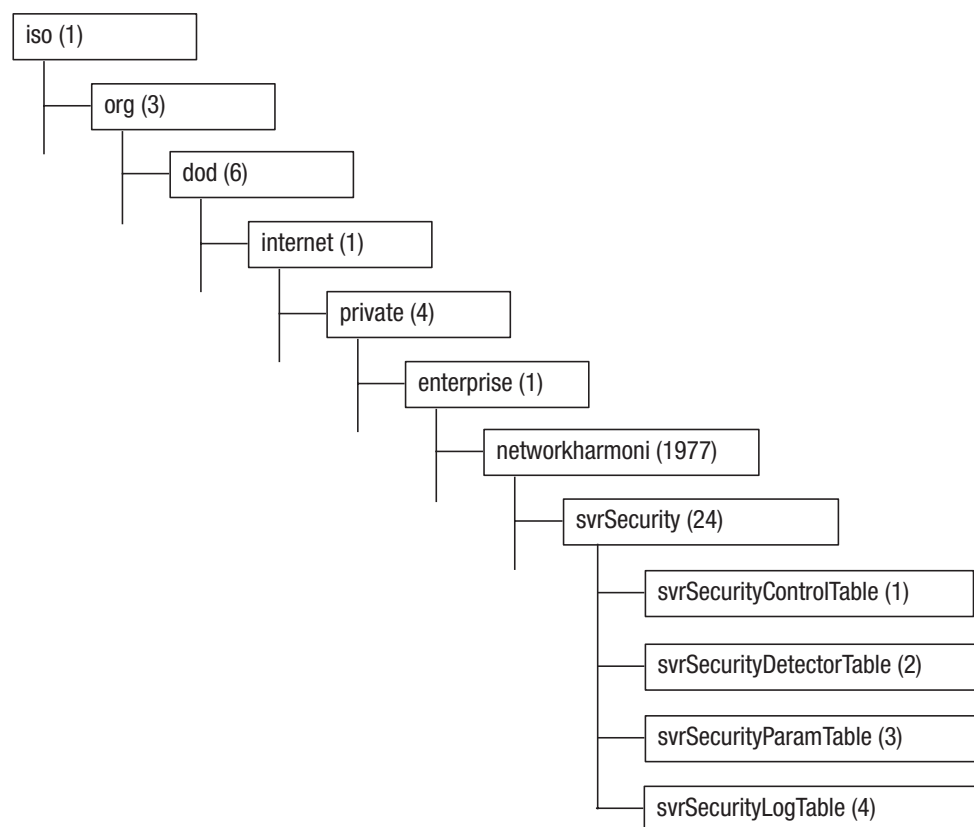


Figure 51. OID tree diagram of the svrSecurity MIB module

This section provides a summary of the objects defined in the svrSecurity MIB module. For detailed information on all objects in the module, see the svrsecurity-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB tables

The svrSecurity MIB provides a standard SNMP interface with control row semantics.

The MIB module contains the following tables:

- A control table, svrSecurityControlTable
- A detector table, svrSecurityDetectorTable
- A parameter table, svrSecurityParamTable
- A log table, svrSecurityLogTable

### Control table

The control table (svrSecurityControlTable) defines server security monitors. Each control row monitors one or more attacks on the host. Each attack type has a set of parameters associated with it, which configure the detector for that attack type.

Table 152 lists the row objects in svrSecurityControlTable.

*Table 152. Server security control table (svrSecurityControlTable)*

Row object	Description
BucketsGranted	Indicates the number of rows in svrSecurityLogTable granted by the subagent to this control row.
BucketsRequested	Sets the number of rows in svrSecurityLogTable requested for storing the data associated with this control row.
CreateTime	The value of sysUpTime when the control row was last activated.
DataControl	Sets the monitoring status of the control row:  on(1) - Enables data collection  off(2) - Suspends data collection
DataSource	Identifies the source of the data (that is, the interface) monitored by the control row. This source can be any interface on the device hosting the subagent. The value of this object identifies the interface by its ifIndex object.  For example, for a control row to monitor data using interface #1, this object must be set to ifIndex.1.
Description	A description of the purpose of the control row.
EventIndex	Contains the RMON eventIndex of the event generated if an attack is detected. If this index does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then no event is generated.
EventStatus	Sets the event flow control for the events associated with this row:  eventReady(1) - A single event may be generated, after which the value of this object changes to eventFired(2).  eventFired(2) - Disables event generation. No events may be generated until the object is modified to eventReady(1) or eventAlwaysReady(3).  eventAlwaysReady(3) - Disables the flow control, allowing events to be generated without restriction. Using this setting is not recommended as it can result in high network traffic or other performance problems.



Table 152. Server security control table (svrSecurityControlTable) (continued)

Row object	Description
Index	Uniquely identifies the control table row.
Owner	Indicates the owner of the control row.
ParamTableSize	Sets the number of entries in svrSecurityParamTable associated with this control row.
Status	An SNMPv2 row status object controlling creation, activation and deletion of the control row.
TurnOffEventIndex	Contains the RMON eventIndex of the event configured to deactivate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then deactivation via an event is disabled.
TurnOnEventIndex	Contains the RMON eventIndex of the event configured to activate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then activation via an event is disabled.

## Detector table

The detector table (svrSecurityDetectorTable) stores a list of the security attack types that the svrsecurity subagent is capable of monitoring. Each row in the table type represents one attack type.

Rows in this table are indexed by the svrSecurityDetectorIndex object and contain the svrSecurityDetectorDetection object, which indicates the detector type that the row represents.

The detector types currently supported and the corresponding detector index values are:

1. SYN flood
2. Ping flood
3. Failed logins
4. Port scan

## Parameter table

The parameter table (svrSecurityParamTable) provides a list of generic parameters. Each parameter table row is associated with a control table row and selects an attack detector and specifies the parameters for that detector. The number of parameter table rows associated with a control row is set by the svrSecurityControlParamTableSize object.

The parameters have generic names and are interpreted according to the type of detector. The function of each parameter for the various attack detector types is explained in Table 153 to Table 157 on page 265.

Table 153. svrSecurityParamTable values for ping flood detection

Parameter	Default value	Description
Parameter1	10	Sample Interval: Defines the interval (in seconds) at which the agent periodically checks for an attack. The valid range is 1 to 2147483647
Parameter2	500	Alert Threshold: If the number of pings in one sample period exceeds this threshold then the detector raises an intrusion alert.

Table 153. *svrSecurityParamTable* values for ping flood detection (continued)

Parameter	Default value	Description
Parameter3	100	Ping Threshold: Defines a 'floor value' for samples. If a sample contains a number of pings less than this threshold, it is discarded. Use this parameter in fine-tuning the agent to prevent spurious alerts in environments that generate a certain amount of ping traffic during normal operation, such as during a node discovery process. Ping flood alerts occur if the current sample contains a number of pings greater than $((2 * \text{stddev}) + \text{mean})$ . If the node hasn't received any ping packets for a long time, the mean and stdev drop to 0, so that receiving 2 or 3 pings in a sample would generate an alert. A correctly set floor value prevents this situation.
ParameterA	N/A	N/A
ParameterB	N/A	Reserved for future use.

Table 154. *svrSecurityParamTable* values for syn flood detection

Parameter	Default value	Description
Parameter1	10	Defines the interval (in seconds) at which the agent periodically checks for an attack. The valid range is 1 to 2147483647.
Parameter2	20	Alert Threshold: If the number of dropped syns in one sample period exceeds this threshold then the detector raises an intrusion alert.
Parameter3	N/A	N/A
ParameterA	N/A	N/A
ParameterB	N/A	Reserved for future use.

Table 155. *svrSecurityParamTable* values for port scan detection

Parameter	Default value	Description
Parameter1	10	Defines the interval (in seconds) at which the agent periodically checks for an attack. The valid range is 1 to 2147483647.
Parameter2	N/A	N/A
Parameter3	N/A	N/A
ParameterA	N/A	N/A
ParameterB	N/A	Reserved for future use.

Table 156. *svrSecurityParamTable* values for login monitor detection

Parameter	Default value	Description
Parameter1	N/A	N/A
Parameter2	N/A	N/A
Parameter3	N/A	N/A
ParameterA	See Table 157 on page 265 for default values.	Log file path and name for UNIX only. Detects failed login attempts as a security attack. This parameter is only relevant on UNIX platforms (on Windows systems, the detector uses the System Event Log). The default value is platform dependent. Ensure that logging is enabled.

Table 156. *svrSecurityParamTable* values for login monitor detection (continued)

Parameter	Default value	Description
ParameterB	N/A	Reserved for future use.

Table 157. *svrSecurityParamParameterA* default values for log monitor detection on UNIX

UNIX platform	ParamParameterA default value
AIX	/etc/security/failedlog
HPUX	/var/adm/btmp
Linux	/var/log/messages
if default not found	/var/log/secure
Solaris	/var/adm/loginlog

**Note:** If a parameter does not apply to the detector, leave it blank.

## Log table

The log table (*svrSecurityLogTable*) stores details of any attack detected. A log table row is created each time the security monitor defined by a control row detects an attack. The number of log table rows associated with a control row is set by *svrSecurityControlBucketsRequested*.

Table 158 lists the row objects in *svrSecurityLogTable*.

Table 158. *Server security log table (svrSecurityLogTable)*

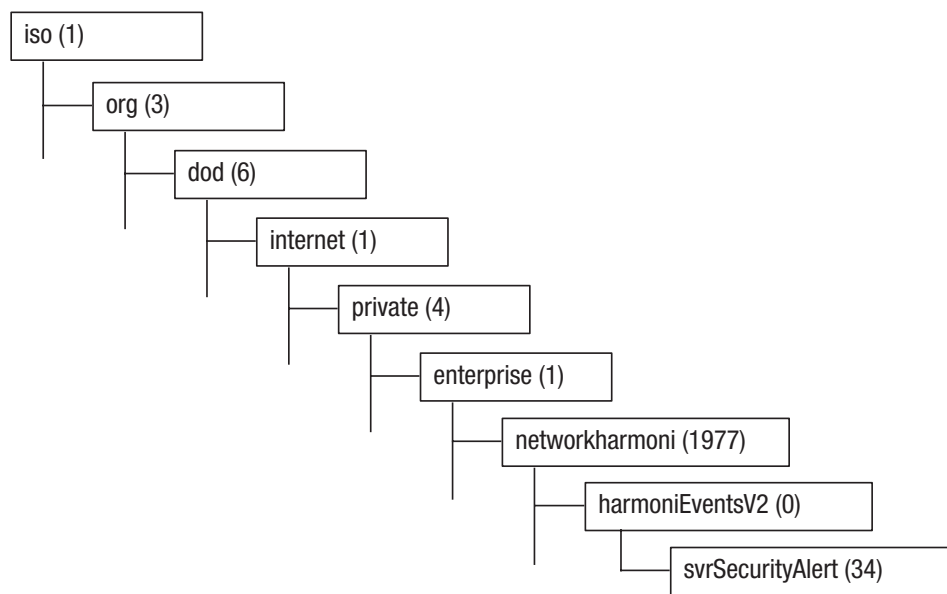
Row object	Description
Index	Uniquely identifies the log table row.
LogMessage	A message generated by the attack detector.
TimeStamp	The data and time at which the attack was detected.

**Note:** Log table rows are also created when exceptional circumstances are detected. For example, the ping flood detector calculates the mean value for ping flooding over 10 samples. If any changes in ping flooding behavior occurs, indicating activity that is 2 standard deviations outside the mean value, a log table row is created to record this.

## Notification types

The *svrSecurity* MIB defines the *svrSecurityAlert* notification type for events generated by the subagent when attacks are detected. The variable bindings in this notification type indicate details of the attack detected.

The location of this notification is indicated in Figure 52 on page 266.



*Figure 52. OID tree diagram for svrSecurityAlert notification type*

The `svrSecurityAlert` notification type contains the following variable bindings:

- `svrSecurityDetectorDescription`
- `svrSecurityLogTimeStamp`
- `svrSecurityLogLogMessage`

---

## Chapter 27. System resources

The sysres subagent extends the functionality of host resource management to include additional data about CPU, memory and disks. It allows you to monitor system and application log files and gather data on system and application events.

The sysres subagent and the associated sysRes MIB module collate and report system information. General system data is stored under the group srSystem, while other data is gathered and stored in tables. The level of system information conforms to RFC 1514/2790, as well as providing additional system information reporting.

---

### Component files

The sysres subagent and sysRes MIB module are comprised of a set of component files.

Table 159 lists the sysres subagent and MIB module component files and their locations.

*Table 159. sysres component files*

File	Location	Description
sysres.dll (Windows) libsysres.so/.sl (UNIX)	bin	Binary implementation of the sysres subagent.
sysres-mib.mib	mibs	sysres MIB definition document.
sysres.oid	config/oid	sysres subagent object identifier file.

---

### Guidelines

Use the subagent to access system resource information and metrics, and to monitor the contents of log files.

To load the subagent, use the command:

```
subagent load sysres
```

### Inivars

The sysres subagent provides the inivars for configuring its operation.

The subagent inivars are listed in Table 160.

*Table 160. System resources subagent inivars*

Inivar	Type	Description
LogMonCompatMode	enum	Controls the behaviour of the logmon state file: 3 - saves the logmon state file when the agent shuts down 4 - saves the logmon state file when the agent is saved

Table 160. System resources subagent inivars (continued)

Inivar	Type	Description
LogMonLineCount	enum	Controls the line counting behavior of the log file monitor: on off  See "Log file line numbers" on page 270 for details.
LogMonStateFile	string	Specifies the file in which srLogMon stores its internal state information. See "State information" on page 269 for details.
sysresUpdateInterval	int	Sets the interval (in seconds) at which the subagent updates the system metrics.  Default - 20 seconds.
WinSysSwapUsePagingFile (Windows only)	enum	Controls how srSystemSwapPercentUsed is calculated: true - uses perfmon Paging File\%Usage metrics instead of Memory\%Committed Bytes in Use false - uses perfmon Memory\%Committed Bytes in Use metrics  Default - false

For general information about inivars, see the *Netcool/SSM Administration Guide*.

## Monitoring log files

The subagent enables you to monitor lines of text in a log file. It can be set up to filter lines of text that have been appended to a log file. The subagent periodically monitors the file at an interval set by the update interval object (srLogMonControlUpdateInterval).

### About this task

To monitor log files:

#### Procedure

1. Establish a control row by setting the srLogMonControlStatus to Create and Wait.
2. Enter the name of the log file (srLogMonControlLogFile) including its absolute path.
3. Specify a filter expression (srLogMonControlFilter) that identifies the log file message that you wish to monitor.  
The default expression is .\*(see "Regular expressions" on page 269 for more details).
4. Set other parameters as required.  
See the object descriptions for details of allowed values.
5. Set srLogMonControlStatus to active.  
The agent commences monitoring the log file.

## Monitoring system event logs on Windows systems

The subagent also enables you to monitor Windows system event logs, which are not text files.

### About this task

Windows event log entries are viewed and presented in the following order:

date, time, type, source, message

To use the subagent to monitor Windows system event logs, use a filter expression (`srLogMonControlFilter`) with the format `*logname` where `logname` denotes a Windows event log name; for example, `System`, `Security` or `Application`.

### Regular expressions

The log file filter expression may contain regular expressions.

The log file monitor evaluates regular expressions according to the following rules:

- String matching is case-sensitive
- Every exact line match is recorded by `srLogMon`
- To match a substring, the filter must contain the expression `*string.*` where `string` denotes the desired matching substring

For more details on regular expression syntax, see Appendix D, “Regular expressions,” on page 615.

### State information

Normally the log file monitor starts log file monitoring from the end of a file. If the agent becomes unavailable for some reason, such as agent reboot, this can lead to new entries in log files going undetected if they were added to the log file while the agent was unavailable. To eliminate this possibility, you can configure the log file monitor to store information about its internal state in a separate file when it shuts down. This it to continue monitoring log files from the point at which it previously finished.

To enable this function, add the following entry to the `init.cfg` file:

```
LogMonStateFile=filename
```

`filename` denotes the file in which the log file monitor stores its internal state information.

**Attention:** The subagent overwrites the file specified by the `logMonStateFile` inivar. Never set the value of this inivar to the name of any Netcool/SSM core configuration file.

**Note:** Control rows that are created after the agent has booted always commence log file monitoring from the end of a file. The `LogMonStateFile` variable does not affect this behavior.

## Log file line numbers

The log file monitor stores the line number of any filter match in the srLogMonLine object. Very large log files containing tens of thousands of lines or more may degrade performance of the log file monitor because of the overhead involved in calculating line numbers.

If necessary, you can disable line counting by adding the following entry to the init.cfg file:

```
LogMonLineCount=off
```

---

## System resources object support

System resources objects are located in the subtree under the OID (.1.3.6.1.4.1.1977.9). Some of these objects are not available on every platform. Sometimes an object is available on a platform but derives from two of the standard objects to form a modified object with different semantics.

Table 161 provides a summary of the objects available on each platform.

*Table 161. System resources object support by platform*

Sub-OID	MIB objects under OID: .1.3.6.1.4.1.1977.9	Win	Linux	Solaris	AIX	HP-UX
srSystem Objects		%	%	%	%	%
1.1	srSystemProcessCount	X	X	X	X	X
1.2	srSystemOpenFileCount	X	X	X	X	X
1.3	srSystemMemoryInUse	X	X	X	X	X
1.4	srSystemActiveMemory	X	X	X	X	X
1.5	srSystemSwapInUse	X	X	X	X	X
1.6	srSystemTotalSwap	X	X	X	X	X
1.7	srSystemSwapPercentUsed	X	X	X	X	X
1.8	srSystemRunQueueLen	X	X	X	X	X
1.9	srSystemTrapCount			X	X	X
1.10	srSystemSyscallCount	X		X	X	X
1.11	srSystemInterruptCount	X	X	X	X	X
1.12	srSystemContextSwitchCount	X	X	X	X	X
1.13	srSystemDiskWaitCount			X		X
1.14	srSystemPageWaitCount					X
1.15	srSystemSwappedRunnableCount	X	X	X	X	X
1.16	srSystemSleepingResidentCount	X	X	X	X	X
1.17	srSystemPageSwapInCount		X	X		X
1.18	srSystemPageSwapOutCount		X	X		X
1.19	srSystemSwapInCount			X		X
1.20	srSystemSwapOutCount			X		X
1.21	srSystemPageInCount	X	X	X	X	X
1.22	srSystemPageOutCount	X	X	X	X	X
1.23	srSystemPageReclaimCount	X		X	X	X



Table 161. System resources object support by platform (continued)

Sub-OID	MIB objects under OID: .1.3.6.1.4.1.1977.9	Win	Linux	Solaris	AIX	HP-UX
1.24	srSystemHardPageFaultCount	X	X	X	+	+
1.25	srSystemSoftPageFaultCount	X	X			
1.26	srSystemLoad1	X	X	X	X	X
1.27	srSystemLoad5	X	X	X	X	X
1.28	srSystemLoad15	X	X	X	X	X
1.29	srSystemCPUUsage	X	X	X	X	X
1.30	srSystemDeviceCount	X	X	X	X	X
1.31	srSystemDiskSize	X	X	X	X	X
1.32	srSystemDiskInUse	X	X	X	X	X
1.33	srSystemCPUUsageAverage	X	X	X	X	X
1.34	srSystemCPUUsageSamplePeriod	X	X	X	X	X
srProcessorTable Objects		X	%	X	X	X
2.1.1	srProcessorIndex	X	X	X	X	X
2.1.2	srProcessorIdle	X	X	X	X	X
2.1.3	srProcessorUser	X	X	X	X	X
2.1.4	srProcessorSys	X	X	X	X	X
2.1.5	srProcessorWait	X		X	X	X
2.1.6	srProcessorIdlePercent	X	X	X	X	X
2.1.7	srProcessorUserPercent	X	X	X	X	X
2.1.8	srProcessorSysPercent	X	X	X	X	X
2.1.9	srProcessorWaitPercent	X		X	X	X
2.1.10	srProcessorLastUpdate	X	X	X	X	X
srDiskTable Objects		X	%	X	%	%
3.1.1	srDiskIndex	X	X	X	X	X
3.1.2	srDiskQueueLength	X	2.4+	X	X	% †
3.1.3	srDiskUtilization	X	2.4+	X		% †
3.1.4	srDiskAccessTime	X	2.4+	X		% †
3.1.5	srDiskReadCount	X	X	X	X	% ‡
3.1.6	srDiskWriteCount	X	X	X	X	% ‡
3.1.7	srDiskAccessCount	X	X	X	X	X
3.1.8	srDiskKB	X	X	X	X	X
3.1.9	srDiskLastUpdate	X	X	X	X	X
srLogMonControlTable Objects		X	X	X	X	X
4.1.1.1	srLogMonControlIndex	X	X	X	X	X
4.1.1.2	srLogMonLogFile	X	X	X	X	X
4.1.1.3	srLogMonFilter	X	X	X	X	X
4.1.1.4	srLogMonUpdateInterval	X	X	X	X	X
4.1.1.5	srLogMonWindowSize	X	X	X	X	X

Table 161. System resources object support by platform (continued)

Sub-OID	MIB objects under OID: .1.3.6.1.4.1.1977.9	Win	Linux	Solaris	AIX	HP-UX
4.1.1.6	srLogMonEvent	X	X	X	X	X
4.1.1.7	srLogMonEventStatus	X	X	X	X	X
4.1.1.8	srLogMonEventTime	X	X	X	X	X
4.1.1.9	srLogMonCreateTime	X	X	X	X	X
4.1.1.10	srLogMonDescription	X	X	X	X	X
4.1.1.11	srLogMonOwner	X	X	X	X	X
4.1.1.12	srLogMonStatus	X	X	X	X	X
4.1.1.13	srLogMonDefaultLevel	X	X	X	X	X
4.1.1.14	srLogMonControlMode	X	X	X	X	X
4.1.1.15	srLogMonControlPosition	X	X	X	X	X
srLogMonStatsTable Objects		X	X	X	X	X
4.2.1.1	srLogMonMatches	X	X	X	X	X
4.2.1.2	srLogMonLastMatches	X	X	X	X	X
4.2.1.3	srLogMonFileSize	X	X	X	X	X
4.2.1.4	srLogMonLastUpdate	X	X	X	X	X
4.2.1.5	srLogMonLastError	X	X	X	X	X
srLogMonTable Objects		X	X	X	X	X
4.3.1.1	srLogMonIndex	X	X	X	X	X
4.3.1.2	srLogMonTime	X	X	X	X	X
4.3.1.3	srLogMonLine	X	X	X	X	X
4.3.1.4	SrLogMonLevel	X	X	X	X	X
srUserLoginTable Objects		%	X	X	X	X
5.1.1.1	srUserLoginIndex	X	X	X	X	X
5.1.1.2	srUserLoginName	X	X	X	X	X
5.1.1.3	srUserLoginDescription		X	X	X	X
5.1.1.4	srUserLoginTime	X	X	X	X	X
5.1.1.5	srUserLoginDevice	X	X	X	X	X
5.1.1.6	srUserLoginOrigin		X	X	X	X
5.1.1.7	srUserLoginTask	X	X	X	X	X
5.1.1.8	srUserLoginActiveTask	X	X	X	X	X
X = fully supported, %= partially supported, + = summed together, †=physical disks only, ‡ = volume groups only * = The same statistics are reported for each row instance						

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### Control table

The logmon commands create rows in the control table (srLogMonControlTable).

The general syntax of these commands is:

```
logmon property=value  
logmon create [property=value ...]  
logmon reset
```

Table 162 lists the properties supported in these commands.

*Table 162. Configuration command parameters - logmon*

Property	Type	Description	Sets MIB object
deflevel	enum	The default severity level assigned to each matching entry:  error  information  unknown  warning	DefaultLevel
description	string	A description of the control row.	Description
event	int	The index of the event generated when a matching log file entry is detected.	Event
eventstatus	enum	Event flow control:  alwaysready  fired  ready	EventStatus
file	string	The name of the log file to be monitored.	LogFile
filter	string	A regular expression specifying the text to be searched for in the log file.	Filter
interval	int	The interval (in seconds) at which the log file is monitored.	UpdateInterval
mode	enum	exclude  include	Mode
windowsize	int	Sets the maximum number of rows maintained in the history table for this control row.	WindowSize

---

## Examples

These examples demonstrate how to use the sysres subagent to monitor log files.

### Monitoring log file warnings

Monitor the log file sys.log located in the directory /var/log/ on a UNIX system, using the word warning as a filter expression:

```
subagent load sysres
logmon reset
logmon file=/var/log/sys.log
logmon filter=warning
logmon windowsize=60
logmon create
```

The subagent monitors the files for the filter expression warning. When a matching entry is found in a log file, a row is added to srLogMonTable, and the appropriate row in the srLogMonStatsTable is updated to reflect this match being made.

### Monitoring log file changes

Monitor the log file c:\logfile.txt on a Windows system at 1-minute intervals, generating a trap notification whenever the log file changes:

```
subagent load rmonc
event reset
event type=snmp-trap
event description="Change to c:\logfile.txt"
event create
lmevent=$?
```

```
subagent load sysres
logmon reset
logmon file=c:\logfile.txt
logmon filter=.*
logmon windowsize=60
logmon interval=60
logmon event=$lmevent
logmon eventstatus=ready
logmon deflevel=information
logmon create
```

The subagent monitors the log file c:\logfile.txt. When a new entry in the log file is detected, a row is added to srLogMonTable, and an SNMP trap is sent (if the RMON event row has been configured appropriately).

### Monitoring a Windows system log file

Monitor a Windows system log file for any changes:

1. In srLogMonControlTable, create a control row by setting srLogMonControlStatus to CreateAndWait(5).
2. Set srLogMonControlLogFile to \*SYSTEM.
3. Set srLogMonControlFilter to .\*.
4. Set srLogMonControlWindowSize to 60.

The srLogMonTable will display the 60 most recent log file matches in the Windows system log file.

5. Activate the control row by setting srLogMonControlStatus to active(1).

The subagent now monitors the Windows system log file. Whenever a new entry is made in this file, a row is added to `srLogMonTable`, and the corresponding row in the `srLogMonStatsTable` is updated.

The subagent configuration commands for creating this log file monitor are:

```
subagent load sysres
logmon reset
logmon file=*SYSTEM
logmon filter=.*
logmon windowsize=60
logmon create
```

## Monitoring access to a Web proxy

Detect unapproved use of a Web proxy by monitoring the access log file for IP addresses that are not permitted to use the Web proxy whose access log is `var/log/squid/access.log`:

```
subagent load sysres
logmon reset
logmon file=var/log/squid/access.log
logmon filter=217\.104\.227\.*\|195\.175\.0\.*
logmon mode=exclude
logmon windowsize=60
logmon create
```

The subagent monitors the access log. Whenever a new entry is made in this file which does not contain the IP address `217.104.227.x` or `195.175.0.x`, a row is added to `srLogMonTable`, and the corresponding row in the `srLogMonStatsTable` is updated.

`srLogMonControlFilter` lists the IP addresses that are permitted to use the Web proxy; for example, `217\.104\.227\.*\|195\.175\.0\.*`. Note the use of backslash characters `\` to escape the period characters.

---

## MIB module

The `sysResources` MIB is a subtree of `networkharmoni(1977)`.

The `sysResources` subtree is shown in Figure 53 on page 276.

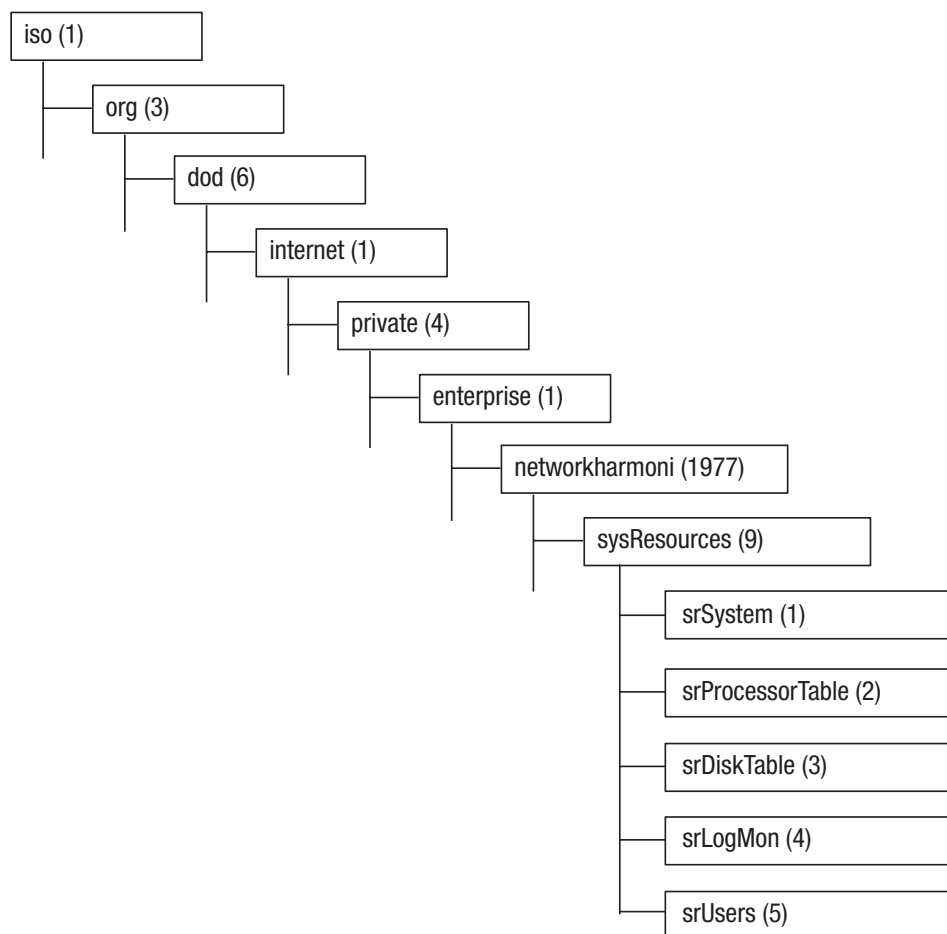


Figure 53. OID tree diagram of the sysResources MIB module

This section provides a summary of the objects defined in the sysResources MIB module. For detailed information on all objects in the module, see the sysres-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB structure

The sysResources MIB provides a standard SNMP interface.

The MIB module contains the following groups and tables:

- A system group (srSystem), which provides system performance metrics.
- A processor table (srProcessorTable), which lists the processors on the host.
- A disk table (srDiskTable), which lists the physical disk storage devices on the host.
- A log monitor group (srLogMon), for log file monitoring.
- A users group (srUsers), which contains login data for current users on the host.

## System group (srSystem)

The system group (srSystem) provides a set of objects describing the system performance of the host.

Table 163 describes the objects in this group.

Table 163. System group (srSystem)

Object	Description
ActiveMemory	The total active physical memory (in KB). Memory is active when it is being used by a process in the run queue.
ContextSwitchCount	The total number of context switches that have occurred since the system was last initialized.
CPUUsage	The total usage of all CPUs in the system over the last sysresUpdateInterval, expressed as a percentage.
CPUUsageAverage	The average of CPU usage on the system.
CPUUsageSamplePeriod	The period over which the CPU average is calculated (in seconds).
DeviceCount	The count of hardware devices on the system (equal to the number of rows in hrDeviceTable).
DiskInUse	The storage used on all hard drives on the system in MB (1048576 byte blocks).
DiskSize	The total size of all hard drives on the system in MB (1048576 byte blocks).
DiskWaitCount	The number of processes waiting on disk I/O.
FreeMem	The amount of free memory (in KB).
HardPageFaultCount	The total number of hard (major) page faults that have occurred since the system was last initialized. A page fault is 'hard' when it results in I/O. A high value indicates a busy system with insufficient memory. If it is not possible to distinguish between hard and soft page faults, the total number of page faults is reported in HardPageFaultCount and SoftPageFaultCount is zero.
InterruptCount	The total number of device interrupts that have occurred since the system was last initialized.
Load1	The system's load average for the past minute, multiplied by 100. Load average is the average number of processes that were in the run queue over a given period. If this value is -1 it means that not enough samples have been gathered yet to give a result.
Load15	The system's load average for the past 15 minutes, multiplied by 100. Load average is the average number of processes that were in the run queue over a given period. If this value is -1 it means that not enough samples have been gathered yet to give a result.
Load5	The system's load average for the past 5 minutes, multiplied by 100. Load average is the average number of processes that were in the run queue over a given period. If this value is -1 it means that not enough samples have been gathered yet to give a result.
MemoryInUse	The total physical memory in use (in KB).
OpenFileCount	The number of files currently open on the system.

Table 163. System group (srSystem) (continued)

Object	Description
PageInCount	The total number of pages that have been paged in since the system was last initialized.
PageOutCount	The total number of pages that have been paged out since the system was last initialized.
PageReclaimCount	The total number of pages that have been reclaimed (from the free list and cache list) since the system was last initialized. A high value indicates a busy system which has ample memory.
PageSwapInCount	The total number of pages that have been swapped in due to a process being moved from secondary storage since the system was last initialized.
PageSwapOutCount	The total number of pages that have been swapped out due to a process being moved to secondary storage since the system was last initialized.
PageWaitCount	The number of processes waiting on page I/O.
ProcessCount	The number of processes currently running on the system.
RunQueueLen	The current (instantaneous) length of the system run queue, excluding the process that is calculating this value.
SleepingResidentCount	The number of processes that are using physical memory but are not runnable.
SoftPageFaultCount	The total number of soft (minor) page faults that have occurred since the system was last initialized. A page fault is 'soft' when it does not result in I/O. If it is not possible to distinguish between hard and soft page faults, the total number of page faults is reported in HardPageFaultCount and SoftPageFaultCount is zero.
SwapInCount	The total number of processes that have been swapped in from secondary storage since the system was last initialized.
SwapInUse	The total swap space in use on the system (in KB).
SwapOutCount	The total number of processes that have been swapped out to secondary storage since the system was last initialized.
SwappedRunnableCount	The number of processes that are runnable but completely swapped out.
SwapPercentUsed	The percentage of the system swap in use. This is calculated approximately as $\text{SwapInUse} \times 100 / \text{TotalSwap}$ .
SyscallCount	The total number of system calls that have occurred since the system was last initialized.
TotalSwap	The total system swap space (in KB).
TrapCount	The total number of system traps that have occurred since the system was last initialized.
UpTime	The operating system uptime (in seconds).



## Processor table

The `srProcessorTable` is a conceptual table of processors contained in the host. This table provides an extension to the `hrProcessorTable` in the Host Resources (RFC-1514) MIB. The indexes used in this table correspond to the relative processor index in the `hrProcessorTable`.

Table 164 describes the row objects in `srProcessorTable`.

*Table 164. Processor table (srProcessorTable)*

Row object	Description
Idle	The total number of system ticks that the processor has spent idle.
IdlePercent	The idle time expressed as a percentage of total time over the <code>sysresUpdateInterval</code> .
Index	References the corresponding index in <code>hrProcessorTable</code> [RFC-1514].
LastUpdate	The time (relative to <code>sysUpTime</code> ) when the last sample took place.
Sys	The total number of system ticks that the processor has spent executing kernel code.
SysPercent	The kernel time expressed as a percentage of total time over the <code>sysresUpdateInterval</code> .
User	The total number of system ticks that the processor has spent executing user code.
UserPercent	The user time expressed as a percentage of total time over the <code>sysresUpdateInterval</code> .
Wait	The total number of system ticks that the processor has spent in a waiting state.
WaitPercent	The wait time expressed as a percentage of total time over the <code>sysresUpdateInterval</code> .

## Disk table

The `srDiskTable` is a conceptual table of physical disk storage devices on the host. This table provides an extension to the `hrDiskStorageTable` in the Host Resources (RFC-1514) MIB. The indexes used in this table correspond to the relative device index in `hrDiskStorageTable`.

Table 165 describes the row objects in `srDiskTable`.

*Table 165. Disk table (srDiskTable)*

Row object	Description
AccessCount	The number of read/write operations on the device.
AccessTime	The average access time in milliseconds over the <code>sysresUpdateInterval</code> .
Index	References the corresponding index in <code>hrDiskStorageTable</code> [RFC-1514].
KB	The number of KB read from and written to the device.
LastUpdate	The time (relative to <code>sysUpTime</code> ) when the last sample took place.
QueueLength	The average number of disk operations waiting over the <code>sysresUpdateInterval</code> .
ReadCount	The number of read operations on the device.
Utilization	The percentage time that the disk was busy over the <code>sysresUpdateInterval</code> .
WriteCount	The number of write operations on the device.

## Log file monitoring

The log file monitoring (srLogMon) group provides facilities for monitoring the contents of system and application log files and triggering events based on matches against filter expressions.

The group contains three tables:

- A control table (srLogMonControlTable) for defining log file monitoring.
- A statistics table (srLogMonStatsTable), which summarizes the results of log file monitoring.
- A history table (srLogMonTable), which stores log file entries detected during log file monitoring.

### Control table

The control table provides control rows for setting up log file monitoring. Each control row defines a monitor for one log file, whose text content is checked for a specific expression at regular intervals. If this expression is found, the log file entry containing the expression is written to the history table. Table 166 describes the row objects in srLogMonControlTable.

Table 166. Control table (srLogMonControlTable)

Row object	Description
CreateTime	The value of sysUpTime when the control row event was last activated.
DataControl	Controls the monitoring status of the control row:  on(1) - Monitoring is enabled  off(2) - Monitoring is disabled
DefaultLevel	Specifies a default severity level that is assigned to each history row created for this control row. This value is only assigned if the agent is unable to determine a severity level from the log file entry itself and would otherwise assign it the value unknown(1).  Default value - unknown(1).
Description	A description of the control row (that is, the purpose of the log file monitor).
Event	The event fired if an entry matching the Filter expression is found in the monitored log file. Set this variable to an RMON event index configured to send traps. A value of 0 indicates that no event is fired.
EventStatus	RMON-standard event control. See Appendix B, "RMON1 MIB group," on page 533 for details.
EventTime	The value of sysUpTime when the event was last fired.
Filter	A regular expression (often called the filter expression or match expression) specifying the information to be searched for in the log file monitored. Whenever the subagent finds a log file entry matching this expression, it writes that log file entry to the history table, srLogMonTable. To match all entries in a log file, use the expression .*.
LogFile	The absolute path name of the log file monitored.

Table 166. Control table (srLogMonControlTable) (continued)

Row object	Description
Mode	<p>Sets the event generation behavior:</p> <p>include(1) indicates that an event is generated if the new log file entry matches the Filter expression</p> <p>exclude(2) indicates that an event is generated if the new log file entry does not match the Filter expression.</p> <p>Default value - include(1)</p>
Position	<p>Sets the position in the log file at which the agent begins monitoring upon activation of the control row: A value of 0 indicates the end of the log file. A negative value n indicates a location n bytes from the end of the log file. A positive value n indicates a location n bytes from the start of the log file. If the value indicates a location that lies outside the log file (that is, if the absolute value is greater than the log file's size in bytes), the agent will commence processing at the end of the log file. After monitoring has commenced, the agent uses this object to store the location in the log file from which it will continue checking during the next sample.</p>
UpdateInterval	<p>Sets the interval (in seconds) at which the monitored log file is checked. The default value is 60 seconds.</p>
WindowSize	<p>Sets the maximum number of rows maintained in the history table srLogMonTable for this control row. The minimum value of this variable is 1. The default value is 1.</p>

## Statistics table

The statistics table contains summary information about the results generated by log file monitors defined in the control table. Statistics table rows map to control table rows via the srLogMonControlIndex variable. Table 167 describes the row objects in srLogMonStatsTable.

Table 167. Statistics table (srLogMonStatsTable)

Row object	Description
FileSize	The size of the log file (in bytes) when the most recent matching log file entry was found.
LastError	The most recent error message reported by the agent when attempting to access the log file.
LastMatch	The value of sysUpTime when the most recent matching log file entry was found.
LastUpdate	The value of sysUpTime when the log file was most recently checked by the agent.
Matches	The number of entries in the monitored log file that match the expression defined in srLogMonControlFilter. This value represents the number of matches found since monitoring started.

## History table

The history table contains information about log file entries detected by control rows. Each history row contains an entry from the log file that matches the filter expression defined in a control row. History table rows map to control table rows via the srLogMonControlIndex variable. Summary information about the history

rows generated by a control row is stored in the statistics table. The number of history rows allocated to a monitored log file is defined in the corresponding control row's `srLogMonControlWindowSize` variable. If the number of rows stored for a log file exceeds this number, the oldest history row is removed each time a new row is created. Table 168 describes the row objects in `srLogMonTable`.

*Table 168. History table (srLogMonTable)*

Row object	Description
Level	The severity level of the log file entry, if it can be determined. Generally, the <code>srLogMon</code> subagent can only determine severity levels from Windows application and system log files.
Line	A log file entry that contains the <code>srLogMonControlFilter</code> expression.
Time	The value of <code>sysUpTime</code> when the history row was added.

## User table

The users table (`srUserLoginTable`) provides a list of users currently logged on to the host system.

Table 169 describes the row objects in `srUserLoginTable`.

*Table 169. User table (srUserLoginTable)*

Row object	Description
ActiveTask	The value of <code>hrSWRunIndex</code> to which this login's current task/process corresponds. Zero if unknown.
Description	A description of the user; usually the full name and any other relevant information.
Device	The device on which the user is logged on.
Index	References the corresponding index in <code>hrProcessorTable</code> [RFC-1514].
Name	The account name (username) used to log on to the machine.
Origin	A description of where the user is logged on from (for example, the host name).
Task	The value of <code>hrSWRunIndex</code> to which this login's task/process corresponds. Zero if unknown.
Time	The value of <code>sysUpTime</code> when this login began. If the login began before the agent booted, then this value is zero.

## Notification types

The `sysResources` MIB module defines notification types for events generated by the subagent.

These types are shown in Figure 54 on page 283 and listed in Table 170 on page 283.

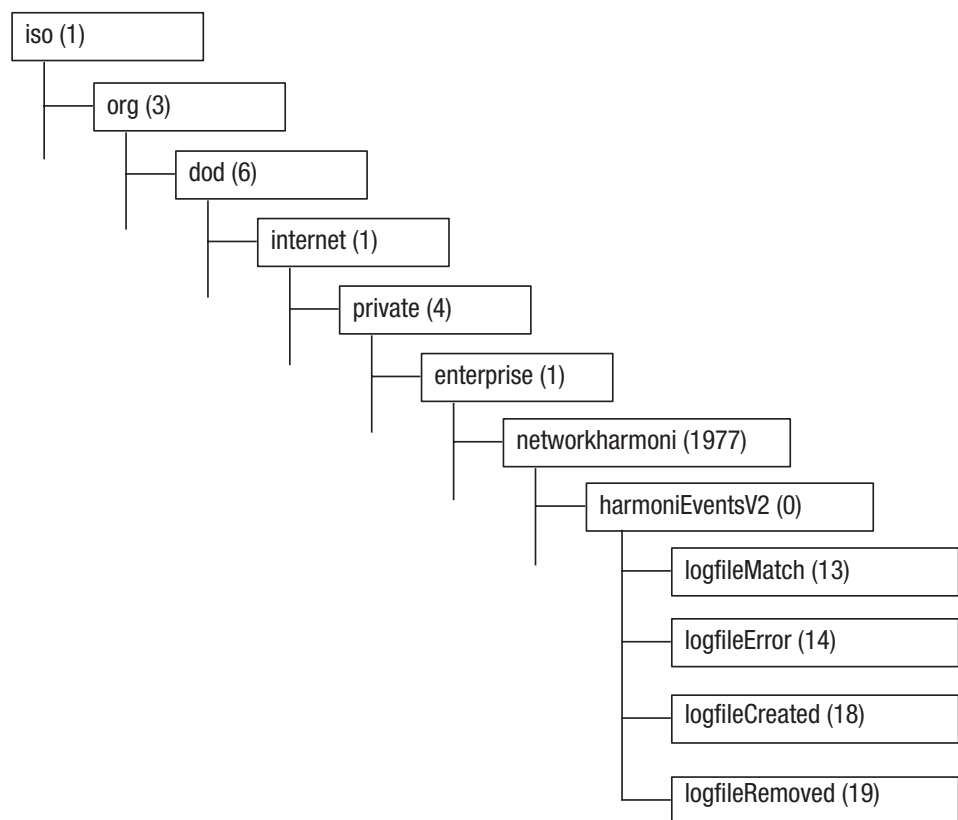


Figure 54. OID tree diagram for system resources notification types

Table 170. Notification types for system resources

Notification type	Description
lofFileRemoved	Generated when a log file being monitored can no longer be opened.  Variable bindings: srLogMonControlLogFile srLogMonControlDescription
logFileCreated	Generated when a file being monitored is created.  Variable bindings: srLogMonControlLogFile srLogMonControlDescription
logFileError	Generated when an attempt by the subagent to read a monitored log file fails.  Variable bindings: srLogMonStatsLastError srLogMonControlDescription

Table 170. Notification types for system resources (continued)

Notification type	Description
logFileMatch	<p>Generated when an entry in a monitored log file matches the expression defined in <code>srLogMonControlFilter</code>.</p> <p>Variable bindings:</p> <p><code>srLogMonControlLogFile</code></p> <p><code>srLogMonLine</code></p> <p><code>srLogMonLevel</code></p> <p><code>srLogMonControlDescription</code></p>

---

## Chapter 28. Timer

The timer subagent MIB module provide a mechanism for generating events at a scheduled point in time.

The subagent enables you to schedule the activation or deactivation of control rows in other MIBs. The scheduling facility supports a range of intervals:

- 5-, 10-, 30- and 60-minute
- day-of-week
- day-of-month
- monthly
- yearly

---

### Component files

The timer subagent MIB module are comprised of a set of component files.

Table 171 list the timer subagent and MIB module component files and their installed locations.

*Table 171. timer component files*

File	Location	Description
timer.dll (Windows)	bin	Binary implementation of the timer subagent.
libtimer.so/.sl (UNIX)		
timer-mib.mib	mibs	timer MIB definition document.
timer.oid	config/oid	timer subagent object identifier file.

---

### Guidelines

Use the timer MIB to schedule an event for some future point in time.

#### Before you begin

Ensure that the timer subagent is loaded. To load the subagent, use the command:  
subagent load timer

#### Procedure

To schedule an event:

1. Create a control row in timerControlTable by setting timerControlStatus to createAndWait(5).
2. Set the event's scheduled date and time using the minute, hour, day, month and year control row objects.
3. Specify the event to be generated by setting timerControlEventIndex to the RMON IfIndex value of the required event.
4. Activate the control row by setting timerControlStatus to active(1).

## Results

By default, the value of each time or date object is undefined or -1. Any object that has either of these values does not affect the schedule. That is, if you wish to specify a schedule independent of a particular date or time object, leave that object set to its default value. For example, to specify a schedule that occurs independent of the actual month, leave the `timerControlMonth` object set to its default value undefined or set it to -1.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The `timer` command create rows in the control table (`timerControlTable`).

The general syntax of these commands is:

```
timer property=value
timer create [property=value ...]
timer reset
```

Table 172 lists the properties supported in these commands.

*Table 172. Configuration command parameters - timer*

Property	Type	Description	Sets MIB object
day	int	The day unit of the schedule time.	Day
dayofweek	int	The day of week unit of the schedule time.  friday  monday  saturday  sunday  thursday  tuesday  undefined  wednesday	DayOfWeek
description	string	A description of the control row.	Description
event	int	The index of the event generated at the scheduled point in time.	EventIndex



Table 172. Configuration command parameters - timer (continued)

Property	Type	Description	Sets MIB object
eventstatus	enum	Event flow control:  alwaysready  fired  ready	EventStatus
hour	int	The hours unit of the schedule time.	Hour
minute10	enum	The minutes unit of the schedule time modulo 10.	Minute10
minute30	enum	The minutes unit of the schedule time modulo 30.	Minute30
minute5	string	The minutes unit of the schedule time modulo 5.	Minute5
minute60	int	The minutes unit of the schedule time modulo 60.	Minute60
month	int	The month unit of the schedule time.  april  august  december  february  january  july  june  march  may  november  october  september  undefined	Month
year	int	The year unit of the schedule time.	Year

---

## Examples

These examples demonstrate how to use the timer MIB.

### Generating an event at five-minute intervals

To generate an event every five minutes:

```
subagent load rmonc
event reset
event type=snmp-trap
event community=public
event description="Scheduled event: 5-minute intervals"
event create
schedEvent=$?
```

```
subagent load timer
timer reset
timer minute5=0
timer event=$schedEvent
timer eventstatus=3
timer create
```

### Generating events at a specific time

To generate an event every Sunday at 0400 hours:

```
subagent load rmonc
event reset
event type=snmp-trap
event community=public
event description="Scheduled event 04:00 Sundays"
event create
schedEvent=$?
```

```
subagent load timer
timer reset
timer minute60=0
timer hour=4
timer dayofweek=sunday
timer event=$schedEvent
timer eventstatus=3
timer create
```

---

## MIB module

The timer MIB is a subtree of networkharmoni(1977).

The subtree is shown in Figure 55 on page 289.

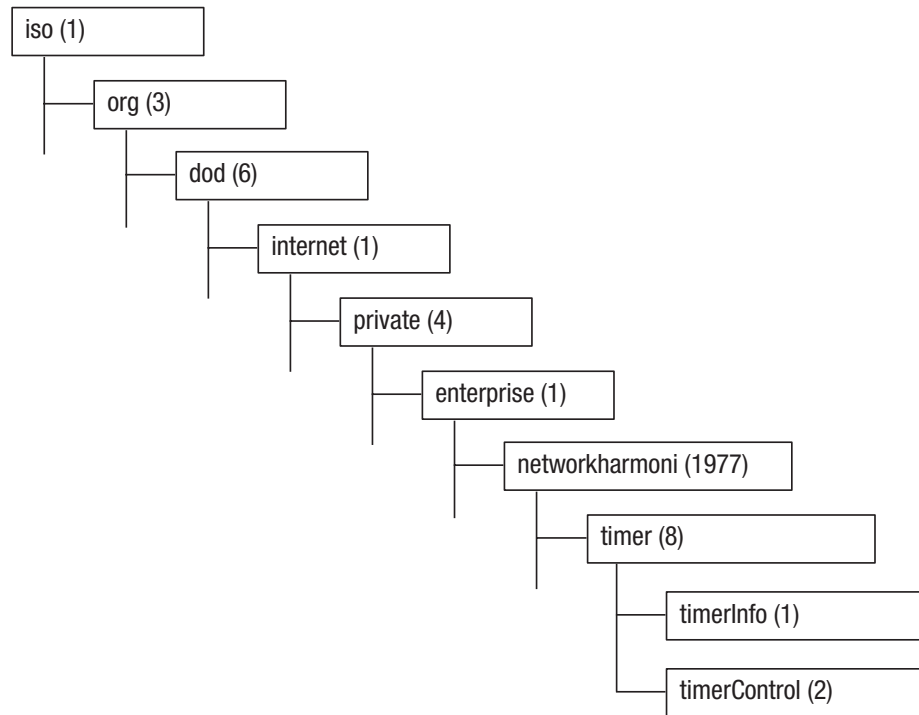


Figure 55. OID tree diagram of the timer MIB module

This section provides a summary of the objects defined in the timer MIB module. For detailed information on all objects in the module, see the `timer-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB tables

The timer MIB provides a standard SNMP interface with control row semantics.

The MIB module contains:

- An information group, `timerInfo`
- A control table, `timerControlTable`

### Information group

The information group of scalar objects (`timerInfo`) specifies the current local time of the agent. Each object represents one time unit comprising the current local time and date.

Table 173 lists the scalar objects in `timerInfo`.

Table 173. Timer information group (`timerInfo`)

Object	Description
Day	The day of month unit of the current local date.

Table 173. Timer information group (timerInfo) (continued)

Object	Description
DayOfWeek	The day of week unit of the current local date. The values are: sunday(1) monday(2) tuesday(3) wednesday(4) thursday(5) friday(6) saturday(7)
Hour	The hours unit of the current local time.
Minute	The minutes unit of the current local time.
Month	The month unit of the current local date. The values are: january(1) february(2) march(3) april(4) may(5) june(6) july(7) august(8) september(9) october(10) november(11) december(12)
Second	The seconds unit of the current local time.
Year	The year unit of the current local date.

## Control table

The timer control table (timerControlTable) defines schedules for event generation. Events are generated when the values of all objects in the row match the current time and date indicated by the timerInfo group; however, if the value of an object undefined or -1, it does not affect scheduling.

Table 174 lists the row objects in timerControlTable.

Table 174. Timer control table (timerControlTable)

Row object	Description
Day	The day unit of the schedule time.

Table 174. Timer control table (timerControlTable) (continued)

Row object	Description
DayOfWeek	The day of week unit of the schedule time. The values are:  undefined(-1)  sunday(1)  monday(2)  tuesday(3)  wednesday(4)  thursday(5)  friday(6)  saturday(7)
Description	A description of the schedule.
EventIndex	Contains the RMON eventIndex of the event generated when the local time matches the schedule time. If this index does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then no event is generated.
EventStatus	Sets the event flow control for the events associated with this row:  eventReady(1) - A single event may be generated, after which the value of this object changes to eventFired(2).  eventFired(2) - Disables event generation. No events may be generated until the object is modified to eventReady(1) or eventAlwaysReady(3).  eventAlwaysReady(3) - Disables the flow control, allowing events to be generated without restriction. Using this setting is not recommended as it can result in high network traffic or other performance problems.
Hour	The hours unit of the schedule time.
Index	Uniquely identifies this row in the control table.
Minute10	The minutes unit of the schedule time modulo 10.
Minute30	The minutes unit of the schedule time modulo 30.
Minute5	The minutes unit of the schedule time modulo 5.
Minute60	The minutes unit of the schedule time modulo 60.

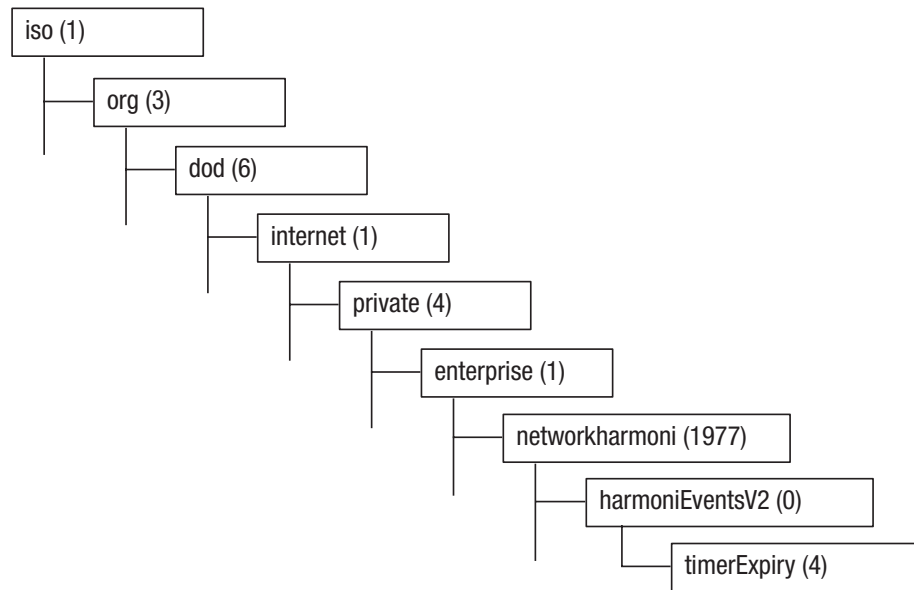
Table 174. Timer control table (timerControlTable) (continued)

Row object	Description
Month	The month unit of the schedule time. The values are:  undefined(-1)  january(1)  february(2)  march(3)  april(4)  may(5)  june(6)  july(7)  august(8)  september(9)  october(10)  november(11)  december(12)
Owner	Indicates the owner of the control row.
Status	An SNMPv2 row status object controlling creation, activation and deletion of the control row.
Year	The year unit of the schedule time.

## Notification types

The timer MIB implements the timerExpiry notification type for events generated when a timer control entry expires.

Figure 56 on page 293 shows the OID tree diagram for this notification type.



*Figure 56. OID tree diagram for timerExpiry notification type*

The timerExpiry notification type contains the following objects:

- timerInfoSecond
- timerInfoMinute
- timerInfoHour
- timerInfoDay
- timerInfoDayOfWeek
- timerInfoMonth
- timerInfoYear





---

## Chapter 29. Traceroute

The traceroute subagent performs periodic traceroutes that show the path that an application takes from the client to the server. Data is stored for future baselining and for historical reporting of routes and hop delay information.

The traceroute subagent and the associated traceroute MIB module provide facilities for performing route traces from the agent's host machine to a specified destination.

To determine the network route to a destination, the subagent sends a series of probe packets to the destination IP address. With each packet sent, it increments the time-to-live (TTL) value in the packet's header and monitors the response to the packet. This process repeats until a packet reaches the destination. The response to each packet provides information about the network hops taken along the route to the destination and the network response time.

The data gathered by the traceroute MIB includes the address of each hop involved in reaching the destination and the time delay incurred in each hop. This information is useful for evaluating network responsiveness or in mapping network paths.

---

### Component files

The traceroute subagent and MIB module are comprised of a set of component files.

Table 175 lists the traceroute subagent and MIB module component files.

*Table 175. traceroute component files*

File	Location	Description
traceroute.dll (Windows) libtraceroute.so/.sl (UNIX)	bin	Binary implementation of the traceroute subagent.
traceroute-mib.mib	mibs	traceroute MIB definition document.
traceroute.oid	config/oid	traceroute subagent object identifier file.

---

### Guidelines

To trace the route to a network destination, create a traceroute control row.

#### Before you begin

Ensure that the traceroute subagent is loaded. To load the subagent, use the command:

```
subagent load traceroute
```

## Procedure

The general procedure for creating and configuring a traceroute control row is:

1. Set `tracerouteControlStatus` to `CreateAndWait(5)`.
2. Select a trace method by setting `tracerouteControlMethod` to the value appropriate to the type of trace you wish to perform.
3. Specify the interval at which you wish the trace to be performed by setting `tracerouteControlSampleInterval`.
4. Set `tracerouteControlDestTableSize` to the number of destinations that you wish to trace.
5. Create a row in `tracerouteDestTable` for each destination that you wish to trace.
6. If you wish to store a history of the traceroute data, specify the number of samples whose data you wish to store by setting `tracerouteControlBucketsRequested` and specify the interval at which data is transferred to history table by setting `tracerouteControlHistoryInterval`.
7. Activate the control row by setting `tracerouteControlStatus` to `active(1)`.

## Trace methods and firewalls

The `tracerouteControlMethod` object selects the route tracing method.

ICMP traces may be restricted by firewalls that filter ICMP messages. Using the UDP or TCP method may enable probe packets to pass through some firewalls. In these two methods, the returned packet is still an ICMP message except for the message from the final host, which is either a UDP datagram or a TCP SYN/ACK packet.

## UDP and TCP port numbers

Using the `tracerouteDestPort` object you can specify a destination port for the UDP and TCP trace methods.

To achieve the best results:

- In UDP traces, specify a port number that corresponds to a port on the destination machine that either does not exist (that is, a port on which no application is listening) or is guaranteed to reply to the packet (for example, the UDP echo port 7).
- In TCP traces, specify a port number that corresponds to a port that does exist on the destination machine.

In either type of trace, specify a port to which the destination machine will respond. If you specify a port that the machine does not respond to, the trace will result in a timeout on the final hop and will then increment the TTL and attempt to trace the route again, and this process will repeat until the maximum TTL set by `tracerouteControlTTL` is reached.

## Packet TTL and timeout values

The `tracerouteControlTTL` and `tracerouteControlTimeout` objects set the time-to-live (TTL) of packets, and the interval between packets.

The `tracerouteControlTTL` object sets the maximum TTL value that the subagent will assign to the TTL field of a probe packet header when attempting to trace a route. Effectively, it represents the maximum number of hops that a packet may take along its route to the destination. To ensure that the trace is able to operate effectively, set this object to a value that provides probe packets with a reasonable chance of reaching the destination. The default value of this object is 30.

The `tracerouteControlTimeout` object sets the interval between each probe packet sent to the destination. Lowering the value of `tracerouteControlTimeout` decreases the overall traceroute time, but may lead to the discarding of results from intermediate hops if they are slow to respond.

## Event activation and deactivation

Traceroute control rows can be activated and deactivated by RMON events.

To configure event-based activation and deactivation, set the `tracerouteControlTurnOnEventIndex` and `tracerouteControlTurnOffEventIndex` objects to the indexes of the RMON events that you wish to activate or deactivate the control row.

When a 'turn on' event has activated a control row, subsequent occurrences of the event have no effect until the current batch of traces is complete. The 'turn off' event deactivates the control row, stopping processing of the route trace.

If the `tracerouteControlEvent` object contains a valid RMON event index, the indexed event is triggered upon completion of processing of the current batch of destination addresses.

## Creating trace destinations dynamically

You can configure traceroute control rows to dynamically add trace destination addresses to the destination table in response to events generated by certain MIBs. This facility enables you to set up route traces that are activated in response to network performance problems when they arise.

For example, you could create a monitor using the `sla` MIB to monitor the level of service provided by a group of servers and create a control row that traces the network route to a server if a service level violation occurs. The offending server's address is indicated by the event generated by the `sla` monitor.

To configure a traceroute control row to dynamically generate destination addresses, set the control row's `tracerouteControlTurnOnEventAction` object to `generateAddr(2)`. When the 'turn on' event indicated by `tracerouteControlTurnOnEventIndex` is generated, the traceroute subagent creates a row in the destination table, setting `tracerouteDestAddr` to the address derived from the 'turn on' event, then activates the control row to trace the new destination.

traceroute control rows can create destination addresses dynamically in response to events generated by the `sla` and `appFlow` MIBs.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### Control table

The tracert commands create rows in the traceroute control table (tracerouteControlTable).

The general syntax of these commands is:

```
tracert property=value
tracert create [property=value ...]
tracert reset
```

Table 176 lists the properties supported in these commands.

*Table 176. Configuration command parameters - tracert*

Property	Type	Description	Sets MIB object
buckets	int	The number of history table rows requested.	BucketsRequested
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl
event	int	The index of the event generated upon completion of a route trace.	EventIndex
history	int	Sets the interval (in seconds) at which data is moved from the data table to the history table.	HistoryInterval
method	enum	Selects the method used to trace the network route from host to destination:  icmp  tcp  udp	Method
offevent	int	The control row deactivation event index.	TurnOffEventIndex
onaction	enum	Sets the action performed when the 'turn on' event occurs:  generateAddr  normal	TurnOnEventAction
onevent	int	The control row activation event index.	TurnOnEventIndex

Table 176. Configuration command parameters - *tracert* (continued)

Property	Type	Description	Sets MIB object
oninterval	int	Sets the minimum interval (in seconds) at which non-unique destination address may be added to the destination table.	TurnOnEventInterval
process	enum	Selects the processing method for destinations:  parallel  sequential	Process
sample	int	Sets the interval (in seconds) at which the route trace is performed.	SampleInterval
timeout	int	Sets the traceroute hop timeout (in milliseconds).	Timeout
ttl	int	Sets the maximum value that the subagent will assign to the TTL field of a probe packet's header.	TTL
update	enum	Selects the method in which rows are added to the data table:  atomic  dynamic	Update

## Destination table

The `tracedst` commands create rows in the traceroute destination table (`tracerouteDestTable`) and associate them with the next row created in the configuration table using the `tracert create` command.

The general syntax of these commands is:

```
tracedst property=value
tracedst store [property=value ...]
tracedst reset
```

Table 177 lists the properties supported in these commands.

Table 177. Configuration command parameters - *tracedst*

Property	Type	Description	Sets MIB object
name	string	The destination IP address of the route traced.	Port
port	int	The target port number for TCP and UDP traces.	Addr

## Alarm table

The tracertalarm commands create rows in the traceroute alarm table (tracerouteAlarmTable).

The general syntax of these commands is:

```
tracertalarm property=value  
tracertalarm create [property=value ...]  
tracertalarm reset
```

Table 176 on page 298 lists the properties supported in these commands.

*Table 178. Configuration command parameters - tracertalarm*

Property	Type	Description	Sets MIB object
eventstatus	enum	Event flow control:  alwaysready  fired  ready	EventStatus
fallevent	int	The index of the event generated when a falling alarm is triggered.	FallingEventIndex
fallthresh	int	Sets the falling alarm threshold for the monitored variable.	FallingThreshold
riseevent	int	The index of the event generated when a rising alarm is triggered.	RisingEventIndex
risethresh	int	Sets the rising alarm threshold for the monitored variable.	RisingThreshold
startup	int	Specifies the alarm activation behavior when the status of this row is first set to valid.  falling  rising  risingOrFalling	StartupAlarm
tracertindex	string	Specifies the control row monitored.	ControlIndex
type	enum	Specifies the method used when sampling the monitored variable:  absolute  delta	SampleType
var	enum	Selects the variable monitored:  hopDelay	Variable

---

## Examples

These examples demonstrate how to use the traceroute subagent to perform simple route tracing tasks.

### Tracing an IP address

To perform a trace on the destination IP address 123.12.3.1 using default values:

```
subagent load traceroute
tracert reset
tracedst reset
tracedst store name=123.12.3.1
tracert create
```

### Trace poor response times

Monitor the response time of HTTP requests made by a host machine. If a response time greater than 2 seconds is detected, activate a trace to the target of the HTTP request, using a maximum TTL value of 50 and a timeout value of 2 seconds for probe packets in the trace.

The subagent configuration commands required to perform this task are as follows:

```
event reset
event type=snmp-trap
event community=public
event description="HTTP response time violation"
event create
slaEvent=$?
```

```
subagent load traceroute
tracert reset
tracert method=icmp
tracert ttl=50
tracert timeout=2000
tracert sample=0
tracert onevent=$slaEvent
tracert onaction=generateAddr
tracert buckets=0
tracert history=0
tracert create
```

```
subagent load sla
sla reset
sla server=0.0.0.0
sla minport=80
sla maxport=80
sla type=application
sla value=2000
sla event=$slaEvent
sla eventstatus=ready
sla datacontrol=on
sla buckets=60
sla history=5
sla description="HTTP response time"
sla create
```

---

## MIB module

The traceroute MIB is a subtree of networkharmoni (1977).

The traceroute subtree is shown in Figure 57.

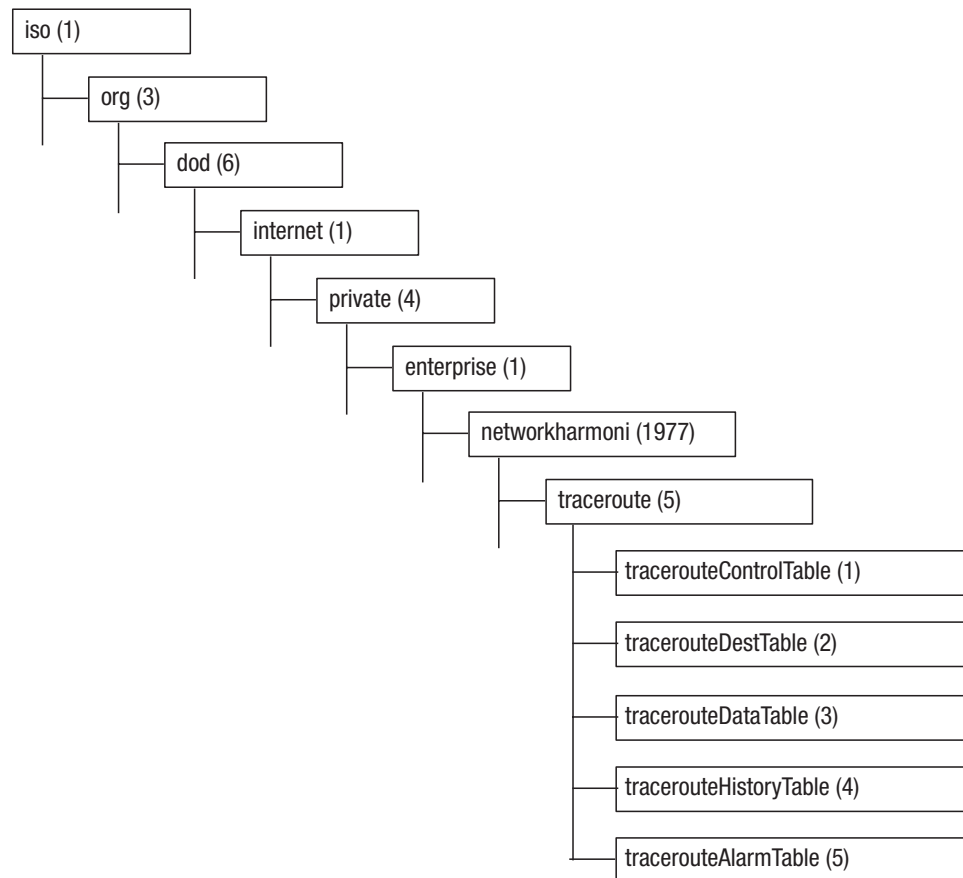


Figure 57. OID tree diagram of the traceroute MIB module

This section provides a summary of the objects defined in the traceroute MIB module. For detailed information on all objects in the module, see the `traceroute-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

### MIB tables

The traceroute MIB provides a standard SNMP interface with control row semantics.

The MIB module contains:

- A control table, `tracerouteControlTable`
- A destination table, `tracerouteDestinationTable`
- A data table, `tracerouteDataTable`
- A history table, `tracerouteHistoryTable`
- An alarm table, `tracerouteAlarmTable`



## Control table

The traceroute control table (tracerouteControlTable) defines route traces.

Each control row specifies parameters for tracing the route to a network destination. The destination of the trace is defined in a corresponding row in tracerouteDestTable. Data generated by control rows is stored in tracerouteDataTable. Table 179 lists the row objects in tracerouteControlTable.

Table 179. traceroute control table (tracerouteControlTable)

Row object	Description
BucketsGranted	Indicates the number of rows in tracerouteHistoryTable granted by the subagent to this control row.
BucketsRequested	Sets the number of rows in tracerouteHistoryTable requested for storing the data associated with this control row.
DataControl	Sets the data collection status of the control row:  on(1) - Enables data collection  off(2) - Suspends data collection
DestTableSize	Sets the number of tracerouteControlTable rows assigned to the control row.
EventIndex	Contains the RMON eventIndex of the event generated on completion of the trace for each row in tracerouteDestTable associated with this control row. If this index does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then no event is generated.
HistoryInterval	Sets the interval (in seconds) at which the information in the tracerouteDataTable row corresponding to this control row is moved to tracerouteHistoryTable. If the value of this object is 0, rows in tracerouteDataTable are kept indefinitely.
Index	Uniquely identifies the control row.
Method	Selects the method used to trace the network route from host to destination. TTL traces increment the TTL field of the probe packet's IP header until a packet reaches the destination address or until the maximum TTL value is reached. The supported trace methods are:  icmpTTL(1)- TTL trace using ICMP probe packets  tcpTTL(3) - TTL trace using TCP probe packets. TCP traces are only supported on Linux platforms  udpTTL(4) - TTL trace using UDP probe packets
Owner	Indicates the owner of the control row.
Process	Selects the method in which the destinations specified by rows in tracerouteDestTable are processed:  sequential(1) - One destination is processed one at a time  parallel(2) - Destinations are processed simultaneously, up to a maximum number of 10 destinations. Using parallel processing with a large number of destinations may lead to longer measured network delay times because the network traffic involved in parallel processing can cause network congestion and slower response times.

Table 179. *traceroute control table (tracerouteControlTable) (continued)*

Row object	Description
SampleInterval	Sets the interval (in seconds) at which the route trace is performed. The corresponding row in tracerouteDataTable is updated with the new traceroute values.
Status	An SNMPv2 row status object controlling creation, activation and deletion of the control row.
Timeout	Sets the traceroute hop timeout (in milliseconds).
TTL	Sets the maximum value that the subagent will assign to the TTL field of a probe packet's header. If a probe packet with this TTL value does not reach the destination, the subagent abandons further attempts to reach the destination.
TurnOffEventIndex	Contains the RMON eventIndex of the event configured to deactivate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then deactivation via an event is disabled.
TurnOnEventAction	<p>Sets the action performed when the 'turn on' event indicated by tracerouteControlTurnOnEventIndex occurs:</p> <p>normal(1) - Activates data collection by setting tracerouteControlDataControl to on(1)</p> <p>generateAddr(2) - Adds a destination address derived from the 'turn on' event to the tracerouteControlDestTable then activates data collection by setting tracerouteControlDataControl to on(1).</p> <p>The 'turn on' event must be generated by one of the supported MIBs. See "Guidelines" on page 295 for more details.</p>
TurnOnEventIndex	Contains the RMON eventIndex of the event configured to activate this control row. If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then activation via an event is disabled.
TurnOnEventInterval	Sets the interval (in seconds) that must expire before a new address that is a duplicate of an existing address may be added to tracerouteDestTable when tracerouteControlTurnOnEventAction is set to generateAddr(2). Duplicates that are generated before this interval has elapsed are discarded.
Update	<p>Selects the method in which rows are added to the tracerouteDataTable:</p> <p>atomic(1) - Rows are added when the entire trace is complete</p> <p>dynamic(2) - Rows are added after each of the trace's hops is complete</p>

## Destination table

The traceroute destination table (`tracerouteDestTable`) specifies destinations for route traces.

Each destination table row specifies one destination to be traced and is associated with one control row. The number of destination table rows associated with a control row is set by the control row's `tracerouteControlDestTableSize` object. Table 180 lists the row objects in `tracerouteDestTable`.

Table 180. *traceroute destination table (tracerouteDestTable)*

Row object	Description
Addr	The destination IP address of the route traced.
CompletionStatus	Indicates the status of the route trace:  pending(1) - The trace has not started  inprogress(2) - The trace is currently in progress  complete(3) - The trace is complete  resolving(4) - The destination address is being resolved  resolvefail(5) - The destination address failed to resolve  findinggw(6) - The source interface is being determined  findgwfail(7) - No interface was found
Index	Uniquely identifies the data table row.
Name	Indicates the name of the trace destination. If this object contains a name, the subagent attempts to resolve that name to an IP address. If the object contains a dotted IP address, the subagent does not attempt name resolution. If the IP address specified in <code>tracerouteDestAddr</code> corresponds to the name specified by this object, the value of this object is reset to 0.0.0.0 and when successfully resolved is updated to reflect the actual IP address used in the trace. If the domain name service returns multiple IP addresses for one name, the first address returned is used.
Port	The target port number for TCP and UDP traces.

## Data table

The traceroute data table (`tracerouteDataTable`) stores information about the hops in a route trace. Each row stores data about one hop along the route to the destination.

Table 179 on page 303 lists the row objects in `tracerouteDataTable`.

Table 181. *traceroute data table (tracerouteDataTable)*

Row object	Description
Address	The destination IP address of the hop.
Delay	The time delay (in milliseconds) between the agent and the hop destination address.
HopIndex	Uniquely identifies the data table row.

Table 181. traceroute data table (tracerouteDataTable) (continued)

Row object	Description
ICMPCode	<p>Indicates the ICMP code returned for this hop address. These codes have the same meaning as those in the ICMP protocol except where indicated below. The supported codes are:</p> <p>icmp-dest-unreach(3)</p> <p>icmp-source-quench(4)</p> <p>icmp-redirect(5)</p> <p>icmp-time-exceeded(11)</p> <p>icmp-parameterprob(12)</p> <p>icmp-timestampreply(14)</p> <p>icmp-info-reply(16)</p> <p>icmp-addressreply(17)</p> <p>udp-reply(252) - A UDP packet was received in response to a UDP probe packet.</p> <p>tcp-reset(253) - A TCP reset packet was received in response to a TCP probe packet</p> <p>tcp-connected(254) - A TCP SYN/ACK packet was received in response to a TCP probe packet</p> <p>probe-timeout(255) - No response was received before the timeout</p>
ICMPSubCode	<p>Indicates the subcode of the ICMP code returned for the hop. The subcodes are the same as those used in the ICMP protocol. The value of this object is only meaningful if the value of tracerouteDataICMPCode is an ICMP code.</p>
Time	The value of SysUpTime when the hop was traced.

## History table

The traceroute history table (tracerouteHistoryTable) contains data transferred from tracerouteDataTable.

Data is transferred at intervals set by the corresponding control row's tracerouteControlHistoryInterval object. Table 182 lists the row objects in tracerouteHistoryTable.

Table 182. traceroute history table (tracerouteHistoryTable)

Row object	Description
Address	The value of tracerouteDataAddress when the data table row was transferred.
Delay	The value of tracerouteDataDelay when the data table row was transferred.
HopIndex	Identifies the hop within the route trace.
ICMPCode	The value of tracerouteDataICMPCode when the data table row was transferred.
ICMPSubCode	The value of tracerouteDataICMPSubCode when the data table row was transferred.

Table 182. traceroute history table (tracerouteHistoryTable) (continued)

Row object	Description
IntervalStart	The value of SysUpTime when the data table row was transferred.
SampleIndex	Identifies the trace sample.
Time	The value of SysUpTime when the trace was started.

## Alarm table

The traceroute alarm table (tracerouteAlarmTable) provides a facility for generating alarms based on the measured delay between the subagent's host and a destination address.

Table 183 lists the row objects in tracerouteAlarmTable.

Table 183. traceroute alarm table (tracerouteAlarmTable)

Row object	Description
ControlIndex	Specifies the control row monitored.
EventStatus	Sets the event flow control for the events associated with this row:  eventReady(1) - A single event may be generated, after which the value of this object changes to eventFired(2)  eventFired(2) - Disables event generation. No events may be generated until the object is modified to eventReady(1) or eventAlwaysReady(3)  eventAlwaysReady(3) - Disables the flow control, allowing events to be generated without restriction. Using this setting is not recommended as it can result in high network traffic or other performance problems.
FallingEventIndex	Contains the RMON eventIndex of the event generated if the sample value of the monitored object crosses the falling threshold.  If this index does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then no event is generated.
FallingThreshold	Specifies the falling threshold for the sampled value of the object specified by tracerouteAlarmVariable.  When the sampled value is less than or equal to this threshold and the value of the previous sample was greater than this threshold, the event specified by tracerouteAlarmFallingEventIndex is generated.  This event is also generated if the first sample taken after this row becomes valid is less than or equal to the rising threshold and the value of the tracerouteAlarmStartupAlarm object is equal to fallingAlarm(2) or risingOrFallingAlarm(3).  After a falling event is generated, another such event cannot be generated until the sample value has risen above this threshold and reached the rising threshold.
Index	Uniquely identifies the alarm table row.
Owner	The owner of this row.

Table 183. traceroute alarm table (tracerouteAlarmTable) (continued)

Row object	Description
RisingEventIndex	<p>Contains the RMON eventIndex of the event generated if the sample value of the monitored object crosses the rising threshold.</p> <p>If this index does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then no event is generated.</p>
RisingThreshold	<p>Specifies the rising threshold for the sampled value of the object specified by tracerouteAlarmVariable.</p> <p>When the sampled value is greater than or equal to this threshold and the value of the previous sample was less than this threshold, the event specified by tracerouteAlarmRisingEventIndex is generated.</p> <p>This event is also generated if the first sample taken after the row becomes valid is greater than or equal to the rising threshold and the value of the tracerouteAlarmStartupAlarm object is equal to risingAlarm(1) or risingOrFallingAlarm(3).</p> <p>After a rising event is generated, another such event cannot be generated until the sample value has fallen below this threshold and reached the falling threshold.</p>
SampleType	<p>Sets the way the value of the object specified by tracerouteAlarmVariable is sampled and the to compared the thresholds:</p> <p>absoluteValue(1) - The value of the object is compared directly with the thresholds</p> <p>deltaValue(2) - The value of the object at the previous sample is subtracted from the current value of the object then compared with the thresholds</p>
StartupAlarm	<p>Specifies the alarm activation behavior when the status of this row is first set to valid.</p> <p>If the first sample taken after this row becomes valid is greater than or equal to the rising threshold and the value of this object is equal to risingAlarm(1) or risingOrFallingAlarm(3), then a single rising alarm is generated.</p> <p>If the first sample taken after this row becomes valid is less than or equal to the falling threshold and the value of this object is equal to fallingAlarm(2) or risingOrFallingAlarm(3), then a single falling alarm is generated.</p>
Status	The status of this row.
Variable	<p>Selects the variable monitored:</p> <p>hopDelay(1) - Monitors the object tracerouteDataDelay.</p>

## Notification types

The traceroute MIB implements notification types for events generated by the subagent.

These types are shown in Figure 58 and listed in Table 184.

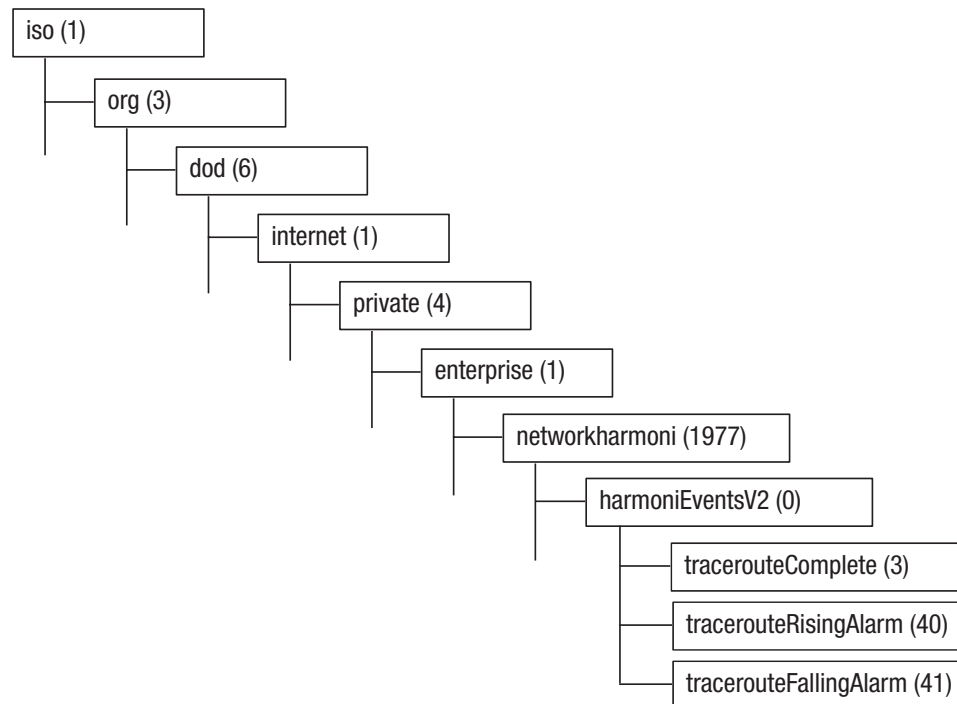


Figure 58. OID tree diagram for traceroute notification types

Table 184. traceroute notification types

Notification type	Description
traceComplete	Generated when a trace is complete.  Variable bindings:  tracerouteDestName
tracerouteFallingAlarm	Generated when an tracerouteAlarmTable entry crosses its falling threshold and generates an event that is configured for sending SNMP traps.  tracerouteAlarmVariable  tracerouteAlarmControlIndex  tracerouteDestAddr  tracerouteDataDelay  tracerouteDataAddress

Table 184. *traceroute notification types (continued)*

Notification type	Description
tracerouteRisingAlarm	<p>Generated when an alarm crosses its rising threshold and generates an event that is configured to send SNMP traps.</p> <p>Variable bindings:</p> <p>tracerouteAlarmVariable</p> <p>tracerouteAlarmControlIndex</p> <p>tracerouteDestAddr</p> <p>tracerouteDataDelay</p> <p>tracerouteDataAddress</p>



---

## Chapter 30. Transaction

The transaction subagent and the associated transaction MIB provide facilities for monitoring application transactions.

Using the subagent, new applications can be added to the list of identified applications that exists in the MIB, or existing application details modified.

---

### Component files

The transaction subagent and MIB module are comprised of a set of component files.

Table 185 lists the transaction subagent and MIB module component files and their installed locations.

*Table 185. transaction component files*

File	Location	Description
transaction.dll (Windows) libtransaction.so/.sl (UNIX)	bin	Binary implementation of the transaction subagent.
transaction-mib.mib	mibs	transaction MIB definition document.
transaction.oid	config/oid	transaction subagent object identifier file.

---

### Guidelines

To monitor transactions, create a control row in transactionControlTable.

#### Before you begin

Ensure that the transaction subagent is loaded. To load the subagent, use the command:

```
subagent load transaction
```

#### Procedure

To create a configure a transaction control row:

1. Set transactionControlStatus to CreateAndWait(5).
2. Set transactionControlClientAddress to the IP name or address of the machine acting as the client in the transactions that you wish to monitor.
3. Set transactionControlServerAddress to the IP name or address of the machine acting as the server in the transactions that you wish to monitor.
4. Set transactionControlApplication to the value of the tracerouteEnumId object in the transaction enumeration table row representing the transaction that you wish to monitor.
5. Set transactionControlTransportProtocol to the type of protocol used by the application that you wish to monitor.
6. Set the transactionControlTransportParameter1 and transactionControlTransportParameter2 to values appropriate to the type of

protocol monitored. These values usually set the minimum and maximum values of the ports used by the monitored application.

7. If you wish to maintain a history of the data gathered, set transactionControlBucketsRequested to the number rows in the history table that you wish to store and set transactionControlInterval to the interval at which you wish the history to be updated.
8. Activate the transaction control row by setting the tracerouteControlStatus to active(1).

Transaction monitoring commences and the data gathered is stored in the data table and in the history table, if so configured.

## Inivars

The transaction subagent provides inivars for configuring its operation.

These inivars are listed in Table 186.

*Table 186. transaction subagent inivars*

Inivar	Values	Description
TransactionLogFile	string	Specifies the name of the file used for transaction logging.
TransactionLogging	true   false	Enables transaction logging.
transactionSvrSide	true   false	Enables connection monitoring from the server side.  Default - false

For general information about inivars, see the *Netcool/SSM Administration Guide*.

## Application and transaction enumeration

The transaction enumeration table lists the applications and their transaction types that are available for monitoring.

Table 187 provides an example of the typical contents of transactionEnumTable.

*Table 187. Typical transactionenumerationtable setup*

TransactionID	Status	Description	TransactionStatus
0.0.0.2.0.0.0.0.0.0.0.0	active	HTTP	default
0.0.0.2.0.0.0.1.0.0.0.0	active	HTTP GET	default
0.0.0.2.0.0.0.2.0.0.0.0	active	HTTP PUT	default
0.0.0.3.0.0.0.0.0.0.0.0	active	POP3	customized
0.0.0.3.0.0.0.1.0.0.0.0	active	POP3 USER	default
0.0.0.3.0.0.0.2.0.0.0.0	notInService	POP3 PASS	customized
0.0.0.4.0.0.0.0.0.0.0.0	notInService	SMTP	customized
0.0.0.4.0.0.0.1.0.0.0.0	notInService	SMTP HELO	customized
0.0.0.5.0.0.0.0.0.0.0.0	notReady	FTP	empty
0.0.0.6.0.0.0.0.0.0.0.0	active	ORACLE	discover
0.0.0.6.0.0.0.1.0.0.0.0	active	ORACLE SELECT	customized
0.0.0.7.0.0.0.0.0.0.0.0	notReady	PEOPLESOFT	empty

Table 187. Typical transactionenumerationtable setup (continued)

TransactionID	Status	Description	TransactionStatus
0.0.0.8.0.0.0.0.0.0.0	active	DNS	default
0.0.0.8.0.0.0.1.0.0.0	active	DNS A	default

## Time-outs

The transactionControlRequestTimeout object determines the length of time over which the subagent monitors an idle data connection before determining it to be inactive.

**Note:** Large values of transactionControlRequestTimeout may exhaust host resources. Small values may lead the subagent to cease monitoring a transaction before the host has responded.

## Transaction logging

You can configure the subagent to log details about each transaction. By default, it logs these details in the file trans.log, which is located in the Netcool/SSM log directory, however you can specify a different file using the TransactionLogFile inivar.

### About this task

The maximum allowed size of the transaction log file is one MB. When the size of the file reaches this limit, the subagent creates a log file in which it logs subsequent transactions and renames the old file to <log name>.x.old, where <log name> is the name of the original log file and x is initially 1 and increments with each occurrence.

To enable transaction logging, set the TransactionLogging inivar to true.

## Minimizing packet loss

If the transactions being monitored are large and bursty, it is possible that packets will be dropped. The transaction subagent contains built-in redundancy to address this problem; however, certain packets, such as packets signalling end of connection, are essential in correctly monitoring specific transactions. If such packets are dropped, this can result in incorrect statistics being reported.

To minimize the likelihood of packets being dropped, configure the control rows as specifically as possible, using exact client and server addresses, and specific application, transport protocol, and port range settings. Leaving a control row to monitor all applications over IP between all hosts could potentially overload the agent and result in dropped packets.

If you wish to monitor more than one application, create a control row for each application, rather than using just one control row to monitor them all. If you configure the control rows carefully, the subagent is able to filter out unnecessary traffic and spend more time collecting the traffic that is required for monitoring.

## Monitoring clients and servers on the same host

If the client and server engaging in a transaction are on the same host, network transactions may not be generated and the subagent will not detect or monitor the transactions.

### About this task

## Monitoring Microsoft Exchange transactions

To monitor Microsoft Exchange transactions on Windows platforms, create and activate the control row before starting Outlook on the host being monitored.

### About this task

## SNMPv1 compatibility

The transaction MIB module uses the Counter64 data type, so it is not compatible with SNMPv1. Only use it with SNMPv2 and later.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The transaction commands create rows in the control table (transactionControlTable).

The general syntax of these commands is:

```
transaction property=value
transaction create [property=value ...]
transaction reset
```

Table 188 lists the properties supported in these commands.

*Table 188. Configuration command parameters - transaction*

Property	Type	Description	Sets MIB object
application	string (hex)	The application or transaction to be monitored.	Application
buckets	int	The number of history table rows requested.	BucketsRequested
client	string	The IP name or address of the monitored client.	ClientAddr
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl

Table 188. Configuration command parameters - transaction (continued)

Property	Type	Description	Sets MIB object
datasource	OID	The interface monitored by the control row.	DataSource
description	string	A description of the control row.	Description
history	int	The interval (in seconds) at which data is transferred from the data table to the history table.	HistInterval
maxport	int	The maximum port of the monitored protocol.	TransportParameter2
minport	int	The minimum port of the monitored protocol.	TransportParameter1
offevent	int	The index of the control row deactivation event.	TurnOffEventIndex
onevent	int	The index of the control row deactivation event.	TurnOnEventIndex
server	string	The IP name or address of the monitored server.	ServerAddr
timeout	int	The connection timeout interval.	RequestTimeout
transport	int	The transport protocol of traffic to be monitored.	TransportProtocol

## Examples

These examples demonstrate how to use the transaction MIB to perform simple transaction monitoring tasks.

### Monitoring Microsoft Exchange transactions

Monitor transactions between the host machine and a Microsoft Exchange server.

```
subagent load transaction
transaction reset
transaction server=exchangeServerName
transaction application="0 0 0 0D 0 0 0 0 0 0 0"
transaction create
```

### Monitoring Web proxy transactions

Monitor traffic between a Web proxy and any client.

```
subagent load transaction
transaction reset
transaction client=*
transaction server=webProxyName
transaction application="0 0 0 2 0 0 0 0 0 0 0"
transaction transport=2
transaction minport=80
transaction maxport=80
transaction create
```

### Monitoring Lotus Notes transactions

Monitor the traffic on a Lotus Notes server daily during the period 8:00AM to 6:00PM.

```

subagent load rmonc
event reset
event type=snmp-trap
event community=public
event description="Lotus Notes monitor on"
event create
lotusOnEvent=$?

event description="Lotus Notes monitor off"
event create
lotusOffEvent=$?

subagent load transaction
transaction reset
transaction datacontrol=off
transaction client=*
transaction server=LotusNotesServerName
transaction application="0 0 0 0C 0 0 0 0 0 0 0"
transaction transport=2
transaction minport=1352
transaction maxport=1352
transaction onevent=$lotusOnEvent
transaction offevent=$lotusOffEvent
transaction create

subagent load timer
timer reset
timer minute60=0
timer hour=8
timer event=$lotusOnEvent
timer eventstatus=alwaysready
timer create
timer reset
timer minute60=0
timer hour=18
timer event=$lotusOffEvent
timer eventstatus=3
timer create

```

---

## MIB module

The transaction MIB is a subtree of networkharmoni(1977).

The subtree is shown in Figure 59 on page 317.

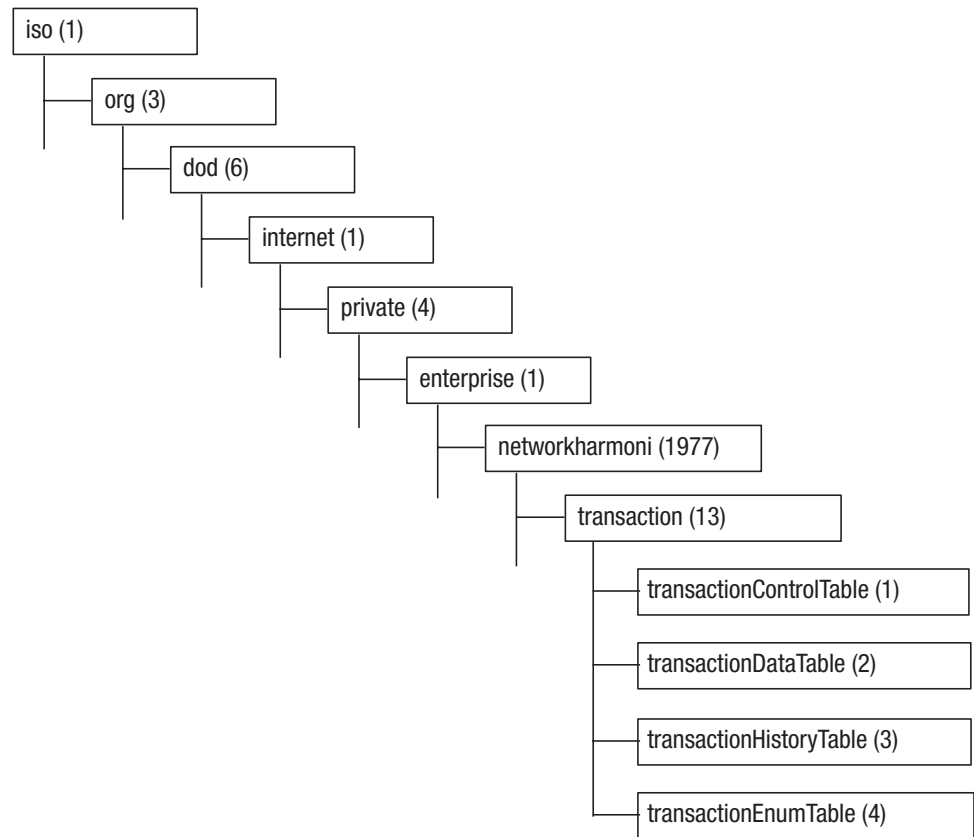


Figure 59. OID tree diagram of the transaction MIB module

This section provides a summary of the objects defined in the agent MIB module. For detailed information on all objects in the module, see the `transaction-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB tables

The transaction MIB provides a standard SNMP interface with control row semantics.

The MIB module contains the following items:

- A control table, `transactionControlTable`
- A data table, `transactionDataTable`
- A history table, `transactionHistoryTable`
- An enumeration table, `transactionEnumTable`
- A resolve names table, `transactionResolveNames`
- Several scalar objects

## Control table

The control table (transactionControlTable) defines transaction monitors. Control rows monitor an application or a specific application transaction listed in the transaction enumeration table. The statistics that the control row generates are stored in the data and history tables.

Table 189 describes the row objects in transactionControlTable.

Table 189. Transaction control table (transactionControlTable)

Row object	Description
Application	Specifies the application or transaction to be monitored. The value of this object refers to a row in transactionEnumTable, identified by transaction ID.
BucketsGranted	Indicates the number of rows in transactionHistoryTable granted by the subagent to this control row.
BucketsRequested	Sets the number of rows in transactionHistoryTable requested for storing the data associated with this control row.
ClientAddr	Sets the IP name or address of the client monitored. When this object is set to the wildcard character (*) then data is collected for all clients. Network numbers are not recognized, but if a name is given that resolves to multiple network addresses, data will be collected for any of those addresses.
CreateTime	Stores the value of sysUpTime when this entry was last activated.
DataControl	Sets the data collection status of the control row: on(1) - Enables data collection.off(2) - Suspends data collection.
DataSource	Selects the data source you wish to monitor. It is specified as the full OID to an entry in the interfaces table within MIB-2. The data source will have the format: .1.3.6.1.2.1.2.2.1.1.ifIndex value.
Description	A description of the application flow being monitored containing details such as protocol and application.
HistInterval	Specifies the interval (in seconds) at which data is transferred from transactionDataTable to transactionHistoryTable. If the value of this object is 0, data is stored in transactionDataTable indefinitely.
Index	Uniquely identifies this table row.
Owner	Indicates the owner of the control row.
RequestTimeout	Defines the timeout period (in seconds) for monitored transaction requests. If a connection does not receive any data for the period of time defined by this object, the connection is no longer monitored.  Default value: 259200 (3 days).
ServerAddr	Sets the IP name or address of the server monitored. When this object is set to the wildcard character (*) then data is collected for all servers. Network numbers are not recognized, but if a name is given that resolves to multiple network addresses, data will be collected for any of those addresses.
Status	An SNMPv2 row status object controlling creation, activation and deletion of the control row.



Table 189. Transaction control table (transactionControlTable) (continued)

Row object	Description
TransportParameter1	Defines a parameter specific to the transport protocol monitored.  For TCP/UDP/IP protocols, it sets the minimum port number for which data is collected.
TransportParameter2	Defines a parameter specific to the transport protocol monitored.  For TCP/UDP/IP protocols, it sets the maximum port number for which data is collected.
TransportProtocol	Specifies the transport protocol of traffic to be monitored:  tcp(2)  udp(3)  ip(4) - Both TCP and UDP
TurnOffEventIndex	Contains the RMON eventIndex of the event configured to deactivate this control row.  If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then deactivation via an event is disabled.
TurnOnEventIndex	Contains the RMON eventIndex of the event configured to activate this control row.  If the value of this object does not correspond to an entry in the RMON eventTable or if the value of this object is 0 then activation via an event is disabled.

## Data table

The data table (transactionDataTable) stores statistics about the network transactions monitored by a control row.

Table 190 lists the row objects in transactionDataTable.

Table 190. Transaction data table (transactionDataTable)

Row object	Description
ClientAddress	The network address of the application's client.
ClientTimestat	Indicates the duration of client processing (in milliseconds).
NetworkRTTstat	Indicates the network round-trip time (in milliseconds).
ResponseNetworkTimestat	Indicates the amount of time (in milliseconds) that the server's response spends on the network.
ServerAddress	The network address of the application's server.
ServerResponseTimestat	Indicates the time (in milliseconds) between the client's transmission of the initial request and the client's receipt of the first response packet.
ServerTimestat	Indicates the time (in milliseconds) between the server's receipt of the initial request and transmission of its first response packet.

Table 190. Transaction data table (transactionDataTable) (continued)

Row object	Description
<p>Statistics about each monitored attribute are provided in 3 forms. The name of each statistic has the format:&lt;attribute&gt;&lt;statistic&gt; formed from the attribute's generic name followed by one of the statistic types:</p> <p>Avg - the average value of the selected statistic across all monitored transactions</p> <p>Sum - the total value of the selected statistic for all monitored transactions</p> <p>SumSqr - the sum-of-squares value of the selected statistic for all monitored transactions.</p> <p>The generic names of each object are listed below.</p>	
TransactionDescription	A description of the transaction.
TransactionId	<p>Uniquely identifies a transaction type between a client/server pair of a certain application.</p> <p>This object contains 12 octets. The first 4 octets identify an application. The second 4 octets identify an application transaction. The third 4 octets may be used for further clarification depending upon the application.</p>
TransactionTimeStat	Indicates the duration of a transaction (in milliseconds).
TransportProtocol	<p>Indicates the transport protocol that the application is delivered on:</p> <p>tcp(2)</p> <p>udp(3)</p>

## History table

The history table (transactionHistoryTable) contains historical transaction data transferred from the data table.

## Enumeration table

The transaction enumeration table (transactionEnumTable) specifies a list of applications and their transaction types. Each row represents one application or a type of transaction performed by an application. To select the type of application and transaction monitored, control rows refer to rows in the transaction enumeration table.

Table 191 lists the row objects in transactionEnumTable.

Table 191. Transaction enumeration table (transactionEnumTable)

Row object	Description
Description	A description of the transaction.
Id	<p>Uniquely identifies an application or an application transaction type.</p> <p>This object contains 12 octets. The first 4 octets identify an application. The second 4 octets identify an application transaction. The third 4 octets may be used for further clarification depending upon the application.</p>
Status	An SNMPv2 row status object controlling creation, activation and deletion of the control row.

Table 191. Transaction enumeration table (*transactionEnumTable*) (continued)

Row object	Description
TransactionStatus	<p>Indicates the current status of the transaction itself:</p> <p>default(1) - Restores settings to default</p> <p>customized(2) - The application or transaction has been modified</p> <p>empty(3) - Empties an application of all transactions</p> <p>discover(4) - Enables transaction discovery for an application</p>

## Scalar objects

The transaction MIB includes two scalar objects that control aspects of the MIB's behavior.

Table 192 lists these objects.

Table 192. transaction MIB scalar objects

Object	Description
transactionFlushHistory	<p>Indicates the status of transactionHistoryTable and activates flushing of transactionHistoryTable:</p> <p>collecting(1) - Indicates that transactionHistoryTable is currently collecting data</p> <p>flushing(2) - Flushes all entries in transactionHistoryTable</p>
transactionResolveNames	<p>Controls row creation in nameResolutionTable for each instance of the transactionDataClientAddress and transactionServerAddress objects contained in transactionDataTable:</p> <p>on(1) - nameResolutionTable rows are created</p> <p>off(2) - nameResolutionTable rows are not created</p>



---

## Chapter 31. Netcool/ASM for Microsoft Exchange

The Netcool/ASM for Microsoft Exchange monitors both the configuration and performance aspects of Exchange servers, providing metrics such as memory and CPU usage, page faults, SMTP queue length, POP3 queue length, and cache hit ratios. It can automatically generate alarms based on the value of particular performance metrics.

The Netcool/ASM for Microsoft Exchange consists of the msxmon subagent and the msexchange MIB module. It operates on Windows platforms and monitors Microsoft Exchange servers.

---

### Component files

The msxmon subagent and the msexchange MIB module are comprised of a set of component files.

Table 193 lists the msxmon subagent and msexchange MIB module component files and their installed locations.

*Table 193. Netcool/ASM for Microsoft Exchange component files*

File	Location	Description
msxmon.dll	bin	Binary implementation.
exchange-mib.mib	mibs	MIB definition document.
msxmon.oid	config/oid	Object identifier file.
msxmon.cfg	config	Sample configuration file.

---

### Guidelines

Use the subagent to gather metrics from a Microsoft Exchange server for use in reporting or threshold monitoring. You can use the predefined threshold monitors supplied with the subagent, or poll the MIB to extract data for custom threshold monitoring or reporting.

The subagent requires minimal configuration; it locates Exchange servers using a set of specific methods then gathers data from the server to populate the MIB tables.

To load the subagent, use the command:

```
subagent load msxmon
```

## Target server names

The msxmon subagent locates Microsoft Exchange servers by name.

The subagent identifies the target Microsoft Exchange server using the following names, in order of priority:

1. The Microsoft Exchange server name indicated by the ExchangeServerName inivar, if the inivar is set.
2. The name of the cluster to which the machine running the subagent belongs.
3. The host name of the machine on which the subagent is running.
4. The name defined in the Windows registry key HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\MSExchangeIS that matches a name returned by an LDAP search for all Microsoft Exchange servers.

## Microsoft Exchange Server 5.5 configuration

On Microsoft Exchange 5.5 servers, to ensure that the correct data is available for the information store configuration group, msxConfIS, you must run the Microsoft Exchange Server Performance Optimizer (*exchange\_install\_dir\bin\Perfwiz.exe*) on the target Exchange server before you start the server and load the msxmon subagent.

### About this task

## Inivars

The msxmon subagent provides inivars that configure its operation.

These inivars are listed in Table 194.

Table 194. msxmon subagent inivars

Inivar	Type	Description
ExchangeServerName	string	Specifies the name of the Microsoft Exchange server to be monitored.
msxmonMonitorServices	enum	When this variable is set to true, the msexchange subagent creates NTSCM control rows for monitoring the Microsoft Exchange services:  true  false
msxmonSampleInterval	enum	Sets the interval (in seconds) at which the Microsoft Exchange performance data is updated:  true  false

For general information about inivars, see the *Netcool/SSM Administration Guide*.

## Events

The msxmon subagent does not generate events; however, you can set up the Netcool/SSM genalarm subagent to monitor thresholds for objects in the exchange MIB using the exchange.cfg file.

The exchange.cfg file specifies a default set of objects and thresholds for the exchange MIB that can be monitored by the genalarm subagent. For each object specified in the file, a row is created automatically in the genAlarm MIB. Table 195 lists the msxPerf objects that are specified in the exchange.cfg file with a description of the monitored metric.

*Table 195. Monitored MS Exchange objects*

Object	Metric monitored
ComponentPageFaults	Exchange Performance: Component page faults
ComponentPercentCPU	Exchange Performance: Component CPU usage
ComponentPercentMemory	Exchange Performance: Component memory usage
ComponentPID.?	Exchange Performance: Component PID
DSCacheHitRatio	Exchange Performance: DS Cache Hit Rate
ExchangePercentCPU	Exchange Performance: CPU usage
ExchangePercentMemory	Exchange Performance: Memory usage
MTAWorkQLen	Exchange Performance: MTA Work Queue Length
POP3CmdFailures.*	Exchange Performance: POP3 command failures
POP3CmdFailures.4.80.65.83.83	Exchange Performance: POP3 'PASS' command failures
POP3CmdFailures.4.85.83.69.82	Exchange Performance: POP3 'USER' command failures
POP3CurrentCons.0	Exchange Performance: POP3 open connections
POP3FailedCons.0	Exchange Performance: POP3 unsuccessful connections
POP3InvalidCmds.0	Exchange Performance: POP3 invalid commands
POP3RejectedCons.0	Exchange Performance: POP3 rejected connections
POP3TotalCons.0	Exchange Performance: POP3 connections
SMTPLocQLen.0	Exchange Performance: SMTP local delivery queue length
SMTPLocRetryQLen.0	Exchange Performance: SMTP local retry queue length
SMTPRemQLen.0	Exchange Performance: SMTP remote delivery queue length
SMTPRemRetryQLen.0	Exchange Performance: SMTP remote retry queue length

**Note:** You can edit the exchange.cfg file to set up thresholds that are suitable for your system.

To use the exchange.cfg file to create entries in the genalarm subagent and monitor thresholds in the exchange MIB, create a row in the agentConfigTable for the exchange.cfg file. When the row is activated, the exchange.cfg file is executed. To ensure the exchange.cfg file has been executed and view the objects being monitored, check that the appropriate rows have been created in the genAlarm MIB.

**Note:** To ensure the `exchange.cfg` file is loaded when the agent is instructed to restore values, save the `agentConfigTable` state to the `agent.state` file.

---

## MIB module

The `msexchange` MIB is a subtree of `networkharmoni` (1977).

The subtree is shown in Figure 60.

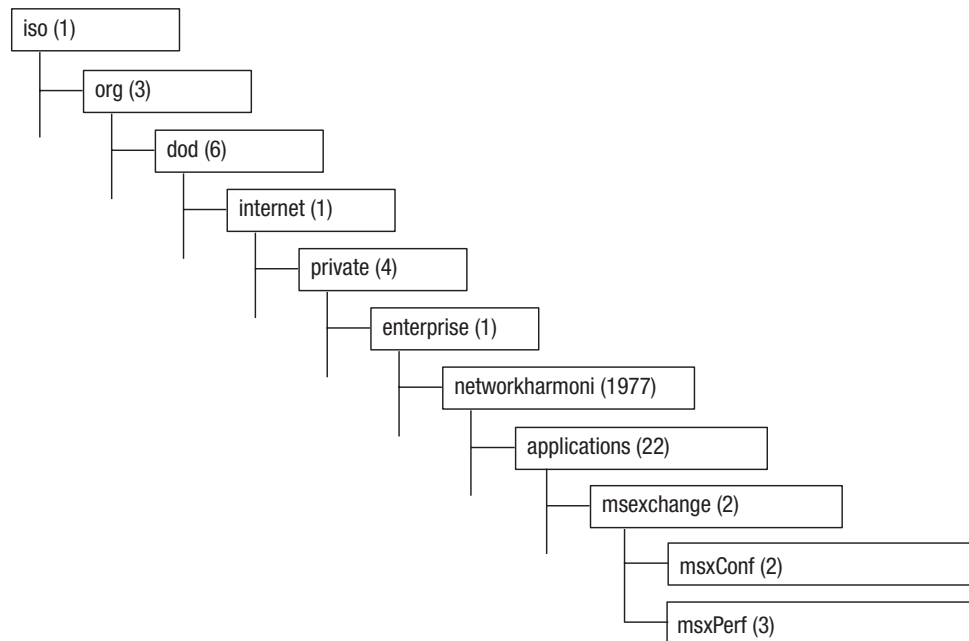


Figure 60. OID tree diagram of the `msexchange` MIB module

This section provides a summary of the objects defined in the `msexchange` MIB module. For detailed information on all objects in the module, see the `exchange-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB objects

The `msexchange` MIB provides data in the form of tables and scalar objects. It does not provide control objects.

There are two groups of objects:

- A configuration group, `msxConf`
- A performance monitoring group, `msxPerf`



## Configuration group

The configuration group ( `msxConf`) provides static configuration information about Microsoft Exchange Server installations.

## Exchange configuration

The Exchange configuration group, `msxConfExchange`, provides a set of objects describing the basic details about the Exchange Server installation. Table 196 describes these objects.

*Table 196. Exchange configuration group (msxConfExchange)*

Object	Description
Build	The build number of the Exchange installation.
InstallPath	The root directory of the Exchange installation.
ServicePack	The number of the service pack most recently applied.
Version	The version number of the Exchange installation:  55 - Exchange 5.5  60 - Exchange 2000

## Component configuration

The configuration component table, `msxConfComponentTable`, provides details of installed components for all the Exchange services currently running on a host. Each row in the table represents a distinguishable component of the Exchange Server installation. As components are usually implemented as NT services, this table effectively describes a number of NT services, but only those related to the Exchange Server. Table 197 describes the objects in `msxConfComponentTable`.

*Table 197. Component configuration group (msxConfComponentTable)*

Row object	Description
Binary	The binary file that implements the component.
Installed	Indicates whether or not the component is installed.
Name	The name or description of the component.
Version	The version of the file that implements the component.

## Information store configuration

The information store configuration group, `msxConfIS`, provides details about the configuration of the Exchange Server information store (IS). Table 198 describes the objects in `msxConfIS`.

*Table 198. Information store configuration group (msxConfIS)*

Object	Description
Buffers	The maximum number of buffers allocated by the information store.
GatewayInThreads	The number of threads available for moving messages from the MTA into the IS MTS-Out queue.
GatewayOutThreads	Number of threads available for moving messages from the IS MTS-In queue to the MTA.

Table 198. Information store configuration group (msxConfIS) (continued)

Object	Description
MaxThreads	The maximum number of threads allocated by the information store.
MinThreads	The minimum number of threads allocated by the information store.
PrivateStoreFile	The file containing the private information store / mailbox.
PublicStoreFile	The file containing the public information store.

## Message transfer agent configuration

The message transfer agent configuration group, msxConfMTA, provides details about the configuration of the Exchange Server Message Transfer Agent (MTA). Table 199 describes the objects in msxConfMTA.

Table 199. Message transfer agent configuration group (msxConfMTA)

Object	Description
DBBufPerObject	The number of DB server buffers allocated per DB object. A higher number indicates that more memory will be used, with lower likelihood that the DB object will be rolled out to disk due to lack of buffer space.
DBFileHandles	Maximum number of DB file handles. This indicates the number of DB files that can be open concurrently.
DBPath	The directory containing the MTA database.
DispatchThreads	The number of message dispatch threads configured.
KernelThreads	The number of kernel threads.
MaxRpcCalls	The maximum number of RPC procedure calls guaranteed to be processed concurrently.
MDBUsers	Defines the number of distinguished names (that is, users) to cache from the directory.
RtsThreads	The number of threads that handle the RTSE level of the OSI stack.
SubmitThreads	The number of message submission (accepting) threads configured.
TcpipBlocks	The number of control blocks to allocate for TCP/IP connections. These blocks are used by the various connectors. The value of this object is usually directly proportional to the number of configured connectors.
TransferThreads	The number of message transfer threads configured.

## Directory services configuration

The directory services configuration group, msxConfDS, provides details about the configuration of the Exchange Server directory services (DS). Table 200 describes the objects in msxConfDS.

Table 200. Directory services configuration group (msxConfDS)

Object	Description
DBFile	The file that contains the public directory service.
MaxThreads	The number of threads in the directory service.

## Performance group

The performance group (msxPerf) provides objects in this group measure dynamic performance of Microsoft Exchange Server installations. The msxParametersSampleInterval object determines the interval at which metrics are sampled.

## Exchange server resource usage group

The Exchange Server resource usage group, msxPerfExchange, provides a set of objects describing the overall Exchange Server performance metrics. Table 201 describes the objects in msxPerfExchange.

Table 201. Exchange server resource usage group (msxPerfExchange)

Object	Description
Memory	The physical memory usage (in KB).
PageFaults	Not implemented.
PercentCPU	CPU usage by Exchange services during last sample interval, expressed as a percentage value.
PercentMemory	Physical memory usage, expressed as a percentage of total physical memory capacity.
Threads	The number of threads currently used by Exchange services.
VirtualMemory	Virtual memory usage (in KB).

## Exchange server component performance

The Exchange Server component performance table, msxPerfComponentTable, provides performance information on Exchange Server components. Each row in the table contains data about one of the components listed in msxConfComponentTable. If a component in msxConfComponentTable is not installed on the server, the objects in the corresponding row of msxPerfComponentTable have the value 0. Table 202 describes the information contained in msxPerfComponentTable.

Table 202. Exchange server component performance table (msxPerfComponentTable)

Row object	Description
Memory	Current RSS/Working Set (in KB).
PageFaults	Not implemented.
PercentCPU	CPU usage over the last sample period, expressed as a percentage value.
PercentMemory	Physical memory usage, expressed as a percentage of total physical memory capacity.
PID	Process ID.
Threads	Number of threads currently used.
VirtualMemory	Virtual memory usage (in KB).

## Information store performance

The information store performance group, msxPerfIS, provides general data about the performance of the information store. Table 203 on page 330 lists the objects in msxPerfIS.

Table 203. Exchange server information store performance group (msxPerfIS)

Object	Description
RPCRequests	Number of client requests processed simultaneously by the Information Store.
Users	Current number of active users connected to the Information Store.

In addition msxPerfIS contains the groups:

- Public store performance group, msxPerfISPublic
- Private store/mailbox performance, msxPerfISPrivate

The public store performance group, msxPerfISPublic, provides performance metrics on the Exchange Server public store. Table 204 describes the objects in msxPerfISPublic.

Table 204. Public store performance group (msxPerfISPublic)

Object	Description
ActiveLogons	The number of clients that performed any action within the last ten minute interval.
AvgLocDeliveryTime	The average time between the submission of a message to the public folder store and the delivery to all local recipients (that is, recipients on the same server), calculated over the last 10 messages.
InMsgs	Total number of messages submitted to the public store.
InMsgsPerMin	The number of messages submitted by clients per minute.
InQLen	Inbound queue length.
OutMsgs	Total number of messages delivered from the public store.
OutMsgsPerMin	The number of messages sent to other storage providers per minute.
OutQLen	Outbound queue length.
RemDeliveryTime	The average time between the submission of a message to the public folder store and submission to other storage providers, calculated over the last 10 messages.

The private store/mailbox performance group, msxPerfISPrivate, provides performance metrics on the Exchange Server private store. Table 205 describes the objects in msxPerfISPrivate.

Table 205. Private store/mailbox performance group (msxPerfISPrivate)

Object	Description
ActiveLogons	The number of clients that performed any action within the last ten minute time interval.
AvgLocDeliveryTime	The average time between the submission of a message to the private folder store and the delivery to all local recipients (that is, recipients on the same server), calculated over the last 10 messages.
InMsgs	Total number of messages submitted to the private store.
InMsgsPerMin	The number of messages submitted by clients per minute.
InQLen	Inbound queue length.

Table 205. Private store/mailbox performance group (msxPerfISPrivate) (continued)

Object	Description
OutMsgs	Total number of messages delivered from the private store.
OutMsgsPerMin	The number of messages sent to the transport per minute.
OutQLen	Outbound queue length.
RemDeliveryTime	The average time between the submission of a message to the private folder store and submission to other storage providers, calculated over the last 10 messages.

## Message transfer agent performance

The message transfer agent performance group, msxPerfMTA, provides performance metrics on the Exchange Server message transfer agent (MTA). Table 206 describes the objects in msxPerfMTA.

Table 206. MTA performance group (msxPerfMTA)

Object	Description
Associations	The number of open associations with other MTAs.
InKBytes	The total volume of message content received since MTA initialization (in KB).
InMsgs	The total number of messages received since MTA initialization.
InRejectedAssocReason	The reject reason code, if any, for the last association rejected by this MTA.
InRejectedAssocs	The total number of inbound (that is, remotely initiated) associations that have been rejected since MTA initialization.
MsgsPerSec	The number of messages processed per second.
OldestMsgQueued	The time (in seconds) since the oldest message in this entity's queue was placed there.
OutFailedAssocReason	The failure reason code, if any, for the last association attempted by this MTA.
OutFailedAssocs	The total number of outbound (that is, locally initiated) associations that have failed since MTA initialization.
OutKBytes	The total volume of message content transmitted since MTA initialization (in KB).
OutMsgs	The total number of messages transmitted since MTA initialization.
QKBytes	The total volume of message content currently stored in the MTA, (in KB).
WorkQLen	The number of outstanding messages in the work queue. This indicates the number of messages not yet processed to completion by the MTA.

## Directory services performance

The directory services performance group, msxPerfDS, provides performance metrics on the Exchange Server 5.5 directory services (DS). Table 207 on page 332 describes the objects in msxPerfDS.

Table 207. Directory services performance group (msxPerfDS)

Object	Description
ABBrowSES	The number of address book browse operations performed by Exchange clients per second.
ABReads	The number of address book read operations performed by Exchange clients per second.
CacheEntries	The total number of entries in the cache. This includes DN, Search, Not Found DN and Not Found GUID type entries.
CacheHitRatio	$(\text{hits delta}/(\text{hits delta}+\text{misses delta}))\times 100$ for the last sample period.
CacheHits	The total number of 'object found in cache' events since system initialization.
CacheMaxEntries	The maximum number of entries allowed in the cache.
CacheMaxMemory	The maximum allowed memory size of the cache (in KB).
CacheMemory	The memory occupied by all entry objects in the cache (in KB), excluding the cache table overhead.
CacheMisses	The total number of 'object not found in cache' events since system initialization.
ExDSReads	The number of read operations performed by extended directory service clients per second.
ExDSWrites	The number of write operations performed by extended directory service clients per second.
ReplicationUpdates	The rate of directory replication activity on this server, measured in updates per second.
Threads	The number of threads in use by the directory service.

## Site replication service performance

The site replication service performance group, msxPerfSRS, provides performance metrics on the Exchange Server 2000 site replication service (SRS). Table 208 describes the objects in msxPerfSRS.

Table 208. Site replication service performance group (msxPerfSRS)

Object	Description
AccessDenials	The number of times that write operations were refused for security reasons.
LDAPSearches	Total number of LDAP searches performed since the SRS started.
LocUpdateRate	The rate at which replication updates are applied by the local site replication service. This indicates how much replication activity is occurring on the server.
ThreadsInUse	The current number of RPC threads active in the site replication service (different than the number of threads in the service process). Threads in Use is the number of RPC-generated threads currently in API calls and can be used to indicate whether additional processors could be of benefit.
UpdateRate	The rate at which replication update messages from other Exchange sites are applied by the local site replication service. This indicates how much inter-site replication activity is occurring on the server.

## Connector performance

The connector performance group provides performance metrics on Exchange Server connectors. It contains the groups:

- POP3 performance, `msxPerfPOP3`
- SMTP performance, `msxPerfSMTP`
- MMC performance, `msxPerfMMC`
- CCMC performance, `msxPerfCCMC`

The POP3 performance group, `msxPerfPOP3`, provides performance metrics on the Exchange Server POP3 connector. Table 209 describes the objects in `msxPerfPOP3`.

*Table 209. POP3 performance group (msxPerfPOP3)*

Object	Description
CurrentCons	Total number of current connections to the POP3 server.
FailedCons	Total number of failed connections to the POP3 server.
InvalidCmds	Total number of invalid commands sent to the POP3 server.
RejectedCons	Total number of rejected connections to the POP3 server.
TotalCons	Total number of connections to the POP3 server.

The POP3 performance group also contains the POP3 server command performance table, `msxPerfPOP3CmdTable`. Each row in the table provides information on the use of a specific POP command. Table 210 describes the objects in `msxPerfPOP3CmdTable`.

*Table 210. POP3 command table (msxPerfPOP3CmdTable)*

Row object	Description
Count	Number of times the command has been issued to the POP3 server.
Failures	Number of times the command has failed.
Name	The name of the command.

The SMTP performance group, `msxPerfSMTP`, metrics on the Exchange Server SMTP connector. Table 211 describes the objects in `msxPerfSMTP`.

*Table 211. SMTP performance group (msxPerfSMTP)*

Object	Description
AvgRetriesMsgsDelivered	Average number of retries per local delivery.
AvgRetriesMsgsOut	Average number of retries per outbound message sent.
CatQLen	The number of messages in the categorizer queue.
InBytes	The total number of bytes received by the SMTP server.
InBytesPerSec	The rate at which messages are received, in bytes per second.
InCons	The current number of inbound connections.
InConsTotal	The total number of inbound connections received.
InMsgs	The total number of messages received by the SMTP server.
LocQLen	The number of messages in the local queue.
LocRetryQLen	The number of messages in the local retry queue.

Table 211. SMTP performance group (msxPerfSMTP) (continued)

Object	Description
OutBytes	The total number of bytes sent by the SMTP server.
OutBytesPerSec	The rate at which messages are sent from the server, in bytes per second.
OutCons	The current number of outbound connections.
OutConsTotal	The total number of outbound connections made.
OutMsgs	The total number of messages sent by the SMTP server.
RemQLen	The number of messages in the remote queue.
RemRetryQLen	The number of messages in the retry queue for remote delivery.

The MMC performance group, msxPerfMMC, provides a set of performance indicators for the Microsoft mail connector (MMC). Table 212 describes the objects in msxPerfMMC.

Table 212. MMC performance group (msxPerfMMC)

Object	Description
InMsgs	The total number of messages received by the MMC.
OutMsgs	The total number of messages sent by the MMC.

The CCMC performance group, msxPerfCCMC, provides a set of performance indicators for the Exchange Server connector for Lotus cc:Mail (CCMC). Table 213 describes the objects in msxPerfCCMC.

Table 213. CCMC performance group (msxPerfCCMC)

Object	Description
InMsgs	The total number of messages received by the CCMC.
InQLen	The length of the inbound queue.
OutMsgs	The total number of messages sent by the CCMC.
OutQLen	The length of the outbound queue.

## Instant messaging group

The instant messaging performance group, msxPerfIM, provides performance indicators for the Instant Messaging (IM) virtual servers configured on the host Exchange server. Table 214 describes the objects in msxPerfIM.

Table 214. IM performance group (msxPerfIM)

Object	Description
InSubscribesPerSec	The average number of SUBSCRIBE requests serviced per second.
OnlineUsers	The number of users currently connected to all IM virtual servers on the host.
Subscriptions	The total number of subscriptions that are currently active, across all IM virtual servers on the host.



## Address list group

Table 215 lists the objects in the address list performance group (msxPerfAL).

*Table 215. Address list group (msxPerfAL)*

Object	Description
QLen	An indication of the load that the Recipient Update Service (RUS) is under. The Recipient Update Service is a service that updates recipient objects within a domain with information such as e-mail addresses and address list membership.

## Mailbox table

The mailbox table, msxPerfMailboxTable, provides a list of mailboxes stored on the Exchange server. Each entry in the table represents a single mailbox stored on the server. Table 216 describes the row objects in msxPerfMailboxTable.

*Table 216. Mailbox table (msxPerfMailboxTable)*

Object	Description
Created	The date and time when this mailbox was created.
Items	The total number of items in the mailbox.
Name	The display name of the mailbox folder.
Size	The total size of all items in the mailbox, in KB.

## Public folder table

The public folder table, msxPerfPublicFolderTable, provides a list of public folders stored on the Exchange server. Each row in the table represents a single public folder stored on the server. Table 217 describes the row objects in msxPerfPublicFolderTable.

*Table 217. Public folder table (msxPerfPublicFolderTable)*

Object	Description
FolderAccessed	The date and time when the folder was last accessed.
FolderCreated	The date and time when the folder was created.
FolderName	The display name of the public folder.
FolderSize	The total size of all items in the folder, in KB.
Items	The total number of items in the folder.



---

## Chapter 32. Netcool/ASM for Microsoft IIS

Netcool/ASM for Microsoft IIS provides comprehensive support for monitoring performance and availability of Microsoft Internet Information Server/Services (IIS) servers. It addresses aspects such as Web, FTP, SMTP and NNTP performance, and contains metrics such as failed Web and FTP logins, pages not found, traffic monitors and FTP traffic.

The Netcool/ASM for Microsoft IIS adapts to site configuration changes and discovers newly added sites or removes sites that are no longer hosted. It can automatically generate alarms based on the value of particular performance metrics.

Netcool/ASM for Microsoft IIS consists of the `iis` subagent and the `iis` MIB module. It operates on Windows platforms.

---

### Component files

The `iis` subagent and MIB module are comprised of a set of files.

Table 218 lists the `iis` subagent and MIB module component files.

*Table 218. Netcool/ASM for Microsoft IIS component files*

File	Location	Description
<code>iis.dll</code>	<code>bin</code>	Binary implementation.
<code>iis-mib.mib</code>	<code>mibs</code>	MIB definition document.
<code>iis.oid</code>	<code>config/oid</code>	Object identifier file.
<code>iis.cfg</code>	<code>config</code>	Sample configuration file.

---

### Guidelines

The `iis` subagent does not require setup; it automatically discovers IIS and gathers the required data.

To load the subagent, use the command:

```
subagent load iis
```

### Events

The `iis` subagent does not generate events, however you can set up the Netcool/SSM `genalarm` subagent to monitor thresholds for objects in the `iis` MIB using the `iis.cfg` file.

The `iis.cfg` file specifies a default set of objects and thresholds for the `iis` MIB that can be monitored by the `genalarm` subagent. For each object specified in the file, a row is created automatically in the `genalarm` MIB. Table 219 on page 338 lists the objects that are specified in the `iis.cfg` file with a description of the metric being monitored.

Table 219. Monitored IIS objects

Object	Monitored Metric
iisFtpSiteFailedLogons	Failed FTP Logons
iisFtpSiteKbytesRecv	FTP incoming traffic monitor
iisFtpSiteKbytesSent	FTP outgoing traffic monitor
iisNntpServerArtRejected	NNTP article rejection monitor
iisSmtplibServerLocalQLen	SMTP local queue length monitor
iisSmtplibServerLocalRetryQLen	SMTP local retry queue length monitor
iisSmtplibServerNdrCount	Non-delivery Reports monitor
iisSmtplibServerRemoteQLen	SMTP remote queue length monitor
iisSmtplibServerRemoteRetryQLen	SMTP remote retry queue length monitor
iisWebAspCompileErrors	ASP compile errors
iisWebAspPrepErrors	ASP preprocessor errors
iisWebAspRuntimeErrors	ASP runtime errors
iisWebConnAttempts	Excessive web connection attempts
iisWebSiteFailedLogons	Failed Web Logons
iisWebSiteKbytesSent	Web traffic monitor
iisWebSiteNotFoundCount	Web pages not found
iisWebSitePeakAnonUsers.\$_siteindex	Peak anonymous users

**Note:** You can edit the `iis.cfg` file to set up thresholds that are suitable for your system.

To use the `iis.cfg` file to create entries in the `genAlarm` MIB and monitor thresholds in the `iis` MIB, create a row in the `agentConfigTable` for the `iis.cfg` file. When the row is activated, the `iis.cfg` file is executed. To ensure the `iis.cfg` file has been executed and view the objects being monitored, check that the appropriate rows have been created in the `genAlarm` MIB.

**Note:** To ensure the `iis.cfg` file is loaded when the agent is instructed to restore values, save the `agentConfigTable` state to the `agent.state` file.

## MIB module

The `iis` MIB is a subtree of `networkharmoni(1977)`.

The subtree is shown in Figure 61 on page 339.

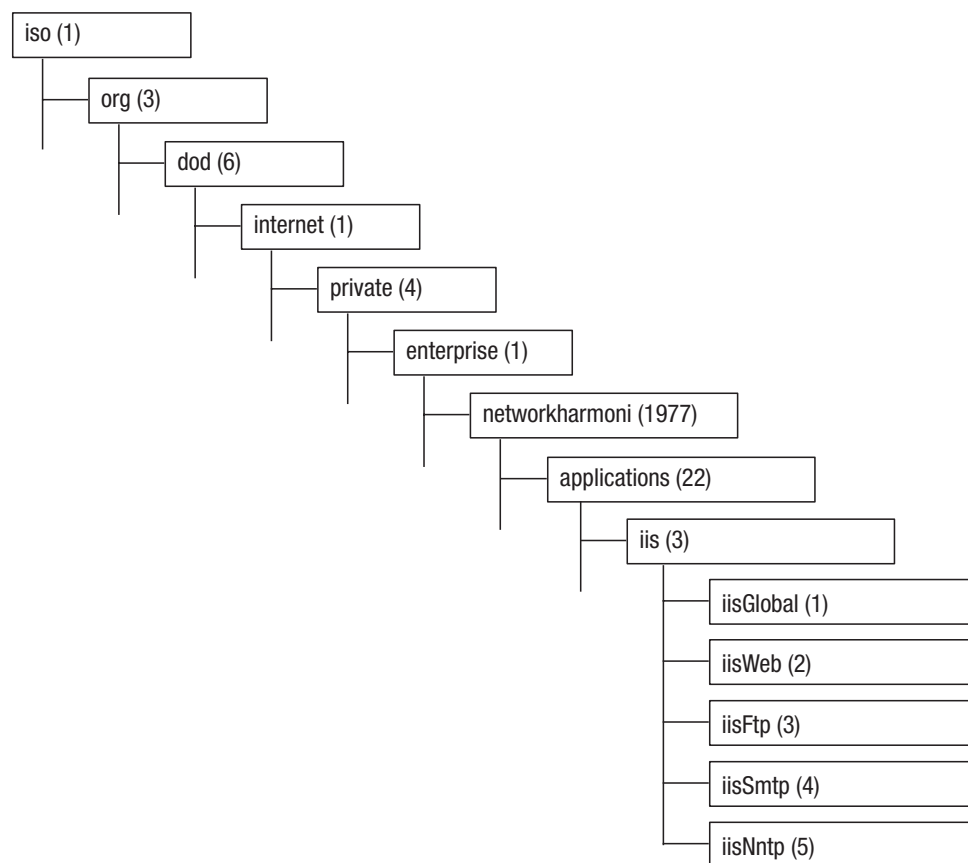


Figure 61. OID Tree Diagram of the IIS MIB Module

This section provides a summary of the objects defined in the `iis` MIB module. For detailed information on all objects in the module, see the `iis-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB structure

The `iis` MIB provides data in the form of scalar objects and tables.

### Global group

The global group contains scalar objects that store global data from IIS. This data is common to all components of IIS.

These scalars are listed in Table 220.

Table 220. IIS global scalar objects

Object	Description
CpuTime	The total CPU time consumed by IIS.
MemoryResident	The amount of MemoryTotal that is currently resident in RAM, in KB.
MemoryTotal	The total amount of memory allocated to IIS in KB.
Pid	The current process ID of the IIS service ( <code>inetinfo.exe</code> ).
ReqBlocked	The number of asynchronous I/O requests that have been blocked due to bandwidth throttling.
ReqServiced	The total number of asynchronous I/O requests serviced.

Table 220. IIS global scalar objects (continued)

Object	Description
Threads	The number of threads IIS currently has.
UserCpuTime	The total user-space CPU time consumed by IIS.
Version	The version of IIS installed.

## Web group

All counters and sums are relative to the time at which the Web service started.

## Web scalars

Total values for Web connection statistics are presented by the following:

- `iisWebConnAttempts`  
The number of connections that have been attempted to the Web service. This is a total, for all Web sites.
- `iisWebPeakConn`  
The maximum number of simultaneous connections established with the Web service.

## Web site table

The site table (`iisWebSiteTable`) represents a table of Web sites served by the IIS Web Service. The table is indexed by the object `iisWebSiteIndex`. Table 221 lists the Web site table objects. The values stored in `iisWebSiteTable` are specific to each site.

Table 221. Site table (`iisWebSiteTable`)

Object	Description
CgiCount	The number of CGI (Common Gateway Interface) requests made to the Web site.
CurAnonUsers	The number of anonymous users currently connected to the Web site.
CurAuthUsers	The number of fully authenticated users currently connected to the Web site.
CurUsers	The number of users currently connected to the Web site.
DeleteCount	The number of HTTP DELETE requests made to the Web site.
FailedLogons	The number of logon attempts to the Web site that have failed. Usually this counter only applies to sites that don't allow anonymous access.
FilesRecv	The number of files uploaded to the Web site.
FilesSent	The number of files served by the Web site.
GetCount	The number of HTTP GET requests made to the Web site.
HeadCount	The number of HTTP HEAD requests made to the Web site.
Index	A logical table index for this Web site, starting at 1. Index 1 is always the special <code>_Total</code> entry, representing statistic totals for all Web sites.
IsapiCount	The number of ISAPI Extension requests made to the Web site.
KbytesRecv	The number of KB received (in requests and uploads) by the Web site.
KbytesSent	The number of KB served by this Web site.

Table 221. Site table (iisWebSiteTable) (continued)

Object	Description
LogonAttempts	The number of logons attempted (but not necessarily granted) to the Web site.
MethodCount	The total number of HTTP requests (including GET, PUT, DELETE, HEAD and POST) made to the Web site.
Name	The name of the Web site, as configured in the Internet Services Manager.
NotFoundCount	The total number of requests that couldn't be satisfied by the server because the requested document could not be found. These are generally reported as an HTTP 404 error code to the client.
PeakAnonUsers	The maximum number of anonymous users to establish concurrent connections with the Web site.
PeakAuthUsers	The maximum number of authenticated users to establish concurrent connections with the Web site.
PeakCgiReq	The maximum number of simultaneous CGI requests that the Web site has processed.
PeakIsapiReq	The maximum number of simultaneous ISAPI Extension requests that the Web site has processed.
PostCount	The number of HTTP POST requests made to the Web site.
PutCount	The number of HTTP PUT requests made to the Web site.
TotalAnonUsers	The total number of anonymous users who have connected to the Web site.
TotalAuthUsers	The total number of authenticated users who have connected to the Web site.
TotalUsers	The total number of users who have connected to the Web site.

## ASP subgroup

The ASP subgroup is a child of the Web group. It consists solely of scalar objects as shown in Table 222.

Table 222. ASP scalars

Object	Description
CompileErrors	The number of requests that have failed due to script compilation errors.
CurSess	The number of sessions currently being serviced.
PrepErrors	The number of requests that have failed due to preprocessor errors.
ReqDis	The number of requests that were disconnected due to a communications failure.
ReqFailed	The total number of requests that failed due to errors, authorization failure or rejections.
ReqNotAuth	The number of requests that have failed due to insufficient access rights.
ReqNotFound	The number of requests for files that were not found.
ReqQueued	The number of requests currently queued, waiting for service.

Table 222. ASP scalars (continued)

Object	Description
ReqRejected	The number of requests that were not executed due to insufficient resources. This may be due to configuration problems or a memory limitation.
ReqSucceeded	The number of requests that were successfully executed.
ReqTimedOut	The number of requests that have timed out.
ReqTotal	The total number of ASP requests made to the Web service.
RuntimeErrors	The number of requests that have failed due to runtime errors.
SessTimedOut	The number of sessions that have timed out.
TotalSess	The number of sessions that have been opened since the service was started.
TransAborted	The number of transactions that were aborted (rolled back).
TransAttempted	Total number of transactions attempted. This is the sum of the three statistics iisWebAspTransPending, iisWebAspTransCommitted and iisWebAspTransAborted.
TransCommitted	The number of transactions that have been committed.
TransPending	The number of transactions that have not yet been committed.

## FTP group

All counters and sums are relative to when the FTP Service started.

## FTP scalars

iisFtpConnAttempts stores the total number of connections attempted to all FTP sites in the FtpSiteTable.

## FTP site table

The FTP Site Table (iisFtpSiteTable) represents a table of FTP sites served by the IIS FTP Service. The table is indexed by the object iisFtpSiteIndex. FTP Site Table objects are listed in Table 223. The values stored in iisFtpSiteTable are specific to each site.

Table 223. FTP site table (iisFtpSiteTable)

Object	Description
CurAnonUsers	The number of anonymous users currently logged into the FTP site.
CurAuthUsers	The number of authenticated users currently logged into the FTP site.
CurUsers	The number of users currently logged into the FTP site.
FailedLogons	The number of failed logon attempts made to the FTP site.
FilesRecv	The number of files that have been uploaded to this FTP site.
FilesSent	Total number of files served by this FTP site.
Index	A logical table index for this FTP site, starting at 1. Index 1 is always the special _Total entry, representing statistic totals for all the FTP sites.
KbytesRecv	The number of KB received at the application layer by the FTP site.
KbytesSent	The number of KB served at the application layer by the FTP site.



Table 223. FTP site table (iisFtpSiteTable) (continued)

Object	Description
LogonAttempts	The total number of logon attempts made to the FTP site, including failed ones. Note that this will be a super-set of the number of connections established with the FTP site.
Name	The name of the FTP site, as configured in the Internet Services Manager.
PeakAnonUsers	The maximum number of simultaneous anonymous users to have been logged into the FTP site.
PeakAuthUsers	The maximum number of simultaneous authenticated users to have been logged into the FTP site.
PeakConn	The maximum number of simultaneous connections (not just logged-on users) that have been established with the FTP site.
TotalAnonUsers	The total number of anonymous users who have logged into the FTP site.
TotalAuthUsers	The total number of authenticated users who have logged into the FTP site.
TotalUsers	The total number of users who have logged into the FTP site.

## SMTP group

The SMTP group contains a single table describing the SMTP service.

### SMTP server table

The SMTP Table (iisSmtServerTable) is a table of SMTP servers, maintained by the SMTP Server service (although there is usually only one). Table 224 lists the objects in iisSmtServerTable.

Table 224. SMTP table (iisSmtServerTable)

Object	Description
ConnErrors	The number of connection errors that have occurred.
CurInConn	The current number of inbound connections.
CurOutConn	The current number of outbound connections.
Index	A logical table index for this SMTP site, starting at 1. Index 1 is always the special _Total entry, representing statistic totals for the entire SMTP service.
KbytesRecv	The total number of KB received by the SMTP server; it also includes additional bytes from headers and control data.
KbytesSent	The total number of KB sent by the SMTP server; it also includes additional bytes from headers and control data.
LocalQLen	The number of messages in the local queue.
LocalRetryQLen	The number of messages in the local retry queue.
MsgKbytesRecv	The total number of KB received in messages.
MsgKbytesSent	The total number of KB sent in messages.
MsgRecv	The number of messages received by the server (incoming).
MsgRetries	The total number of outbound message sends that were retried.
MsgSent	The number of messages sent by the server (outgoing).
Name	The name of the SMTP server.

Table 224. SMTP table (iisSmtpServerTable) (continued)

Object	Description
NdrCount	The number of non-delivery reports that have been generated.
RemoteQLen	The number of messages in the remote queue.
RemoteRetryQLen	The number of messages in the retry queue for remote delivery.
TotalInConn	The total number of inbound connections that have been established.
TotalOutConn	The total number of outbound connections that have been established.

## NNTP group

The NNTP group contains a single table describing the NNTP service.

### NNTP server table

The NNTP table (iisNntpServerTable) is a table representing the NNTP servers maintained by the Network News Transport Protocol (NNTP) service. Table 225 lists the objects in iisNntpServerTable.

Table 225. NNTP table (iisNntpServerTable)

Object	Description
ArtPosted	The total number of articles posted by the server (a subset of ArtRecv).
ArtRecv	The total number of articles received (not necessarily posted) by the server.
ArtRejected	The total numbers of articles rejected by the server (ArtRecv minus ArtPosted).
ArtSent	The number of articles sent by the server to other servers.
CurAnonUsers	The current number of anonymous users logged into the server.
CurAuthUsers	The current number of authenticated users logged into the server.
CurInConn	The current number of inbound connections to the server.
CurOutConn	The current number of outbound connections from the server.
CurUsers	The current number of users logged into the server.
Index	A logical table index for this NNTP server, starting at 1. Index 1 is always the special _Total entry, representing statistic totals for the entire NNTP service.
KbytesRecv	The total number of KB received by the server.
KbytesSent	The total number of KB sent by the server.
Name	The name of the NNTP server.
TotalAnonUsers	The total number of anonymous users who have logged into the server.
TotalAuthUsers	The total number of authenticated users who have logged into the server.
TotalInConn	The total number of inbound connections that have been established with the server.
TotalOutConn	The total number of outbound connections that have been established with the server.
TotalUsers	The total number of users who have logged into the server.

---

## Chapter 33. Netcool/ASM for Oracle

The Netcool/ASM for Oracle monitors various aspects of Oracle database performance such as database, table space, buffer pool, processes, sessions, and I/O.

Netcool/ASM for Oracle provides metrics that include free space, process memory usage, process page faults, transaction rates and average wait time. It can automatically generate alarms based on the value of particular performance metrics

Netcool/ASM for Oracle consists of the `oracle` subagent, which loads and runs on Netcool/SSM, and the `oracle` MIB module, which stores configuration, performance, and efficiency data about Oracle database servers and database instances. The subagent monitors Oracle database servers by querying Oracle internal performance tables using the Oracle Call Interface (OCI) libraries. It automatically detects any Oracle installation running on the host machine on which it is installed, and all database instances present.

---

### Overview

The `oracle` subagent and MIB module are comprised of a set of component files.

Table 226 lists the `oracle` subagent and MIB module component files and their installed locations.

*Table 226. Netcool/ASM for Oracle component files*

File	Location	Description
<code>oracle.dll</code> (Windows) <code>liboracle.so/.sl</code> (UNIX)	<code>bin</code>	Binary implementation.
<code>oracle-mib.mib</code>	<code>mibs</code>	MIB definition document.
<code>oracle.oid</code>	<code>config/oid</code>	Object identifier file.
<code>oracle.cfg</code>	<code>config</code>	Sample configuration file.

### Deployment

To monitor an Oracle database server, you must install Netcool/SSM on the machine that runs the Oracle server. A single Netcool/ASM for Oracle subagent is able to monitor multiple Oracle database instances running on the same machine; however it cannot monitor Oracle servers remotely. If you wish to monitor Oracle servers running on more than one machine, you must install Netcool/SSM on each machine.

## Oracle Real Application Clusters (RAC)

Netcool/ASM for Oracle enables you to monitor Oracle RAC nodes. To monitor Oracle RAC, install Netcool/SSM on each RAC node.

---

### Guidelines

To use the oracle subagent, first load it on Netcool/SSM.

#### About this task

To load the oracle subagent, use the following configuration command:

```
subagent load oracle
```

**Note:** Netcool/SSM must be running on the same machine as the Oracle server. It cannot monitor an Oracle server remotely.

When the oracle subagent loads, it automatically detects the Oracle installation and database instances, then updates the tables oraInstallationTable and oraInstanceTable accordingly.

### General monitoring procedure

To monitor an Oracle database, configure the oraInstanceTable with information about the target database instance.

#### Procedure

1. In oraInstanceTable, identify the row representing the database instance you wish to monitor.
2. Set the row's oraInstanceUpdateFreq to the sample interval at which you wish the statistics to be updated.  
For a busy database, a setting of 300 (5 minutes) is recommended (this is the default value). Do not set this object to a value less than 20 (20 seconds).
3. Set the oraInstanceUsername and oraInstancePassword objects to the username and password of an existing Oracle user.  
For more information about database users, see "Specifying the Oracle database user" on page 347.
4. Set the value of oraInstanceMonitored to yes.
5. Check the oraInstanceStatus object to determine the sampling status and whether any errors have occurred.

#### Results

After the sample interval has elapsed, objects in the Configuration, Statistics, and Efficiency groups contain data.

## Locating the Oracle home directory

During the installation process, the Netcool/SSM installer attempts to determine the home directory of any Oracle database running on the host machine.

### About this task

On UNIX systems, if you do not specify the Oracle home directory during installation, the installer attempts to obtain it from information contained in the Oracle installation's oratab file, which it locates using the environment variable ORATAB\_LOCATION. If ORATAB\_LOCATION does not exist, the installer then attempts to find the file using the standard pathnames /etc/oratab and /var/opt/oracle/oratab.

If the installer cannot locate the oratab file using either of these methods, it cannot determine the Oracle home directory. In this case, you must specify the Oracle home directory and oratab file by setting the OracleHome and OratabLocation inivars manually; see "Inivars" for details.

## Inivars

Netcool/ASM for Oracle provides inivars that specify information about the Oracle database server installed on the host machine.

Table 227 lists these inivars. Normally, the Netcool/SSM installer sets their values automatically during the installation process, however you may alter them using the set inivar configuration command or by editing their definitions in the init.cfg file.

Table 227. oracle subagent inivars

Inivar	Type	Description
OracleHome	string	Specifies the location of the Oracle home directory. This inivar sets the ORACLE_HOME environment variable, which is necessary for running the OCI libraries.
OratabLocation	string	(UNIX only) Specifies the location of the Oracle oratab file.

For general information about inivars, see the *Netcool/SSM Administration Guide*.

## Specifying the Oracle database user

The oracle subagent requires user credentials to access an Oracle database instance.

The oraInstanceUsername and oraInstancePassword objects specify the Oracle username and password. This user must have privileges sufficient for executing SQL SELECT statements on the Oracle v\$ and dba tables. If the user does not have the required privileges, some statistics will be unavailable to the subagent and the corresponding MIB objects will not contain values. These privileges are provided by the CONNECT and SELECT\_FROM\_CATALOG roles.

To create a user with the required privileges, use the following SQL commands:

```
CREATE USER username IDENTIFIED BY password
GRANT CONNECT TO username
GRANT SELECT_FROM_CATALOG TO username
```

## Encrypting passwords

To keep Oracle passwords secure, use the Netcool/SSM password encryption facility rather than storing passwords in clear text.

### About this task

To encrypt a password, open a Netcool/SSM command console and enter the command:

```
encrypt password
```

where *password* is the password you wish to encrypt. In response, Netcool/SSM displays the encrypted password on the command console. Make note of the encrypted password and use it in place of the original password.

To provide an encrypted password as a property in a configuration command use the @= operator in place of the usual = operator as follows:

```
command password_property@=encrypted_password
```

For example, to supply the encrypted password WMQIa0j0LeVHA with the oracle configuration command, use the format:

```
oracle index=n password@=WMQIa0j0LeVHA
```

## Controlling queries

The query control group oraQueryControl provides a set of objects that control data collection from particular MIB tables in the oracle MIB data. Using these objects, you can customize the amount of data collected.

On large Oracle installations, these objects help you prevent excessive query times by disabling sections of the MIB module. Table 228 lists the control objects available and the tables that they control.

Table 228. Query control objects

Query Control Object Name	Data Tables Controlled
oraDatafileMonitoring	oraDBFileTable
	oraPerfIOWDataFileTable
oraLockMonitoring	oraStatLockTable
oraSessionMonitoring	oraSessionTable
oraTablespaceMonitoring	oraTableSpaceTable
	oraPerfIOWTablespaceTable

To control data gathering for a particular table, use the corresponding query control object as follows:

- To permit data gathering for the table, set the control object to enabled(1)
- To prevent data gathering for the table, set the control object to disabled(2)

**Note:** Disabling monitoring by setting a query control object to disabled(2) destroys any data in the corresponding tables. If you wish to suspend data collection and preserve any existing data, use the oraInstanceDataControl object instead.

## Monitoring performance thresholds and generating events

The oracle subagent does not generate events; however, you can use the Netcool/SSM genalarm subagent to monitor thresholds on objects in the oracle MIB and generate events in response to threshold violations.

The configuration file `oracle.cfg`, located in the Netcool/SSM config directory, contains a set of pre-defined threshold monitors for a range of oracle MIB objects. You may modify the monitors and thresholds defined in this file to suit your monitoring requirements or use it as the basis for creating your own monitors.

To execute `oracle.cfg`, use the command:

```
config execute oracle.cfg
```

## Monitoring Oracle RAC

To monitor Oracle RAC systems, Netcool/SSM and Netcool/ASM for Oracle must be running on each RAC node that you wish to monitor.

Each Netcool/SSM installation monitors only the RAC node running on its host machine. Each monitor automatically ensures that the statistics collected relate only to that node and are not affected by RAC failover and load balancing operations.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Instance table

The oracle command sets objects in the instance table (`oraInstanceTable`).

The general syntax of this command is:

```
oracle index=n property=value ...  
oracle reset
```

**Tip:** Always issue the `oracle reset` command before the `oracle` command. This ensures that any existing property values are cleared and not carried over to the newly issued `oracle` command.

Table 229 lists the properties supported in this command.

*Table 229. Configuration command parameters - oracle*

Property	Type	Description	Sets MIB object
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl
gdn	string	Sets the global database name of the Oracle instance.	GDN

Table 229. Configuration command parameters - oracle (continued)

Property	Type	Description	Sets MIB object
index	int	Selects the target Oracle database instance using oraInstanceIndex.	n/a
monitored	enum	Controls monitoring of the database instance:  yes - The database is monitored  no - The database is not monitored	Monitored
password	string	Specifies the password used when connecting to the monitored database instance.	Password
sid	string	Sets the system identifier of the Oracle database instance.	SID
update	int	Sets the interval (in seconds) at which statistics for the monitored database instance are updated.	UpdateFreq
username	string	Specifies the username provided when connecting to the monitored database instance.	Username

## Query control table

The `oraclectrl` command sets the scalar objects in the query control group (`oraQueryControl`).

The general syntax of this command is:

`oraclectrl property=value`

Table 230 lists the properties supported in this command.

Table 230. Configuration command parameters - oraclectrl

Property	Type	Description	Sets MIB object
datafiles	enum	Controls data gathering for the tables <code>oraDBFileTable</code> and <code>oraPerfIODataFileTable</code> :  disabled - Prevents collection  enabled - Permits collection	oraDatafileMonitoring
lockhint	string	Specifies an optimizer hint used when monitoring lock statistics. When specifying a hint, omit the comment delimiters ( <code>/* */</code> and <code>!--+</code> ) normally required in Oracle optimizer hints. For example, to specify the rule optimizer hint use the command: <code>oraclectrl lockhint="RULE"</code>	oraLockQueryHint
locks	enum	Controls data gathering for the table <code>oraStatLockTable</code> :  disabled - Prevents collection  enabled - Permits collection	oraLockMonitoring



Table 230. Configuration command parameters - oraclectrl (continued)

Property	Type	Description	Sets MIB object
sessions	enum	Controls data gathering for the table oraSessionTable:  disabled - Prevents collection  enabled - Permits collection	oraSessionMonitoring
tablespaces	enum	Controls data gathering for the tables oraTableSpaceTable and oraPerfIOTablespaceTable:  disabled - Prevents collection  enabled - Permits collection	oraTablespaceMonitoring

## Examples

This example demonstrates how to use the oracle subagent to perform simple monitoring tasks.

### Monitoring buffer cache hit ratio

Monitor the buffer cache hit ratio of all Oracle instances running on the host machine. If this value falls below 90% on any Oracle instance, generate an alarm.

The configuration commands for implementing this monitor are:

```
# Load the required subagents
subagent load rmonc
subagent load oracle
subagent load genalarm

# Define the Oracle credentials and sample interval
oracle index=1 username="name"
oracle index=1 password@="encrypted_password"
oracle index=1 update=60
oracle index=1 monitored=yes
oracle reset

# Define events for threshold violation and threshold normal
event reset
event community=public
event type=snmp-trap

event description="Oracle metric violation"
event create
violationevent=$?event description="Oracle metric nominal"event createnormalevent=$?

genalarm reset

# Identify the statistic monitored
genalarm var="$oraEffMemBufCache.?"
genalarm vardescr="Oracle Performance: Buffer cache hit ratio"

# Select the alarm mode
genalarm mode=singleEdge
genalarm interval=300
genalarm type=absolute
genalarm startup=rising

# Configure the falling alarm
```

```

genalarm fallthresh=9000
genalarm fallduration=0
genalarm fallevent=$normalevent
genalarm fallseverity=warning
genalarm fallevent=$violationevent
genalarm falldescr="SID: $$11 - Buffer cache hit ratio is below 90%"

# Configure the rising alarm
genalarm risethresh=9000
genalarm riseduration=0
genalarm riseevent=$violationevent
genalarm riseseverity=info
genalarm risedescr="SID: $$11 - Buffer cache hit ratio is above 90%"
genalarm riseevent=$normalevent

# Bind the SID of the monitored Oracle instance to notifications
genalarmvb store oid="$oraInstanceSID.*"
genalarm create

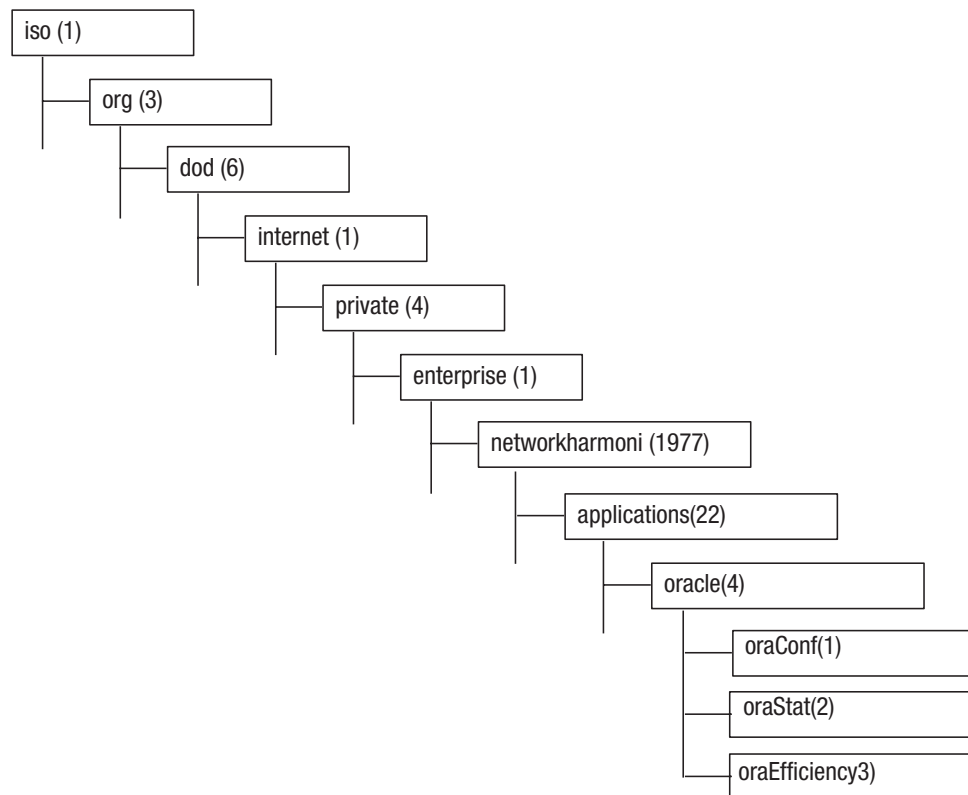
```

**Tip:** Use the .? operator to specify all Oracle instances on the host machine.

## MIB module

The oracle MIB is a subtree of networkharmoni(1977).

The subtree is shown in Figure 62.



*Figure 62. OID Tree Diagram of the Netcool/ASM for Oracle MIB Module*

This section provides a summary of the objects defined in the oracle MIB module. For detailed information on all objects in the module, see the `oracle-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB structure

The oracle MIB module organizes performance statistics into groups.

- Configuration, oraConf, providing monitor configuration and static database information
- Statistics, oraStat, providing database performance information
- Efficiency, oraEfficiency, providing data for trending and reporting

**Note:** In the oracle MIB module, an object name that ends with 9i—for example, oraDBInfoStandbyMode9i— indicates that the object contains a performance statistic available only on Oracle database server versions 9 and later.

## Configuration group

The Configuration group (oraConf) provides objects for configuring monitors, and objects that report static information about the monitored database instances.

### Installation table

The installation table (oraInstallationTable) provides a list of Oracle installation locations.

Table 231 lists the row objects in oraInstallationTable.

*Table 231. Installation table (oraInstallationTable)*

Object	Description
Index	Uniquely identifies the row.
Path	The full path to the Oracle database installation.

### Instance table

The instance table (oraInstanceTable) provides information about each Oracle instance. This information is obtained from Oracle configuration files or from the system registry.

Table 232 lists the objects in oraInstanceTable.

*Table 232. Instance table (oraInstanceTable)*

Object	Description
DataControl	Controls data collection by the control row:  on(1) - Data collection is enabled  off(2) - Data collection is suspended
GDN	The Global Database Name of the monitored database instance.
Index	Uniquely identifies the row.
InitFile	The initialization file used to configure the monitored database instance.
InstallationIndex	Indicates the Oracle installation to which this row corresponds by referencing a row in the table oraInstallationTable.
LastUpdate	The value of sysUpTime when the monitored database instance was last interrogated for statistics. The value is updated at the start of sampling.

Table 232. Instance table (oraInstanceTable) (continued)

Object	Description
Monitored	<p>Controls the update of statistics from the monitored database instance:</p> <p>yes(1) - Starts an update of statistics, even if the value of oraInstanceMonitored is already yes(1). If an update is already in progress, no action is taken.</p> <p>no(2) - Stops statistics updates.</p>
Password	The password used to connect to the monitored database instance.
SID	The Oracle System Identifier of the monitored database instance.
Status	<p>The current sampling status of this instance. In the event that no connection can be made to the instance (the listener is not running, or the provided credentials are invalid) this field will indicate the error as follows:</p> <p>sampling - A sample is in progress.</p> <p>idle - The agent is not actively sampling.</p> <p>invalidUsernameOrPassword - The username or password is invalid.</p> <p>connectionFailed - Generic connection failure.</p> <p>unableToResolveSID - The instance is not running, or the entry is not valid (configured incorrectly in the system).</p> <p>tnsListenerNotRunning - The listener service is not active.</p> <p>oracleEnvironmentFailed - The agent failed to determine the correct value of \$ORACLE_HOME and the correct location of the OCI shared library. Manually set the environment variable \$ORACLE_HOME to the correct value and restart (not cold boot) the agent.</p> <p>failedToInitialiseOracle - The Oracle library was found, but the agent could not create a server or service context.</p>
UpdateFreq	<p>Sets the interval (in seconds) between updates of statistics for the monitored database instance. If the time taken to retrieve data from the instance is longer than this value, the value is increased so that it is greater than the sample time, and a warning is output to the Netcool/SSM log file.</p> <p>Valid range: 1 to 86400</p>
Username	The database username used to access the monitored database instance. See "Specifying the Oracle database user" on page 347 for information about the privileges necessary for the specified user.

## Instance information table

The instance information table (oraInstanceInfoTable) provides information each Oracle instance. This information requires connectivity to the Oracle instance for it to be determined.

Table 233 lists the objects in oraInstanceInfoTable.

*Table 233. Instance information table (oraInstanceInfoTable)*

Object	Description
ActiveState9i	Indicates the rest state of the monitored database instance. If the state is quiescing(2) or quiesced(3) then the database is not accepting new transactions.
ArchiverState	The state of the archiver for this instance.
ClusterInstance	If the database being monitored is a clustered database, this object specifies the SID of the monitored instance.
DBStatus	Indicates the status of the monitored database instance.
DBVersion	Indicates the version of Oracle RDBMS software running the monitored database instance.
Logins	Indicates whether logins are allowed for this instance.
Parallel	Indicates if this instance is part of an Oracle Application Cluster.
RedoThread	The thread number of the Redo thread for this instance.
Role	The role of the monitored database instance.
Shutdown	Indicates whether a shutdown is pending for this instance.
StartTime	Indicates the time that the monitored database instance was started.
Status	Indicates the start status of this instance.

## Database file table

The database file table (oraDBFileTable) provides information about physical storage used for database instances.

Table 234 lists the row objects in oraDBFileTable.

*Table 234. Database file table (oraDBFileTable)*

Object	Description
AuxName	The auxiliary name for this database file. If no auxiliary name is specified, then the value is NONE.
BlkSize	The block size for this data file.
Blocks	The size in blocks of this database file.
ChkpntChngNum	The change at the last checkpoint.
ChkPntCount	The number of checkpoints made against this database file.
ChkPntTime	The time of the last checkpoint.
CreateChngNum	The initial change number when this data file was created.
CreateSize	The initial size of this file when created.
CreateTime	The time when this data file was created.
Enabled	Indicates how accessible the file is from SQL.
Error	Contains no error if there are no errors, otherwise the error that occurred reading the data file.

Table 234. Database file table (oraDBFileTable) (continued)

Object	Description
Format	The Oracle file format version.
Index	The Oracle specified file number of this data file.
LastChange	Last change number made to this file.
LastChangeTime	The time the last change was made to this file.
Name	The path and name of this database file.
Recover	Indicates where this database file requires recovery.
SizeMB	The current size of this file (in MB).
Status	The status of this database file.
TablespaceIdx	The corresponding oraTableSpaceIndex for the tablespace stored in the data file.
UnrecChngNum	The change number of the last unrecoverable change made to this file.
UnrecChngTime	The time of the last unrecoverable change made to this database file.

## Table space table

The table space table (oraTableSpaceTable) provides a breakdown of defined table spaces for each instance.

Table 235 lists the row objects in oraTableSpaceTable.

Table 235. Table space table (oraTableSpaceTable)

Object	Description
AllocationType	The method of allocation used for extending this tablespace.
BlockSize	The size (in bytes) of a block for this tablespace. This variable is only available in the context of a tablespace on Oracle 9 databases. If this value is 0, see the default block size of the database (oraParameterDBBlockSize).
Contents	The type of data stored in this tablespace.
ExtentCount	Actual number of extents in this tablespace.
ExtentMgmt	The method of extent management employed for this tablespace.
FileCount	The number of data files used to store this tablespace.
FreeBlocks	The number of free database blocks for this tablespace.
FreeExtents	The current number of free extents for this tablespace.
FreeSpace	The number of MB free for this tablespace. This object is deprecated. Use oraTableSpaceFreeSpaceMB instead.
FreeSpaceMB	The number of MB free for this tablespace.
FreeSpaceRatio	The percentage of space available for this tablespace.
Index	A locally assigned index.
InitExtents	Default initial extent size.
LrgExtentMB	The size of the largest extent for this tablespace (in MB).
MaxExtents	Maximum number of extents allowed for a segment in this tablespace.
MinExtents	Minimum number of extents allowed for a segment in this tablespace.

Table 235. Table space table (oraTableSpaceTable) (continued)

Object	Description
Name	The name of this tablespace.
NextExtent	Default incremental extent size.
SmlExtentMB	The size of the smallest extent for this tablespace (in MB).
Status	The current status of this tablespace.
TotalBlocks	The total number of database blocks allocated for this tablespace.
TsNumber	The corresponding tablespace number from Oracle.
UsedBlocks	The current number of blocks used in this tablespace.

## Rollback table

The rollback table (oraRollbackTable) provides statistics on rollback segments.

Table 236 lists the row objects in oraRollbackTable.

Table 236. Rollback table (oraRollbackTable)

Object	Description
AveActive	Average extent size.
AveShrink	The average shrink size.
Extends	The number of times the segment was extended.
Extents	The number of extents in this rollback segment.
Gets	The number of header gets.
HWMSize	The high water mark of the segment size.
Name	The rollback segment name.
OptSize	The optimal size for this rollback segment.
RSSizeMB	The size of this rollback segment (in MB).
Shrinks	Number of times the size of the rollback segment decreased.
Status	The status of this rollback segment.
USN	The rollback segment number for this segment.
Waits	The number of header waits.
Wraps	The number of times the segment was wrapped.
WritesMB	The number of MB written to this rollback segment.
XActs	The number of active transactions in this rollback segment.

## Redo log table

The redo log table (oraRedoLogTable) provides a table of redo log entries.

Table 237 lists the row objects in oraRedoLogTable.

Table 237. Redo log table (oraRedoLogTable)

Object	Description
ArchiveStatus	Indicates if this redo log is included in backups.
ContentStatus	The status of the contents of this redo log.
FileStatus	The status of this redo log file.
FirstChngNum	The change number of the first change to this redo log.

Table 237. Redo log table (oraRedoLogTable) (continued)

Object	Description
FirstTime	The time of the first change to this redo log.
Index	The redo log index corresponds to the group number of the redo log.
MemberCount	The number of members in this log group.
MemberName	The member name of this redo log.
SeqNum	The sequence number for this redo log.
Size	The size (in bytes) of this redo log.
ThreadNum	The log thread number for this redo log.

## Parameter table

The parameter table (oraParameterTable) provides general configuration parameters for each database instance.

Table 238 lists the row objects in oraParameterTable.

Table 238. Parameter table (oraParameterTable)

Object	Description
DBBlockBuffers	The size of database block buffers for the monitored database instance.
DBBlockSize	The size of a database block for the monitored database instance.
MaxCPUs	The maximum number of CPUs available for Oracle to run on.
MaxCursors	The maximum number of cursors that can be created in this instance.
MaxDBFiles	The maximum number of database files that this instance can open.
MaxProcesses	The maximum number of user processes allowed to connect to this instance.
MaxSessions	The maximum number of sessions allowed to be created in the system.
MaxTransactions	The maximum number of concurrent transactions that this instance can process.
MaxTransPerSeg	The maximum number of transactions handled by a rollback segment.
RedoChkptInterval	The redo check point interval for the monitored database instance.
RollbackSegs	The number of rollback segments allocated to this instance.
SortAreaSize	The maximum amount of memory this instance can use for sort.
TimedStatistics	Some statistics require that time statistics be set for them to be collected.



## Database information table

The database information table (oraDBInfoTable) contains information for this instance pertaining to the database.

Table 239 lists the row objects in oraDBInfoTable.

*Table 239. Database information table (oraDBInfoTable)*

Object	Description
ArchiveLogMode	Indicates whether an Archive Log is available for the monitored database instance.
CreateTime	The date and time that the monitored database instance was created.
CtrlFileType	The type of control file for the monitored database instance.
ID	The internal database ID of the monitored database instance.
Name	The internal name for the monitored database instance.
OpenMode	The mode that the database was opened in.
OpenReset	Indicates whether the resetlog option is allowed on the next database open.
StandbyMode9i	The standby mode for the database.

## Query control group

The query control group (oraQueryControl) contains objects that control data collection for particular MIB tables.

Table 240 lists the objects in this group.

*Table 240. Query control group objects (oraQueryControl)*

Object	Description
oraDatafileMonitoring	Controls data collection for the tables oraDBFileTable and oraPerfIODataFileTable:  enabled(1) - Data collection is enabled  disabled(2) - Data collection is disabled
oraLockMonitoring	Controls data collection for the table oraStatLockTable:  enabled(1) - Data collection is enabled  disabled(2) - Data collection is disabled
oraLockQueryHint	Specifies an Oracle optimizer hint to use when executing the potentially expensive lock monitoring query for oraStatLockTable. Applies only if oraLockMonitoring is enabled. When specifying a hint, omit the comment delimiters, /*+ */ and --+, which are normally required in Oracle optimizer hints.
oraSessionMonitoring	Controls data collection for the table oraSessionTable:  enabled(1) - Data collection is enabled  disabled(2) - Data collection is disabled

Table 240. Query control group objects (oraQueryControl) (continued)

Object	Description
oraTablespaceMonitoring	Controls data collection for the tables oraTableSpaceTable and oraPerfIOTablespaceTable:  enabled(1) - Data collection is enabled  disabled(2) - Data collection is disabled

## Statistics group

The Statistics group (oraStat) provides database performance information including sessions, data files, and table space.

### SGA table

The SGA table (oraSGATable) provides a list of SGA statistics for each database instance.

Table 241 lists the row objects in oraSGATable.

Table 241. SGA table (oraSGATable)

Object	Description
BufferCache9i	The size (in bytes) allocated to the Buffer Cache. The Buffer Cache is a feature of Oracle 9; on versions of Oracle that do not support this feature the value of this object is -1.
DBBlockBuffers	The size (in bytes) allocated to database block buffers in the SGA.
FixedSize	Indicates the size (in bytes) of the fixed portion of the SGA.
JavaPoolSize	The size (in bytes) of the Java Pool in the SGA.
JavaPoolUsed	The size (in bytes) used by the Java Pool.
LargePool	The size (in bytes) allocated to the Large Pool. This object is deprecated and has the value 0 on Oracle 8 and above.
LargePoolUsed	The size (in bytes) used by the Large Pool. This object is deprecated and has the value 0 on Oracle 8 and above.
LogBuffer	The configured (requested) size (in bytes) of the redo log buffers in the SGA.
RedoBuffers	The size (in bytes) allocated to database block buffers in the SGA. This value is the actual memory consumption of the redo log buffers, including Oracle's internal guard pages.
SharedPoolReserved	The size (in bytes) reserved for the Shared Pool in the SGA.
SharedPoolSize	The size (in bytes) allocated to the Shared Pool in the SGA.
SharedPoolUsed	The size (in bytes) of the used portion of the Shared Pool.
TotalSize	The total size of the SGA memory area.
TotalUsed	The total amount of the SGA area used.
VariableSize	The size (in bytes) allocated to shared pools in the SGA.

## Buffer pool table

The buffer pool table (oraBufferPoolTable) provides a breakdown of the database blocks that are stored in the Buffer Pool.

Table 242 lists the row objects in oraBufferPoolTable.

*Table 242. Buffer pool table (oraBufferPoolTable)*

Object	Description
BlockCount	Total number of blocks from this file stored in the Buffer Pool.
DistinctBlockCount	Total number of distinct blocks from this file that are stored in the Buffer Pool.

## SGA dictionary table

The SGA dictionary table (oraSGADictionaryTable) provides information about the state of the data dictionary for this instance.

Table 243 lists the row objects in oraSGADictionaryTable.

*Table 243. SGA dictionary table (oraSGADictionaryTable)*

Object	Description
DLMConflicts	Number of Distributed Lock Manager requests that resulted in a conflict with another process.
DLMReleases	Number of times the Distributed Lock Manager was asked to release an object in the Dictionary.
DLMRequests	Number of requests made of the Dictionary by the Distributed Lock Manager.
Entries	The number of entries in the Dictionary.
FixedEntries	Total number of fixed entries.
Flushes	Number of times the Dictionary was flushed to disk.
Gets	Total number of Gets made to the dictionary.
Misses	Total number of dictionary misses.
Modifications	Number of insert/update/delete operations that have been performed on the Dictionary.
ScanCompletes	Number of times the list was completely scanned.
ScanMisses	Number of scans that did not find data in the Dictionary.
Scans	Number of times the dictionary was scanned.
ValidEntries	Total number of Dictionary Entries that are valid.

## Shared pool table

The shared pool table (oraSGASharedPoolTable) provides statistics about the performance of the shared pool for each database instance.

Table 244 lists the row objects in oraSGASharedPoolTable.

*Table 244. Shared pool table (oraSGASharedPoolTable)*

Object	Description
DDCacheSize	The size (in bytes) allocated to the Data Dictionary cache.
LibraryCacheSize	The size (in bytes) allocated to the Library cache.

Table 244. Shared pool table (oraSGASharedPoolTable) (continued)

Object	Description
RsvdFails	The number of requests of the Shared Pool Reserved List that resulted in a fail.
RsvdMisses	The number of requests of the Shared Pool Reserved List that resulted in a miss.
RsvdRequests	The number of requests made of the Shared Pool Reserve List.
SessionHeapSize	The size (in bytes) allocated to the Session heap.
SessionSize	The size (in bytes) allocated to sessions.
SQLAreaSize	The size (in bytes) allocated to the SQL area.

## Session table

The session table (oraSessionTable) contains information about active sessions at the time of sampling.

Table 245 lists the row objects in oraSessionTable.

Table 245. Session table (oraSessionTable)

Object	Description
BlockChanges	The number of block changes caused by this session.
BlockGets	The number of database block gets generated by this session.
BytesReceivedFromClientMB	The number of MB received from the client via SQL*Net.
BytesReceivedFromDBLinkMB	The number of MB received from DBLink via SQL*Net.
BytesSentToClientMB	The number of MB send to the client via SQL*Net.
BytesSentToDBLinkMB	The number of MB sent to DBLink via SQL*Net.
Calls	The number of calls for this session.
CommitCount	The number of commits for this session.
ConsChanges	The number of consistent changes caused by this session.
ConsGets	The number of consistent gets generated by this session.
CPU	The CPU usage of this session as reported by Oracle.
CPUParsePercent	The percentage of CPU used by this session parsing SQL over the last sample interval.
CPUPercent	The percentage of CPU used by this session over the last sample interval.
CPURecursivePercent	The percentage of CPU used by this session parsing recursive SQL over the last sample interval.
CursCacheCount	The number of cached cursors.
CursCacheHits	The number of times a cached cursor was found and an SQL statement parse was avoided.
DiskSorts	The number of sorts generated by this session that required at least one disk write.
HardParses	The number of hard parses (real parses) caused by this session.
LogicalReads	The total number of reads caused by this session.
LogonTime	The connection time of this session.

Table 245. Session table (oraSessionTable) (continued)

Object	Description
Machine	The name of the client machine
MemSorts	The number of sorts generated by this session that we processed in memory.
Name	The Oracle-assigned name for this session.
OSPID	The OS process ID of the client software.
ParseCount	The number of SQL parses generated by this session.
PhysicalSortPct	The percentage of sorts performed on disk as opposed to memory over the last sample interval.
PhysReads	The number of physical reads caused by this session.
Program	The client program name.
RecursiveCalls	The number of recursive calls for this session.
RollbackCount	The number of rollbacks for this session.
SID	The Oracle session identifier for this session.
SortRows	The total number of rows sorted by this session.
Status	The current status of this session.
TotalParses	The total number of parses caused by this session.
UserName	The username used to access oracle.

## Process table

The process table (oraProcessTable) provides a list of Oracle operating system processes for each database instance.

Table 246 lists the row objects in oraProcessTable.

Table 246. Process table (oraProcessTable)

Object	Description
Background	True if this is a background process.
Handles	The number of system handles that this process currently has open.
KernelTime	Total time that this process has spent executing kernel code.
OSPID	The OS process identifier for this process.
PageFaults	The total number of page faults (hard and soft) incurred by this process.
PctSize	Percent physical memory usage.
PGAA11oc9i	The memory allocated by the process.
PGAMax9i	The maximum memory available to this process.
PGAUsed9i	The count of memory currently in use by this process.
PID	The oracle process identifier for this process.
Program	The program being run by this process.
Size	Size of this process in KB.
Terminal	The terminal identifier for this process.
Threads	The maximum memory available to this process.
Username	The OS username that this process runs as.
UserTime	Total time that this process has spent executing user code.

## Performance input/output table

The performance input/output table (oraPerformanceIOWDataFileTable) provides statistics relating to file access for each database file.

Table 247 lists the row objects in oraPerformanceIOWDataFileTable.

*Table 247. Performance input/output table (oraPerformanceIOWDataFileTable)*

Object	Description
AvgReadTime	The average time taken to read from this file.
AvgWriteTime	The average time taken to write to this file.
PhysBlocksRead	The total number of database blocks read from this file.
PhysBlocksWrite	The total number of database blocks written to this file.
PhysReads	The total number of physical reads from this file.
PhysWrites	The total number of writes to this file.

## Performance input/output summary table

The performance input/output table (oraPerformanceIOWDataFileSummaryTable) provides a summary of the file access information.

Table 248 lists the row objects in oraPerformanceIOWDataFileSummaryTable.

*Table 248. Performance input/output summary table (oraPerformanceIOWDataFileSummaryTable)*

Object	Description
BlockChanges	The total number of block changes.
BlockReads	The total number of block reads for all files.
BlockWrites	The total number of block writes for all files.
LogicalReads	The total number of logical reads (reads where the data was already cached).
PhysReads	The total number of physical reads for all files.
PhysWrites	The total number of physical writes for all files.
RedoSize	The allocated redo size.

## Performance input/output tablespace table

The performance input/output tablespace table (oraPerformanceIOWTablespaceTable) provides statistics for tablespace file input/output.

Table 249 lists the row objects in oraPerformanceIOWTablespaceTable.

*Table 249. Performance input/output tablespace table (oraPerformanceIOWTablespaceTable)*

Object	Description
ReadPercentage	The percentage of physical reads to the indexed data file for the indexed tablespace.
WritePercentage	The percentage of block writes from the indexed data file for the index tablespace.

## Statistics lock table

The statistics lock table (oraStatLockTable) provides a summary of locks and locking statistics.

Table 250 lists the row objects in oraStatLockTable.

*Table 250. Statistics lock table (oraStatLockTable)*

Object	Description
Blocking	The lock is blocking another lock.
Duration	Time since current mode was granted.
Index	A locally assigned index for this lock.
ModeGiven	Lock mode in which the session holds the lock.
ModeRequested	Lock mode in which the process requests the lock.
ObjectName	The name of the object being locked.
ObjectOwner	The owner of the object that is locked.
ObjectType	Type of user or system lock.
ResourceID1	Lock identifier 1 (depends on type).
ResourceID2	Lock identifier 2 (depends on type).
Type	Type of user or system lock.

**Tip:** If the Oracle query used to populate this table creates a performance issue, use the oraLockQueryHint object to specify a query optimizer hint.

## Statistics latch table

The statistics latch table (oraStatLatchTable) provides a summary of latches and latch statistics.

Table 251 lists the row objects in oraStatLatchTable.

*Table 251. Statistics latch table (oraStatLatchTable)*

Object	Description
Gets	Number of times obtained a wait.
IMMGets	Number of times obtained without a wait.
IMMMisses	Number of times failed to get without a wait.
IMMWTWRatio	The ratio of latch gets that resulted in a wait.
Index	A locally assigned index for this latch.
Misses	Number of times obtained a wait but failed on the first try.
Name	Latch name.
Number	Latch number.
Sleeps	Number of times slept when wanted a wait.
SpinGets	Gets that missed first try, but succeeded on spin.
WaitsHeld	Number of waits while holding a different latch.

## Statistics wait table

The statistics wait table (oraStatWaitTable) provides a summary of wait events and related statistics.

Table 252 lists the row objects in oraStatWaitTable.

*Table 252. Statistics wait table (oraStatWaitTable)*

Object	Description
Avg	The average time waited for this event, in hundredths of a second.
Event	The name of the wait event.
Index	A locally assigned index.
Time	The total time waited for this event, in hundredths of a second.
Timeouts	The total number of time-outs for this event.
Waits	The total number of waits for this event.

## Statistics wait table for 64-bit

The 64-bit statistics wait table (ora64StatWaitTable) provides a summary of wait events and related statistics for 64-bit values.

Table 253 lists the row objects in ora64StatWaitTable. These row objects use Counter64 objects rather than the Integer32 objects used in the 32-bit oraStatWaitTable. The Counter64 objects may not be compatible with SNMPv1.

*Table 253. Statistics wait table (ora64StatWaitTable)*

Object	Description
Avg	The average time waited for this event, in hundredths of a second.
Event	The name of the wait event.
Index	A locally assigned index.
Time	The total time waited for this event, in hundredths of a second.
Timeouts	The total number of time-outs for this event.
Waits	The total number of waits for this event.

## Statistics application table

The statistics application table (oraStatApplicationTable) provides application information relating to the performance of the Oracle instances.

Table 254 lists the row objects in oraStatApplicationTable.

*Table 254. Statistics application table (oraStatApplicationTable)*

Object	Description
BBlockChanges	The total number of block changes processed by this instance.
BlockGets	The total number of block gets processed by this instance.
Commits	The total number of commit operations executed on this instance.
ConsChange	The total number of consistent changes processed by this instance.
ConsGets	The total number of consistent gets processed by this instance.
CPU Parse	The CPU time (in hundredths of seconds) this instance has spent parsing SQL statements.



Table 254. Statistics application table (oraStatApplicationTable) (continued)

Object	Description
CPU Parse Recursive	The CPU time (in hundredths of seconds) this instance has spent parsing recursive SQL statements.
Current Logons	The current number of logons for this instance.
Disk Sorts	The total number of sorts, that generated at least one disk write, processed by this instance.
Hard Parses	The number of hard parses performed by this instance.
Parse Count	The number statement parses performed by this instance, including statements where a previous parse was cached.
Recurse Calls	The total number of recursive calls executed on this instance.
Redo Entries	The number of redo entries generated by transactions on this instance.
Redo Switches	The number of times the redo logs were switched.
Rollbacks	The total number of rollback operations executed on this instance.
Rows Scanned	The number of rows scanned.
Session Max	The high water mark for sessions active on this instance.
Sessions	The current number of sessions active on this instance.
Sort Rows	The total number of rows sorted by this instance.
Tbl Cont Fetch	The number of continued rows fetched.
Tbl Rowid Fetch	The number of rows fetched by row ID.
Total Logons	The total number of logons (cumulative) for this instance.
User Calls	The total number of user calls executed on this instance.

## Statistics library cache table

The statistics library cache table (oraStatLibCacheTable) provides library cache statistics for each database instance.

Table 255 lists the row objects in oraStatLibCacheTable.

Table 255. Statistics library cache table (oraStatLibCacheTable)

Object	Description
GetHitRatio	The ratio of get hits to get requests.
GetHits	The number of get requests that resulted in a cache hit
Gets	The number of get requests to the library cache.
Namespace	The namespace of the library cache.
PinHitRatio	The ratio of pin hits to pin requests.
PinHits	The number of pin requests which resulted in a cache hit.
Pins	The number of issued pin requests.
Reloads	The number of times library objects have been reloaded.

### Statistics queue cache table

The statistics queue cache table (oraStatQueueTable) provides message queue statistics for each database instance.

Table 256 lists the row objects in oraStatQueueTable.

*Table 256. Statistics queue cache table (oraStatQueueTable)*

Object	Description
AverageWait	The average time (in hundredths of seconds) that a message waits in the queue.
Queued	The total number of responses that have been queued in this queue.
Type	A unique identifier for the row, formed by the queue type and the value of the PADDR column.
Wait	The total waiting time (in hundredths of seconds) for all responses in this queue.

### Dispatcher statistics table

The dispatcher statistics table (oraStatDispatcherTable) provides statistics about Oracle dispatcher processes.

Table 257 lists the row objects in oraStatDispatcherTable.

*Table 257. Dispatcher statistics table (oraStatDispatcherTable)*

Object	Description
Busy	The total busy time (in hundredths of seconds) for this dispatcher.
BusyRatio	The percentage busy time of this dispatcher.
Idle	The total idle time (in hundredths of seconds) for this dispatcher.
Network	A unique identifier formed by the dispatcher queue type and the PADDR column of the dispatcher row.

### Parallel servers statistics table

The parallel servers statistics table (oraStatPQTable) provides statistics about Oracle parallel servers.

Table 258 lists the row objects in oraStatPQTable.

*Table 258. Parallel servers statistics table (oraStatPQTable)*

Object	Description
ServersBusy	The number of currently busy servers on this instance.
ServersHighWater	The number of active servers that have partaken in one or more operations for this instance.
ServersIdle	The number of currently idle servers on this instance.

## Batch job statistics table

The batch job statistics table (oraStatBatchJobTable) provides current batch jobs for each monitored instance.

Table 259 lists the row objects in oraStatBatchJobTable.

*Table 259. Batch job statistics table (oraStatBatchJobTable)*

Object	Description
JobBroken	true if this job is broken, false otherwise.
JobId	The identifier for this job assigned by the database.
JobInterval	The date function which is evaluated to determine the next run date.
JobLastRun	The date and time this job was last successfully executed.
JobNextRun	The date and time this job will next be executed.
JobPrivilegeUser	The name of the user whose privileges will apply to this job when run.
JobSubmitUser	The name of the user who was logged in when this job was submitted.
JobWhat	The first 1024 characters of the PL/SQL block to execute.

## Replication statistics table

The replication statistics table (oraStatReplicationTable) provides statistics about Oracle replication.

Table 260 lists the row objects in oraStatReplicationTable.

*Table 260. Replication statistics table (oraStatReplicationTable)*

Object	Description
BrokenJobs	The number of replication refresh jobs which are broken.
QueuedCalls	The current size of the defcall queue.
QueuedErrors	The current size of the deferror queue.
QueuedTransactions	The current size of the deftran queue.

## Efficiency group

The Efficiency group (oraEfficiency) provides per-sample, delta average values suitable for trending and reporting for performance metrics in the Statistics group.

### Efficiency memory table

The efficiency memory table (oraEffMemTable) provides statistics about the efficiency of memory usage as a percentage. The subagent requires a minimum of two samples to calculate the efficiency.

Table 261 lists the row objects in oraEffMemTable.

*Table 261. Efficiency memory table (oraEffMemTable)*

Object	Description
BufCache	Percentage of buffer requests already in the buffer cache.
CursCache	Percentage of parse requests where the cursor is already in the cache.

Table 261. Efficiency memory table (oraEffMemTable) (continued)

Object	Description
DDFlushes	Number of times that the Data Dictionary was flushed to disk each second.
DDGets	Rate per second of requests made to the Data Dictionary.
DDHitRatio	The percentage of gets to the Data Dictionary that resulted in a hit.
DDMisses	Rate of data dictionary requests resulting in a cache miss (misses/second).
DDMissRatio	The percentage of gets to the Data Dictionary that resulted in a miss.
DDMods	Number of modifications (inserts/updates/deletions) made to the Data Dictionary per second.
DDScanCompletes	Number of complete scans made, each second, of the Data Dictionary.
DDScanMisses	Number of scan requests per second that failed to find data in the cache.
DDScanMissRatio	The percentage of scans of the data dictionary that resulted in a scan miss.
DDScans	Number of scans per second made of the Data Dictionary.
DLMConflicts	Number of conflicts reported by the Distributed Lock Manager each second.
DLMReleases	Number of releases reported by the Distributed Lock Manager each second.
DLMRequests	Number of requests made to the Distributed Lock Manager each second.
FreeSP	Percentage of the free pool that is free.
LibCache	Percentage of parse requests where the cursor is already in the cache.
RsvdHit	Percentage of hits on the shared pool reserved list.
UGACurrent	The current UGA memory usage.
UGAHW	Indicates the high water mark (bytes) for session memory usage.

## Efficiency storage table

The efficiency storage table (oraEffStorageTable) provides statistics about the efficiency of storage usage as a percentage. The subagent requires a minimum of two samples to calculate these values.

Table 262 lists the row objects in oraEffStorageTable.

Table 262. Efficiency storage table (oraEffStorageTable)

Object	Description
BlockChanges	The number of block changes per second.
LogicalReads	The number of logical reads per second.
PhysBReads	The number of physical block reads per second.
PhysBWrites	The number of physical writes per second.
PhysReads	The number of physical reads per second.
PhyWrites	The number of physical writes per second.

## Efficiency application table

The efficiency application table (oraEffAppTable) provides application usage rates over the last sample period as a percentage. The subagent requires a minimum of two samples to calculate these values.

Table 263 lists the row objects in oraEffAppTable.

Table 263. Efficiency application table (oraEffAppTable)

Object	Description
ActiveSessions	The number of active sessions for the monitored database instance.
AvgWait	The average time that a message waits in the message queue.
BlockChangeRate	The number of block changes per second.
BlockChangeTransRatio	The average number of block changes per transaction.
BlockFreeWaits	The number of times a free block request did not need to wait.
BlockFreeWaitsPct	The percentage of block free waits to block gets.
BlockGetRate	The number of block gets per second.
BlockGetsTransRatio	The average number of block gets per transaction. If a database is mainly read oriented, the number will be high, as there are few commit or rollback operations.
CallsParseRatio	The average number of calls generated for each parse.
CallsTransRatio	The average number of user calls per transaction. If a database is mainly read oriented, the number will be high, as there are few commit or rollback operations.
CommitRate	The number of commits per second.
ConsChangeRate	The number of consistent changes per second.
ConsGetRate	The number of consistent gets per second.
CPUParse	The percentage of CPU time spent parsing SQL statements.
CPUParseRecursive	The percentage of CPU time spent parsing recursive SQL statements.
DiskSortRate	The number of sort operations, that required at least one disk write, per second.
DiskSortRatio	The percentage of all sorts that required writes to disk.
HardParseRate	The number of hard parses per second.
HardParseRatio	Percentage of SQL statements executed where a statement had to be loaded into the shared pool.
IOBusy	The percentage of time spent waiting for I/O.
LockedSessionPct	The percentage of active sessions that are waiting on a lock.
LockWaitSessions	The number of sessions waiting on a lock for the monitored database instance.
LogonRate	The number of logons processed per second.
MaxProcessPct	The percentage of maximum oracle processes in use.
MaxSessionPct	The percentage of maximum sessions in use.
MemSortRate	The number of sort operations, performed in memory, per second.
OpenCursorPct	The percentage of maximum cursors currently open.

Table 263. Efficiency application table (oraEffAppTable) (continued)

Object	Description
ParseRate	The number of parse calls per second.
ParseTransRatio	The average number of parse calls per transaction. If a database is mainly read oriented, the number will be high, as there are few commit or rollback operations.
RecursiveCallRate	The number of recursive calls per second.
RecursiveRatio	The percentage of user calls that were recursive.
RedoEntryRate	The number of redo entries generated per second.
RedoNoWaitRatio	The percentage of redo entries that generated a log switch.
RedoSwitchRate	The number of redo log switches per second.
RollbackRate	The number of rollbacks per second.
RollbackRatio	The percentage of transactions that ended with a rollback. If the database is mainly used for reading, this ratio will be low.
RowSortRate	The number of rows sorted per second.
RowsScanRate	The number of rows scanned per second.
ScanFetchRatio	The ratio of rows fetched to rows scanned.
TblContFetchRate	Number of continued rows fetched per second.
TblRowidFetchRate	Number of rows fetched by row id per second.
TotalSessions	The total number of sessions for the monitored database instance.
TransactionRate	The number of transactions per second. A transaction is taken to be a commit or a rollback. Sessions that cause a large number of 'read only' style calls will have a low transaction rate.
UserCallRate	The number of user calls per second.
WaitSessions	The number of sessions waiting (other than for a lock) for the monitored database instance.

## SQL control table

The SQL control table (oraEffSQLControlTable) provides a mechanism to determine the least efficient SQL statements that are running on a given database instance. The oraEffSQLControlTable enables you to select the database instance you want to monitor by specifying the InstanceIdx parameter. This value corresponds to the index in the oraInstanceTable for the database instance that you wish to monitor.

The SortBy parameter enables you to select one of four variables to use for sorting the results in the oraEffSQLDataTable. The RequestSize parameter allows you to specify the number of top SQL statements that you want displayed in the oraEffSQLDataTable.

Table 264 lists the row objects in oraEffSQLControlTable.

Table 264. SQL control table (oraEffSQLControlTable)

Object	Description
ActivateTime	The time that this entry was activated.

Table 264. SQL control table (oraEffSQLControlTable) (continued)

Object	Description
DataControl	Controls the monitoring status of the control row:  on(1) - Monitoring is enabled  off(2) - Monitoring is disabled
Description	A description for this entry.
GrantedSize	The maximum number of SQL statements that will be retrieved.
Index	The unique index for this entry.
InstanceIdx	The index of the oracle instance that will be monitored.
Owner	The owner of this entry.
RequestSize	The desired maximum number of statements to retrieve.
SampleInterval	The interval at which the agent polls the instance for SQL statements.
SortBy	The parameter that will be used to sort SQL statements to determine those that are least efficient.
Status	The SNMPv2 row status for this entry.
StmtType	Currently all statement types are monitored.

## SQL data table

The SQL data table (oraEffSQLDataTable) provides a TopN style list of the least efficient SQL statements processed by an Oracle instance. Sorting and the number of statements is specified in the oraEffSQLControlTable.

Table 265 lists the row objects in oraEffSQLDataTable.

Table 265. SQL data table (oraEffSQLDataTable)

Object	Description
BGPE	The average number of block gets per execution of this statement.
BGPR	The average number of block gets per row for this statement.
BufferGets	The number of buffer gets generated by this statement.
DiskRead	The number of disk reads generated by this statement.
DRPE	The average number of disk reads per execution of this statement.
Executions	The number of times this statement has been executed.
Index	A locally assigned index. Rows are ordered by the variable defined in oraEffSQLControlSortBy.
ParseCalls	The number of parse calls generated by this statement.
PCPE	The average number of parse calls per execution of this statement.
RowsProcessed	The total number of rows processed by this statement.
Sorts	The number of sorts generated by this statement.
Statement	The first 1000 characters of this SQL statement.

## Library cache efficiency table

The library cache efficiency table (oraEffLibCacheTable) provides library cache statistics over the last sample period for each monitored instance.

Table 266 lists the row objects in oraEffLibCacheTable.

*Table 266. Library cache efficiency table (oraEffLibCacheTable)*

Object	Description
GetHitRatio	The ratio of get hits to get requests for the last sample interval.
GetHits	The number of get requests during the last sample interval that resulted in a cache hit.
Gets	The number of get requests to the library cache during the last sample interval.
Namespace	The namespace of the library cache.
PinHitRatio	The ratio of pin hits to pin requests for the last sample interval.
PinHits	The number of pin requests during the last sample interval that resulted in a cache hit.
Pins	The number of pin requests issued during the last sample interval.
Reloads	The number of times library objects have been reloaded during the last sample interval.

## Queue efficiency table

The queue efficiency table (oraEffQueueTable) provides library cache statistics over the last sample period for each monitored instance.

Table 267 lists the row objects in oraEffQueueTable.

*Table 267. Queue efficiency table (oraEffQueueTable)*

Object	Description
AverageWait	The average time that a message waited in the queue (in hundredths of seconds) during the last sample interval.
Queued	The total number of responses that have been queued in this queue during the last sample.
Type	A unique identifier for the row. Formed by the queue type and the value of the PADDR column.
Wait	The amount of time messages waited in the queue during the last sample interval.

## Dispatcher efficiency table

The dispatcher efficiency table (oraEffDispatcherTable) provides statistics about Oracle dispatcher processes for the last sample interval.

Table 268 lists the row objects in oraEffDispatcherTable.

*Table 268. Dispatcher efficiency table (oraEffDispatcherTable)*

Object	Description
Busy	The busy time for this dispatcher (in hundredths of seconds) during the last sample interval.
BusyRatio	The percentage busy time of this dispatcher during the last sample interval.



Table 268. Dispatcher efficiency table (oraEffDispatcherTable) (continued)

Object	Description
Idle	The idle time for this dispatcher (in hundredths of seconds) during the last sample interval.
Network	A unique identifier formed by the dispatcher queue type and the PADDR column of the dispatcher row.

## Wait efficiency table

The wait efficiency table (oraEffWaitTable) provides statistics about wait events for each session for the last sample interval.

Table 269 lists the row objects in oraEffWaitTable.

Table 269. Wait efficiency table (oraEffWaitTable)

Object	Description
Avg	The average time waited for this event, in hundredths of a second.
Event	The name of the wait event.
Ratio	The percentage of total time waited for this event over the last sample interval.
Time	The total time waited for this event, in hundredths of a second.
Timeouts	The total number of timeouts for this event.
Waits	The total number of waits for this event.

## Parallel query efficiency table

The parallel query efficiency table (oraEffPQTable) provides efficiency statistics about Oracle parallel query servers.

Table 270 lists the row objects in oraEffPQTable.

Table 270. Parallel query efficiency table (oraEffPQTable)

Object	Description
AverageCPU	The average percentage of CPU time this server has used to process queries.
CPU	The percentage of CPU time this server has used to process queries for this sample interval.
SlaveName	The name of the parallel execution server.



---

## Chapter 34. Netcool/ASM for Microsoft SQL Server

The Netcool/ASM for Microsoft SQL monitors performance metrics for the server, database configuration, internal processes, and locks as well as SQL statistics and available buffer, cache, and memory.

Netcool/ASM for Microsoft SQL Server consists of the `sqlsvr` subagent and the `sqlsvr` MIB module. It monitors the activity of selected instances of the Microsoft SQL Server database and provides statistical and database performance information. It can automatically generate alarms based on the value of particular performance metrics.

---

### Component files

The `sqlsvr` subagent and MIB module are comprised of a set of component files.

Table 271 lists the `sqlsvr` subagent and MIB module component files and their installed locations.

*Table 271. Netcool/ASM for Microsoft SQL Server component files*

File	Location	Description
<code>sqlsvr.dll</code>	<code>bin</code>	Binary implementation.
<code>sqlsvr-mib.mib</code>	<code>mibs</code>	MIB definition document.
<code>sqlsvr.oid</code>	<code>config/oid</code>	Object identifier file.
<code>sqlsvr.cfg</code>	<code>config</code>	Sample configuration file.

---

### Guidelines

To use the `sqlsvr` subagent, first load it on Netcool/SSM.

To load the subagent, use the following configuration command:

```
subagent load sqlsvr
```

### General monitoring procedure

To monitor an SQL Server installation using the `sqlsvr` subagent, create a row in `sqlsvrServerTable` identifying the server that you wish to monitor.

#### Procedure

1. Set `sqlsvrServerStatus` to `CreateandWait(5)`.
2. Set `sqlsvrServerName` to `localhost`.
3. Set `sqlsvrServerAuthMethod` as required.
4. Set `sqlsvrServerUserName` and `sqlsvrServerPassword` if required.
5. Set the `sqlsvrServerMonitorServer` to `on(1)` or `off(2)` as required.
6. Activate monitoring of the selected SQL server by setting `sqlsvrServerStatus` to `active(1)`.

## Events

The sqlsvr subagent does not generate events, however you can set up the genalarm subagent to monitor thresholds for objects in the sqlsvr MIB using the sqlsvr.cfg file.

The sqlsvr.cfg file specifies a default set of objects and thresholds for the sqlsvr MIB that can be monitored by the genalarm subagent. For each object specified in the file, a row is created automatically in the genAlarm MIB. Table 272 lists the objects that are specified in the sqlsvr.cfg file with a description of the metric being monitored.

*Table 272. Monitored SQL Server objects*

Object	Monitored Metric
sqlsvrBufferMgrBuffCacheHitRatio	SQL Server Buffer Manager: Cache Hit Ratio
sqlsvrBufferMgrLazyWrite	SQL Server Buffer Manager: Lazy Writes/second
sqlsvrBufferMgrPageWrites	SQL Server Buffer Manager: Page Writes/second
sqlsvrDbPerfLogUtilized	SQL Server DB Perf: %Utilized
sqlsvrDbStatsPcntUsed	SQL Server Database Stats: Percent Used
sqlsvrInternalProcBlocked	SQL Server Internal Process: Blocked
sqlsvrInternalProcCPU	SQL Server Internal Process: CPU
sqlsvrLocksDeadlocks	SQL Server Locks: Deadlocks
sqlsvrLocksLockWaits	SQL Server Locks: Waits/sec
sqlsvrSQLStatsSQLComp	SQL Server SQL Stats: SQL Compilations/second
sqlsvrSvrProcessPrivilegedTime	SQL Server Process: %Privileged Time
sqlsvrSvrStatsPhysErr	SQL Server Server Stats: Physical Disk Read/Write Errors
srDiskQueueLength	SQL Server Physical Disk Queue
srDiskUtilization	SQL Server Physical Disk Utilization
srSystemContextSwitchCount	SQL Server Context Switches/second
srSystemRunQueueLen	SQL Server System Run Queue

**Note:** You can edit the sqlsvr.cfg file to set up thresholds that are suitable for your system.

To use the sqlsvr.cfg file to create entries in the genAlarm MIB and monitor thresholds in the SQL Server MIB, create a row in the agentConfigTable for the sqlsvr.cfg file. When the row is activated, the sqlsvr.cfg file is executed. To ensure the sqlsvr.cfg file has been executed and view the objects being monitored, check that the appropriate rows have been created in the genAlarm MIB.

**Note:** To ensure the sqlsvr.cfg file is loaded when the agent is instructed to restore values, you need to save the agentConfigTable state to the agent.state file.

## Server connections

Connections between the sqlsvr subagent and the monitored SQL Server are affected by factors such as availability of the SQL Server and network failures.

The control row object sqlsvrServerConnectionStatus indicates the state of each control row's connection to the monitored SQL Server. Whenever a connection fails, or is re-established after failure, the value of sqlsvrServerConnectionStatus changes accordingly. The sqlsvr subagent automatically attempts to re-connect to SQL Server at 60-second intervals (this interval is not configurable).

## Server aliases

If the target SQL Server was installed with a non-default instance name, you must create a server alias for it. The sqlsvr subagent uses the server alias to collect performance data. You can create a server alias by defining a named pipe with the default pipe name.

To determine the default pipe name:

1. On the target SQL Server, start the Server Network Utility.
2. Select the **General** tab, then from the **Enabled protocols** list select **Named Pipes**, and click **Properties....**

The default pipe name is displayed in the **Default pipe** field.

3. Copy the default pipe name.

To configure the server alias:

1. On the target SQL Server, start the Client Network Utility.
2. Select the **Alias** tab and click **Add**.  
The Add Network Library Configuration dialog is displayed.
3. Set the **Server alias** field to localhost.
4. In the **Network Libraries** group, select Named Pipes.
5. In the **Connection Parameters** group, set the **Pipe name** field to the default pipe name determined previously.
6. Click **OK**.

The Add Network Library Configuration dialog closes.

7. In the Client Network Utility, click **OK**.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Server table

The sqlsvr commands create rows in the server table (sqlsvrServerTable).

The general syntax of these commands is:

```
sqlsvr property=value  
sqlsvr create [property=value ...]  
sqlsvr reset
```

Table 273 lists the properties supported in these commands.

*Table 273. Configuration command parameters - sqlsvr*

Property	Type	Description	Sets MIB object
authmethod	enum	Sets the authentication method for accessing the SQL server:  nt  sql	AuthMethod
monitor	enum	Activates monitoring of the SQL server:  off  on	MonitorServer
password	string	The password supplied when connecting to the SQL server.	Password
svrname	string	The name of the monitored server.	ServerName
username	string	The username supplied when connecting to the SQL server.	UserName

---

## MIB module

The sqlSvr MIB is a subtree of networkharmoni(1977).

The subtree is shown in Figure 63 on page 381.

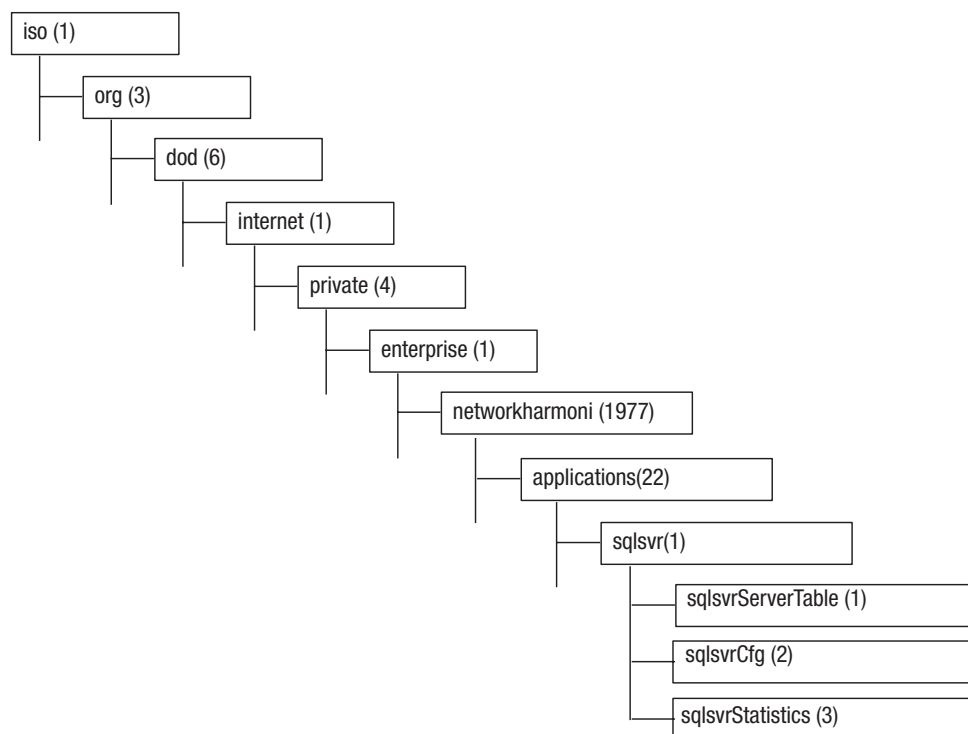


Figure 63. OID tree diagram of the sqlsvr MIB module

This section provides a summary of the objects defined in the sqlSvr MIB module. For detailed information on all objects in the module, see the sqlsvr-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB tables

The sqlSvr MIB tables provide information about the activities and performance of an SQL Server database instance.

### Server table

The server table (sqlsvrServerTable) contains a list of control information for SQL server monitoring.

Table 274 lists the row objects in sqlsvrServerTable.

Table 274. Server table (sqlsvrServerTable)

Row object	Description
AuthMethod	Controls the authentication method used for accessing the SQL server:  nt (1) - The username and password specified are used as a NT username and password to connect to the server  sql (2) - The username and password are used as an SQL Server login

Table 274. Server table (sqlsvrServerTable) (continued)

Row object	Description
ConnectionStatus	Reports the connection status of this control row:  connecting(1) - The agent is attempting to establish a connection to the SQL Server specified  connected(2) - The SQL Server specified in this row has been successfully connected to and is being monitored  disconnected(3) - The agent does not have a connection to the SQL Server
CreateTime	The value of sysUpTime when this entry was last activated. This can be used by the management station to ensure that the entry has not been deleted and recreated between polls.
Description	A comment describing the configuration of this row.
Index	Uniquely identifies this row in the table.
LastUpdate	Stores the value of sysUpTime at which data was last collected from the SQL Server.
MonitorServer	Controls the monitoring status of this control row:  on(1) - The SQL server is monitored  off(2) - The SQL server specified in this row is not monitored
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Password	The password for the user specified in sqlsvrServerUserName for connecting to the SQL server.
SQLServerName	Specifies the name or IP address of the monitored SQL server. Set this object to localhost or 127.0.0.1.
Status	The status of this row. An entry may not exist in the active state unless all objects in the entry have an appropriate value. If this object is not equal to active(1), all associated entries in the other sqlsvr monitoring tables are deleted.
UpdateInterval	Sets the interval (in seconds) at which the SQL Server is monitored and the data tables are updated.
UserName	The name of the user to connect to the SQL Server as. This object is only used for SQL authentication. For Windows authentication, leave it empty. Windows authentication uses the userid of the agent's process.

## Server description table

The server description table (sqlsvrSvrDescTable) stores basic attributes of the monitored SQL servers.

Table 275 describes the objects in sqlsvrSvrDescTable.

Table 275. Server description table (sqlsvrSvrDescTable)

Row object	Description
Contact	The contact individual or organization registered at the time of installation of the particular SQL Server.



Table 275. Server description table (sqlsvrSvrDescTable) (continued)

Row object	Description
IsClustered	If the particular SQL Server has been configured in a failover cluster:  notclustered(0)  clustered(1)  unknown(2)
Product	The product name of the database server.
ServerName	The name of the server running this particular instance of SQL Server.
StartTime	The date and time when the particular SQL Server instance was started.
Status	SQL Server status as follows:  initialized(1)  ready(2)  running(3)  standby(4)  terminated(5)  waiting(6)  transitioning(7)  unknown(8)
Uptime	The amount of time (ms) that this particular instance of SQL Server has been running.
Version	The version of SQL Server running for this particular instance.

## Configuration table

The configuration table (sqlsvrSvrCfgTable) lists the configuration options available for the monitored SQL servers.

Table 276 describes the objects in sqlsvrSvrCfgTable.

Table 276. Configuration table (sqlsvrSvrCfgTable)

Row object	Description
OptIndex	Index for the different configuration options.
OptMax	The maximum value for the configuration option.
OptMin	The minimum value for the configuration option.
OptName	The name of the configuration option.
Run	The current value set for the configuration option.
Value	The value set for the configuration option using the stored procedure sp_configure.

## Database description table

The database description table (sqlsvrDbDescTable) stores basic attributes of the monitored SQL servers.

Table 277 describes the objects in sqlsvrDbDescTable.

*Table 277. Database description table (sqlsvrDbDescTable)*

Row object	Description
CreateDate	The create date and time of the specific database.
Creator	The username of the creator of a particular database.
DbID	The unique ID of the SQLServerDBidx.
DbName	The name of the database.
Lastdifferential	Not implemented.
Lastfilegroup	Not implemented.
Lastfull	Not implemented.
Lasttranslog	Not implemented.
Reserv	Not implemented.

Table 277. Database description table (sqlsvrDbDescTable) (continued)

Row object	Description
Status	<p>The status of the database, indicated by a series of bit flags. Multiple bits may be set at the same time. Some of the flags available are:</p> <p>0x00000001= autoclose</p> <p>0x00000004= select into/bulkcopy</p> <p>0x00000008= trunc. log on chkpt</p> <p>0x00000010= torn page detection</p> <p>0x00000020= loading</p> <p>0x00000040= pre recovery</p> <p>0x00000080= recovering</p> <p>0x00000100= not recovered</p> <p>0x00000200= offline</p> <p>0x00000400= read only</p> <p>0x00000800= dbo use only</p> <p>0x00001000= single user</p> <p>0x00004000= ANSI null default</p> <p>0x00008000= emergency mode</p> <p>0x00010000= concat null yields null</p> <p>0x00020000= recursive triggers</p> <p>0x00100000= default to local cursor</p> <p>0x00400000= autoshrink</p> <p>0x00800000= quoted identifier</p> <p>0x02000000= cursor close on commit</p>
Status continued	<p>0x04000000= ANSI nulls</p> <p>0x10000000= ANSI warnings</p> <p>0x20000000= full text enabled</p> <p>0x40000000= cleanly shutdown</p>
Sz	Not implemented.
Unalloc	Not implemented.
Unused	Not implemented.
Useddata	Not implemented.
UsedIdx	Not implemented.

## Database configuration table

The database configuration table (sqlsvrDbCfgTable) lists the configuration options available for the monitored SQL Servers.

Table 278 describes the objects in sqlsvrDbCfgTable.

*Table 278. Database configuration table (sqlsvrDbCfgTable)*

Row object	Description
OptIndex	Index for the different database configuration options.
Option	The name of the database option which is set.

## SQL Server database backup table

The server database backup table (sqlsvrDbBackupTable) describes the most recent backup sets for the databases for each backup type available.

Table 279 describes the objects in sqlsvrDbBackupTable.

*Table 279. SQL Server database backup table (bacsqsvrDbBackupTable)*

Row object	Description
Desc	The description of the backup set.
Expiry	The date and time the backup set expires.
Index	An index into the different backup sets.
LastCompleted	The date and time when the backup operation was completed.
Name	The name of the backup set.
Size	The size of the backup set in bytes.
Type	Backup type.

## SQL Server jobs table

The server jobs table (sqlsvrJobsTable) describes the memory usage of the database system.

Table 280 describes the objects in sqlsvrJobsTable.

*Table 280. SQL Server jobs table (sqlsvrJobsTable)*

Row object	Description
Instance	Uniquely identifies each job in the database.
Name	The name of the job.
OperatorIdEmailed	The ID of the operator who was notified when the job completed.
OperatorIdNetSent	The ID of the operator who was notified by a message when the job completed.
OperatorIdPaged	The ID of the operator who was notified by pager when the job completed.
RetriesAttempted	Number of times SQL server agent attempted execution of the step. A value of 0 indicates that the step executed successfully on the first attempt or that no retry attempts were specified for the job step.
RunDate	The date on which the job was completed, displayed in the format YYYYMMDD.

Table 280. SQL Server jobs table (sqlsvrJobsTable) (continued)

Row object	Description
RunDuration	The time required for completion of the job, displayed in the format HHMMSS.
RunStatus	The execution status of the job.
RunTime	The time at which the job was started, displayed in the format HHMMSS.
ServerName	The name of the target server.
SqlMessage	The text content of the SQL server message raised by the job (where applicable).
SqlMessageId	The message number of the SQL server message raised by the job (where applicable).
SqlSeverity	The severity of the SQL server message raised by the job (where applicable).
StepId	The ID of the job step.
StepName	The name of the job step.

The sqlsvrJobsLogSize object sets the maximum number of rows allowed in the sqlsvrJobsTable per sqlsvrServerTable control row.

### SQL Server jobs summary table

The server jobs summary table (sqlsvrJobsSummaryTable) provides summary information about SQL server jobs.

Table 282 on page 388 describes the objects in sqlsvrJobsSummaryTable.

Table 281. SQL Server jobs summary table (sqlsvrJobsSummaryTable)

Row object	Description
CurrentInstance	Uniquely identifies a job in the database.
CurrentOperatorIdEmailed	The ID of the operator notified when the job completed.
CurrentOperatorIdNetSent	The ID of the operator notified by a message when the job completed.
CurrentOperatorIdPaged	The ID of the operator notified by pager when the job completed.
CurrentRetriesAttempted	The number of times the SQL Server Agent attempted execution of the step. A value of 0 indicates that the step executed successfully on the first attempt or that no retry attempts were specified for the job step.
CurrentRunDate	The date when the job was completed, in the format YYYYMMDD.
CurrentRunDuration	The amount of time required for the completion of the job, in the format HHMMSS.

Table 281. SQL Server jobs summary table (sqlsvrJobsSummaryTable) (continued)

Row object	Description
CurrentRunStatus	The execution status of the job:  failed(1)  succeeded(2)  retry(3)  canceled(4)  inprogress(5)
CurrentRunTime	The time when the job was started, in the format HHMMSS.
CurrentServerName	The target server name.
CurrentSqlMessage	The text contained in the SQL server message currently raised by the job, if applicable.
CurrentSqlMessageId	The SQL server message number of the message raised by the step, where applicable.
CurrentSqlSeverity	The severity of the SQL Server message raised by the step, where applicable.
CurrentStepId	The identifier of the current job step.
CurrentStepName	The name of the current job step.
Name	The name of the job.

## SQL Server statistics table

The server statistics table (sqlsvrSvrStatsTable) provides general SQL Server information.

Table 282 describes the objects in sqlsvrSvrStatsTable.

Table 282. SQL Server statistics table (sqlsvrSvrStatsTable)

Row object	Description
CPUBusy	The time (ms) that the CPU has spent working for this particular server since its start.
Idle	The time (ms) that this particular server has been idle since its start.
I/O	The time (ms) that this particular server has spent performing input and output operations since its start.
PacketsErr	The number of network packet errors that have occurred on this server's connections since its start.
PacketsRecv	The number of input packets read from the network by this particular server since its start.
PacketsSent	The number of output packets written to the network by this particular server since its start.
PhysErr	The number of physical disk read/write errors encountered by this particular server since its start.
PhysRead	The number of physical disk reads (not cache reads) completed by this particular server since its start.
PhysWrite	The number of physical disk writes (not cache writes) completed by this particular server since its start.

## SQL Server process table

The server process table (sqlsvrSvrProcessTable) displays some of the process statistics for the SQL Server process.

Table 283 describes the objects in sqlsvrSvrProcessTable.

*Table 283. SQL Server process table (sqlsvrSvrProcessTable)*

Row object	Description
PrivilegedTime	The amount of time elapsed during which the threads of the process executed code in privileged mode, expressed as a percentage value.
ProcessHandles	The number of handles held by the SQL Server process.
ProcessID	The ID of the SQL Server process.
ProcessMemoryUsed	The total amount of memory used by the SQL Server process.
ProcessorTime	The amount of time elapsed during which the threads of the process used the processor to execute instructions, expressed as a percentage value.
ProcessPageFaults	The number of page faults caused by the SQL Server process.
ProcessThreads	The number of threads used by the SQL Server process.
UserTime	The amount of time elapsed during which this process's threads executed code in user mode, expressed as a percentage value.

## Internal process table

The internal process (sqlsvrInternalProcTable) displays information about the processes running on the SQL Server. These processes can be client processes or system processes.

Table 284 describes the objects in sqlsvrInternalProcTable.

*Table 284. Internal process table (sqlsvrInternalProcTable)*

Row object	Description
Blocked	Process ID (SPID) of a blocking process.
Cmd	Command currently being executed.
CPU	Cumulative CPU time for the process (in milliseconds). The entry is updated for all processes.
DbID	ID of the database currently being used by the process.
Host	Name of the workstation.
IO	Cumulative disk reads and writes for the process.
Login	Name of the Windows username for the process (if using Windows Authentication) or a trusted connection.
LoginTime	Time at which a client process logged into the server. For system processes, the time at which the SQL Server started is reported.
Memory	Number of pages in the procedure cache that are currently allocated to this process. A negative number indicates that the process is freeing memory allocated by another process.
OpenTrans	Number of open transactions for the process.
Program	Name of the application program.
SPID	The ID of the SQL Server internal process.
Status	Process status.

Table 284. Internal process table (sqlsvrInternalProcTable) (continued)

Row object	Description
WaitTime	Current wait time in milliseconds. The value is 0 when the process is not waiting.

## Database statistics table

The database statistics table (sqlsvrDbStatsTable) describes the size of the databases available for a particular SQL Server.

Table 285 describes the objects in sqlsvrDbStatsTable.

Table 285. Database statistics table (sqlsvrDbStatsTable)

Row object	Description
PcntUsed	The percentage of allocated space for the particular database. It is calculated by the formula $(TotalSize - Unalloc) / TotalSize$ .
SpaceData	The total amount (KB) of space used by data.
SpaceIndex	The amount (KB) of space used by indexes.
SpaceRsvd	The total amount (KB) of reserved space.
SpaceUnused	The amount (KB) of unused space.
TotalSize	The size (KB) of the particular database.
Unalloc	The size (KB) of the unallocated space for the particular database.

## Database files table

The database files table (sqlsvrDbFilesTable) lists the database files used by the SQL Server.

Table 286 describes the objects in sqlsvrDbFilesTable.

Table 286. Database files table (sqlsvrDbFilesTable)

Row object	Description
CtrlType	The controller number of the file or device. Allowed values are:  0 - Database device  2 - Hard disk device  3 or 4 - Disk dump device  5 - Tape device
Desc	The description of the file or device.
FileIndex	An index into the database files for a SQL Server.
FileName	The file or device name.
FilePath	The physical location of the file or device.
Size	The size (KB) of the file or device.
Status	The status of the file or device. This status corresponds to the value of sqlsvrDbFilesDesc.



## Database performance table

The database performance table (sqlsvrDbPerfTable) displays statistics for all the database manager objects for a SQL Server.

Table 287 describes the objects in sqlsvrDbPerfTable.

*Table 287. Database performance table (sqlsvrDbPerfTable)*

Row object	Description
ActvTrans	The number of active transactions for the database.
BkpRestThrpt	The read/write throughput for backup/restore of a database.
BlkCpRows	The number of rows bulk copied.
BlkCpThroughpt	The amount of data (KB) that has been bulk copied.
DataFileSizes	The cumulative size of all the data files (KB) in the database.
DBCCLogicScan	The logical read scan rate for DBCC commands.
LogBytesFlushed	The total number of log bytes flushed.
LogCacheHitRatio	The percent of log cache reads that were satisfied from the log cache.
LogCacheReads	The reads performed through the log manager cache.
LogFileSize	The cumulative size of all the log files in the database.
LogFlushes	The number of log flushes.
LogFlushWaits	The number of commits waiting on log flush.
LogFlushWaitTime	The total wait time (ms).
LogGrowths	The total number of log growths for this database.
LogShrinks	The total number of log shrinks for this database.
LogTruncs	The total number of log truncations for this database.
LogUsedFileSize	The cumulative used size of all the log files in the database.
LogUtilized	The percent of space in the log that is in use.
ReplPendTrans	The number of pending replication transactions in the database.
ReplTransRate	The replication transaction rate (replicated transactions/sec.).
ShrinkData	The rate data is being moved by Autoshrink, DBCC SHRINKDATABASE or SHRINKFILE.
Transactions	The number of transactions started for the database.

## General statistics table

The general statistics table (sqlsvrGenStatsTable) displays general server statistics for the SQL Server.

Table 288 describes the objects in sqlsvrGenStatsTable.

*Table 288. General statistics table (sqlsvrGenStatsTable)*

Row object	Description
Logins	The total number of logins started.
Logouts	The total number of logouts started.
UserConns	The number of users connected to the system.

## SQL statistics table

The SQL statistics table (sqlsvrSQLStatsTable) displays the statistics associated with SQL requests for the database system.

Table 289 describes the objects in sqlsvrSQLStatsTable.

*Table 289. SQL statistics table (sqlsvrSQLStatsTable)*

Row object	Description
AttmpAutoParam	The number of auto-parameterization attempts.
BatchReq	The number of SQL batch requests received by the server.
FailedAutoParam	Number of failed auto-parameterizations.
SafeAutoParam	Number of safe auto-parameterizations.
SQLComp	The number of SQL compilations.
SQLReComp	The number of SQL re-compiles.
UnsafeAutoParam	Number of unsafe auto-parameterizations.

## Locks table

The locks table (sqlsvrLocksTable) describes statistics for individual server lock requests.

Table 290 describes the objects in sqlsvrLocksTable.

*Table 290. Locks table (sqlsvrLocksTable)*

Row object	Description
AvgWaitTime	The average amount of wait time (milliseconds) for each lock request that resulted in a wait.
Deadlocks	Number of lock requests that resulted in a deadlock.
LockIndex	Index into the locks table.
LockName	Name of the Lock instance.
LockReq	Number of new locks and lock conversions requested from the lock manager.
LockTimeouts	Number of lock requests that timed out. This includes internal requests for NOWAIT locks.
LockWaits	Number of lock requests that could not be satisfied immediately and required the caller to wait before being granted the lock.
LockWaitTime	Total wait time (milliseconds) for locks.

## Latches table

The latches table (sqlsvrLatchesTable) displays statistics related to the SQL Server latches.

Table 291 describes the objects in sqlsvrLatchesTable.

*Table 291. Latches Table (sqlsvrLatchesTable)*

Row object	Description
AvgWait	Average latch wait time (milliseconds) for latch requests that had to wait.
TotalWait	Total latch wait time (milliseconds) for latch requests that had to wait.

Table 291. Latches Table (sqlsrvLatchesTable) (continued)

Row object	Description
Waits	Number of latch requests that could not be granted immediately and had to wait before being granted.

## Access methods table

The access methods table (sqlsrvAccessMethodsTable) displays statistics associated with the database server access methods.

Table 292 describes the objects in sqlsrvAccessMethodsTable.

Table 292. Access methods table (sqlsrvAccessMethodsTable)

Row object	Description
ExtAlloc	The number of extents allocated to database objects for storing index or data records.
ExtDealloc	The number of extents deallocated to database objects.
FreeSpcPgFtchs	The number of pages returned by free space scans to satisfy requests to insert record fragments.
FreeSpcScans	The number of scans initiated to search for free space to insert a new record fragment.
FrwdRec	The number of records fetched through forwarded record pointers.
FullScans	The number of unrestricted full scans, either be base table or full index scans.
IdxSearch	The number of index searches. Index searches are used to start range scans, single index record fetches, and to reposition within an index.
MxdAlloc	The number of pages allocated from mixed extents. Used for storing the first eight pages allocated to an index or table.
PageDealloc	The number of pages deallocated to database objects.
PagesAlloc	The number of pages allocated to database objects for storing index or data records.
PageSplits	The number of page splits occurring as the result of index pages overflowing.
ProbeScans	The number of probe scans. A probe scan is used to directly look up rows in an index or base table.
RangeScans	The number of qualified range scans through indexes.
ScanPtReval	The number of times the scan point had to be revalidated to continue the scan.
SkipGhostRec	The number of ghosted records skipped during scans.
TableLockEsc	The number of times locks on a table were escalated.
WkFilesCreated	The number of work files created.
WkTableCacheRatio	The percent of worktables created where the initial pages were available in the worktable cache.
WkTableCreated	The number of worktables created.

## Buffer manager table

The buffer manager table (sqlsvrBufferMgrTable) displays statistics related to the SQL Server buffer manager.

Table 293 describes the objects in sqlsvrBufferMgrTable.

*Table 293. Buffer manager table (sqlsvrBufferMgrTable)*

Row object	Description
BuffCacheHitRatio	The percentage of pages that were found in the buffer pool without having to incur a read from disk.
ChkptPages	The number of pages flushed by checkpoint or other operations that require all dirty pages to be flushed.
FreePages	The total number of pages on all free lists.
LazyWrite	Number of buffers written by buffer manager's lazy writer.
PageLkps	The number of requests to find a page in the buffer pool.
PageReads	The number of physical database reads issued.
PageWrites	The number of physical database writes issued.
RdaheadPages	The number of pages read in anticipation of use.
RsvdPages	The number of buffer pool reserved pages.
StlnPages	The number of pages used for miscellaneous server purposes (including procedure cache).
TotalPages	The number of pages in the buffer pool (includes database, free and stolen).

## Cache manager table

The cache manager table (sqlsvrCacheMgrTable) displays the cache manager statistics for the SQL Server.

Table 294 describes the objects in sqlsvrCacheMgrTable.

*Table 294. Cache manager table (sqlsvrCacheMgrTable)*

Row object	Description
CacheHitRatio	The ratio between cache hits and lookups
CacheIndex	Index into the cache manager table for a specific cache.
CacheName	The name of the cache.
NumObj	The number of cache objects in the cache.
NumPages	Number of 8k pages used by cache objects.
UseCounts	The number of times each type of cache object has been used.

## Memory manager table

The memory manager table (sqlsvrMemMgrTable) displays the memory usage of the database system.

Table 295 describes the objects in sqlsvrMemMgrTable.

*Table 295. Memory manager table (sqlsvrMemMgrTable)*

Row object	Description
ConnMem	The total amount of dynamic memory the server is using for maintaining connections.
GrantWkSpcMem	The total amount of memory granted to executing processes. This memory is used for hash, sort and create index operations.
LockBlks	The current number of lock blocks that are in use on the server. Refreshed periodically.
LockBlksAlloc	The current number of allocated lock blocks.
LockMem	The total amount of dynamic memory the server is using for locks.
LockOwnrBlks	The current number of lock owner blocks that are in use on the server. Refreshed periodically.
LockOwnrBlksAlloc	The current number of allocated lock owner blocks.
MaxWkSpcMem	The total amount of memory granted to executing processes. This memory is used primarily for hash, sort and create index operations.
MemGrantsOut	The current number of processes that have successfully acquired a workspace memory grant.
MemGrantsPend	The current number of processes waiting for a workspace memory grant.
OptMem	The total amount of dynamic memory the server is using for query optimization.
SQLCacheMem	The total amount of dynamic memory the server is using for the dynamic SQL cache.
TargetSrvMem	The total amount of dynamic memory the server is willing to consume.
TotalSrvMem	The total amount of dynamic memory the server is currently consuming.



---

## Chapter 35. Netcool/ASM for Apache

The Netcool/ASM for Apache provides monitoring for aspects of the overall Apache server performance, including details about specific Web sites hosted by the server.

Netcool/ASM for Apache consists of the apache subagent and the apache MIB module. It monitors the performance of Apache Web servers. The Netcool/ASM for Apache adapts to HTTP site configuration changes and discovers newly added sites or removes sites that are no longer hosted.

---

### Overview

Netcool/ASM for Apache is comprised of a set of files.

Table 296 lists the subagent and MIB module component files.

*Table 296. Netcool/ASM for Apache component files*

File	Location	Description
apache.dll (Windows)	bin	Binary implementation.
libapache.so/.sl (UNIX)		
apache-mib.mib	mibs	MIB definition document.
apache.oid	config/oid	Object identifier file.
apache.cfg	config	Sample configuration file.

---

### Guidelines

The apache subagent automatically detects any Apache Web server installed on the host machine and updates the apacheSiteTable with statistics about any Web sites hosted by the Apache installation.

To load the subagent, use the following configuration command:

```
subagent load apache
```

**Tip:** If the apacheRoot MIB object is empty, it indicates that the subagent has not detected the Apache installation. To correct this, set the ApacheRoot inivar to the location of the Apache Web server configuration file.

### Enabling the Apache status module

For the subagent to obtain performance statistics from Apache Web servers, the Apache Status module mod\_status must be enabled and configured to provide extended status information.

#### About this task

For Apache 2.2.x servers, you need only ensure that `http://hostname:port/server-status` works. For all other Apache servers, complete the following steps.

## Procedure

To enable and configure mod\_status:

1. Add the following configuration settings to the Apache Web server configuration file:  

```
#-----  
ExtendedStatus On  
<Location /server-status>  
  SetHandler server-status  
  Order deny,allow  
  Deny from all  
  Allow from 127.0.0.1  
</Location>  
#-----
```
2. Ensure that the file contains the following entries:  

```
LoadModule status_module modules/mod_status.so  
AddModule mod_status.c
```
3. If the Apache configuration file is not in the default location, set the ApacheRoot inivar to the location of this file.  
For example, set ApacheRoot to /etc/httpd/2.0/conf/httpd2.conf.
4. Restart the Apache Web server.  
The apache subagent automatically detects all Apache Web server installations and collects performance statistics for them.

## Monitoring Apache Web servers in HTTPS mode

The subagent uses HTTP to access Apache server status pages. If you wish to monitor an Apache Web server running in HTTPS mode, you must create a virtual host running in HTTP mode on the Apache server, bind it to a specific port, then configure the subagent to use that port.

### About this task

For example, to enable the subagent to access an Apache the server's status page on port 1234:

## Procedure

1. Create a virtual host by adding the following configuration to the Apache Web server configuration file:  

```
Listen 1234  
<VirtualHost _default_:1234>  
  <Location /server-status>  
    SetHandler server-status  
    Order deny,allow  
    Deny from all  
    Allow from 127.0.0.1  
  </Location>  
</VirtualHost>
```
- Tip:** You may also wish to prevent the virtual host from serving content over HTTP by setting its DocumentRoot to a blank page.
2. Restart the Apache server.
3. Set the subagent inivar ApacheStatusPort to 1234:  

```
set inivar ApacheStatusPort=1234
```



## Inivars

The apache subagent provides the inivars for configuring its operation.

The inivars are listed in Table 297.

Table 297. *apache subagent inivars*

Inivar	Type	Description
ApacheRoot	string	Specifies the root directory of the Apache installation being monitored, for example /usr/local/apache. The subagent uses this path to locate the Apache configuration file. If the configuration file is not in the default location, set the value of the inivar to its full path and filename.
ApacheStatusPort	integer	Specifies the port used by the subagent to obtain the server status page from the Apache server.

For general information about inivars, see the *Netcool/SSM Administration Guide*.

---

## MIB module

The apache MIB is a subtree of networkharmoni(1977).

The subtree is shown in Figure 64.

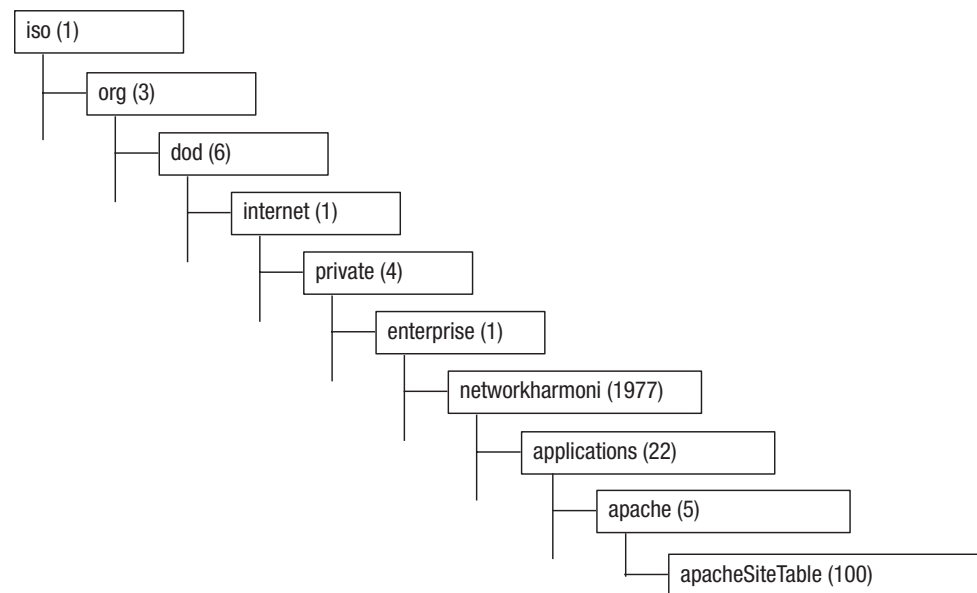


Figure 64. *OID tree diagram of the apache MIB module*

This section provides a summary of the objects defined in the apache MIB module. For detailed information on all objects in the module, see the `apache-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB objects

The apache MIB provides data in the form of scalar objects and one table. It contains no control objects.

### Global objects

A number of top-level scalar objects store global data from the Apache Web Server. This data is common to all HTTP sites monitored by the subagent.

Table 298 describes these scalars.

*Table 298. Top-level scalars for Apache MIB*

Object	Description
apacheAccessCount	The total number of accesses made to the Web server (including all virtual hosts) since httpd started (that is, in the period indicated by apacheUptime).
apacheCPU	Total CPU time consumed by the main apache server process (apachePid) and all of its immediate child processes.
apacheCurConn	Current number of connections to the server.
apacheMemResident	The amount of apacheMemSize (in KB) that is currently paged into physical memory.
apacheMemSize	Total memory usage (in KB) of the apache server process and its immediate children.
apachePid	The process ID of the active httpd server. If no server is running this value will be zero.
apacheRoot	The root path of the Apache installation, to which relative paths are appended.
apacheScriptLog	Full path of the CGI script error log file.
apacheServers	The number of server sub-processes that are waiting for DNS lookups to complete.
apacheServersFinishing	The number of server sub-processes that are finishing up in order to return to an idle state.
apacheServersIdle	The current number of server sub-processes that are idle (waiting for requests).
apacheServersKeepAlive	The number of server sub-processes that are occupied keeping their connections open in preparation for further requests.
apacheServersLogging	The number of server sub-processes that are busy writing to log files.
apacheServersReading	The number of server sub-processes that are busy reading client requests.
apacheServersReplying	The number of server sub-processes that are busy sending replies to clients.
apacheServersStarting	The number of server sub-processes that are currently starting up.
apacheServersTotal	The current number of server sub-processes available for servicing HTTP requests (one per sub-process).
apacheTraffic	The total amount of traffic into and out of the Web server (including all virtual hosts) in KB since httpd started.
apacheUptime	Uptime of the uptime of the current server instance within the httpd process identified by apachePid.

Table 298. Top-level scalars for Apache MIB (continued)

Object	Description
apacheVersion	The complete server ID string, which usually starts with Apache/x.y.z. This string will be empty when no active server is detected.

## Site table

The Site Table (apacheSiteTable) represents a single Web site hosted by the specified host. The table is indexed by port (apacheSitePort) and address (apacheSiteAddr).

Site Table objects are listed in Table 299. The values stored in apacheSiteTable are specific to each site.

Table 299. Site table (apacheSiteTable)

Object	Description
AccessLog	The full file system path of the access log file for this site. Log files may be shared between multiple Web sites on the same host. This string may also be a pipe command (starting with ' ').
Addr	The name (usually fully qualified) or IP address by which the Web site is known. The site address is included in the index in order to distinguish between named virtual hosts.
Admin	The email address of the person responsible for administration of the Web site.
DocumentRoot	The full file system path of where documents for this Web site are stored.
ErrorLog	The full file system path of the error log file for this site. Log files may be shared between multiple Web sites on the same host. This string may also be a pipe command (starting with ' '), in which case error counters will not be updated.
ForbiddenErrors	The number of HTTP 403 (forbidden) errors that have been logged in apacheSiteErrorLog since the agent started.
NotFoundErrors	The number of HTTP 404 (not found) errors that have been logged in apacheSiteErrorLog since the agent started.
Port	The port on which the site is hosted. This forms the first part of the table index.
TotalErrors	The total number of errors logged in apacheSiteErrorLog since the agent started.



---

## Chapter 36. Netcool/ASM for WebSphere

The Netcool/ASM for IBM WebSphere contains a wide range of performance metrics for IBM WebSphere servers, derived from the WebSphere Performance Monitoring Infrastructure (PMI). It provides a flexible monitoring scheme, enabling you to monitor particular counters in a collection.

Netcool/ASM for WebSphere consists of the websphere subagent and the webSphere MIB module. It provides facilities for gathering performance data from IBM WebSphere servers.

The subagent utilizes a servlet that loads on the target WebSphere server. The servlet gathers performance data by interrogating the WebSphere Performance Monitoring Infrastructure (PMI) and transfers this data to the subagent in XML format via HTTP.

---

### Component files

Netcool/ASM for WebSphere is comprised of a set of component files.

Table 300 lists the component files of Netcool/ASM for WebSphere.

*Table 300. Netcool/ASM for WebSphere component files*

File	Location	Description
websphere.dll (Windows) libwebsphere.so/.sl (UNIX)	bin	Binary implementation.
websphere-mib.mib	mibs	MIB definition document.
websphere.oid	config/oid	Object identifier file.
websphere.cfg	config	Sample configuration file.
PerfServlet.ear	bin	WebSphere performance monitoring servlet.

---

### Guidelines

To use the websphere subagent for monitoring performance data on a WebSphere Application Server, install the server, set up the performance monitoring servlet on the target Websphere server, configure the subagent, then create control rows identifying the performance metrics that you wish to monitor.

## Integrating with WebSphere Application Server V7.0 and later

Netcool/ASM for WebSphere uses the performance servlet included with WebSphere Application Server V7.0 and later.

### Procedure

To set up the performance servlet, complete the following steps:

1. In the Integrated Solutions Console, select **Monitoring and Tuning > Performance Monitoring Infrastructure (PMI)** and select your server.
2. Ensure that **Enable Performance Monitoring Infrastructure (PMI)** is selected.
3. Set **Currently monitored statistic set** to **Extended**.
4. Restart WebSphere Application Server for the changes to take effect.
5. Use the Integrated Solutions Console or whatever means you currently use to deploy applications to deploy the servlet: `WAS_HOME/installableApps/PerfServletApp.ear`
6. Start the servlet and verify that it is running from the agent host. With a web browser, go to the following address: `http://hostname/wasPerfTool/servlet/perfservlet` If an HTTP server is not installed, the servlet is available on port 9080 by default: `http://hostname:9080/wasPerfTool/servlet/perfservlet`

### What to do next

After the servlet is running, the websphere subagent automatically finds it and lists the collection names in `wsCollectionTable`. Ensure that Netcool/ASM for WebSphere is configured for the same host name and port number as used in step 6. A sample configuration script for WebSphere Application Server V7.0 is provided in `config/websphere7.cfg`. For further information about PerfServlet, see WebSphere Application Server V7.0 Information Center (<http://pic.dhe.ibm.com/infocenter/wasinfo/v7r0/index.jsp>).

## Configuring the subagent

The subagent provides inivars for configuring its operation.

### About this task

The websphere subagent provides an inivar named `WebsphereRefreshInterval` that defines the interval at which the servlet refreshes the WebSphere node and server objects in its cache. This refresh process eliminates problems associated with memory leaks in the PMI Client API.

To set this variable, add the following entry to the agent configuration file `init.cfg`: `WebsphereRefreshInterval=x`, where `x` represents the number of requests that must be received from the agent before the servlet cache is refreshed. If `init.cfg` does not contain this entry, the websphere subagent assumes a default value of 0, which means that the servlet never refreshes its cache.

The subagent also provides the inivar `WebsphereRMIPort`, which sets the RMI port (the WebSphere `BOOTSTRAP_ADDRESS` port) that the subagent uses when obtaining metrics from the target WebSphere server. By default, the subagent uses port 2809.

## Monitoring performance data

Use the subagent to collect performance data from the Websphere server for threshold monitoring and reporting.

### About this task

After installing the performance monitoring servlet on the target Websphere server, monitor performance data by loading the subagent then creating control rows to gather the performance data that you wish to monitor.

### Procedure

1. Load the websphere subagent using the command:  
`subagent load websphere`
2. Configure the `wsServer` and `wsPort` objects with the IP address and port of the WebSphere server.  
The subagent adds rows to `wsCollectionTable`, one row for each collection found on the server.
3. Examine `wsCollectionTable` for the list of collections found on the WebSphere server and identify the collections that you wish to monitor.
4. Create a control row containing a regular expression identifying the collections that you wish to monitor.  
The websphere subagent uses substring matching in regular expression evaluation. For more details on regular expressions, see Appendix D, "Regular expressions," on page 615.
5. If you set the control row object `wsControlCounterMode` to `manual(1)`, create a row in the counter table `wsCounterTable` for each counter that you wish to monitor and set the `wsCounterName` object to the name of the counter. If you set the control row object `wsControlCounterMode` to `automatic(2)`, all counters will be monitored.  
Counter data from the monitored counters is then logged in `wsDataTable`.

### Results

To determine the counters available in a collection:

1. Set the `wsControlCounterMode` object of a `wsControlTable` control row that references the collection to `automatic(2)`.  
This creates a row in `wsCounterTable` for each counter available in the collection.
2. If you then wish to remove any of these rows:
  - Set `wsControlCounterMode` to `manual(1)`.
  - Set the row's `wsCounterName` object to an empty (zero length) string.

**Tip:** Load data values (`RangeStatistic`, `BoundedRangeStatistic`, `CpdLoad`) are multiplied by 100.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### Control table

The websphere commands create rows in the webSphere control table (wsControlTable).

The general syntax of these commands is:

```
websphere property=value  
websphere create [property=value ...]  
websphere reset
```

Table 301 lists the properties supported in these commands.

*Table 301. Configuration command parameters - websphere*

Property	Type	Description	Sets MIB object
collection	string	A regular expression specifying the expressions to be monitored.	Collection
countermode	enum	The counter row creation mode for the counter table:  automatic  manual	CounterMode

### Counter table

The webspherectr commands create rows in the websphere counter table (wscountertable) and associate them with the next row created in the configuration table using the websphere create command.

The general syntax of these commands is:

```
webspherectr property=value  
webspherectr store [property=value ...]  
webspherectr reset
```

Table 302 lists the properties supported in these commands.

*Table 302. Configuration Command Parameters - webspherectr*

Property	Type	Description	Sets MIB object
name	string	The name of the counter to be monitored.	Name



## Scalars

The `websphereglobal` command sets the value of websphere scalar objects.

The syntax of this command is:

`websphereglobal property=value`

Table 303 lists the properties supported in these commands.

*Table 303. Configuration command parameters - websphereglobal*

Property	Type	Description	Sets MIB object
password	string	Specifies the password supplied by the subagent when logging on to the monitored WebSphere server.	wsPassword
port	int	The port number of the WebSphere server to be monitored.	wsPort
sampleinterval	int	The interval (in seconds) at which the data is collected from the WebSphere server.	wsSampleInterval
server	string	The IP address or DNS name of the WebSphere server to be monitored.	wsServer
username	string	Specifies the username supplied by the subagent when logging on to the monitored WebSphere server.	wsUsername

---

## Examples

These examples demonstrate how to use the websphere subagent configuration commands to set up simple monitoring tasks for a WebSphere server.

### Monitoring all counters

Monitor at 5-minute intervals all available counters in all collections on a WebSphere server with IP address 127.0.0.1 and port 80:

```
subagent load websphere
websphereglobal server=127.0.0.1 port=80 sampleinterval=300
websphere reset
websphere collection=.*
websphere countermode=automatic
websphere create
```

### Monitoring Web app module errors

Monitor at 1-minute intervals the counters `webAppModule.numErrors` and `webAppModule.totalRequests` in the collection `root/ws/server1/webAppModule` on a WebSphere server `ws` running on port 80:

```
subagent load websphere
websphereglobal server=ws port=80 sampleinterval=60
websphere reset
websphere collection=root,ws,server1,webAppModule
websphere countermode>manual
webspherectr store name=webAppModule.numErrors
webspherectr store name=webAppModule.totalRequests
websphere create
```

## Monitoring WebSphere server CPU usage

Monitor the CPU usage of the WebSphere server named ws at 1-minute intervals and generate an alarm if it exceeds 80%. CPU usage information is obtained from the counter systemModule.cpuUtilization in the collection systemModule.

```
subagent load rmonc
event reset
event type=snmp-trap
event description="WebSphere CPU utilization alarm"
event create
wsevent=$?

subagent load websphere
websphereglobal server=ws port=80 sampleinterval=60
websphere reset
websphere collection="systemModule"
websphere countermode=manual
webspherectr store name="systemModule.cpuUtilization"
websphere create
wssmcontrolindex=$?

subagent load genalarm
genalarm reset
genalarm interval=60
genalarm type=absolute
genalarm mode=singleEdge
genalarm var=$wsDataValue.$wssmcontrolindex.1.?
genalarm risethresh=80
genalarm riseverity=warning
genalarm riseevent=$wsevent
genalarm vardescr="WebSphere: CPU utilization"
genalarm risedescr="The Websphere CPU utilization is at $$6%"
genalarm create
```

---

## MIB module

The webSphere MIB is a subtree of networkharmoni (1977).

The subtree is shown in Figure 65 on page 409.

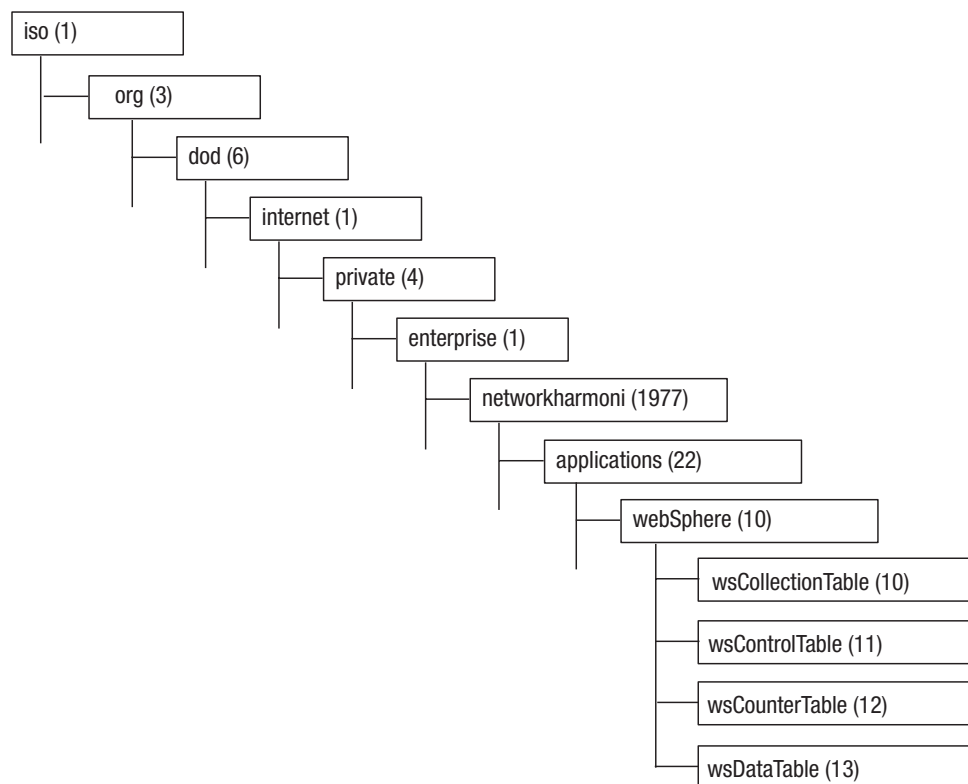


Figure 65. OID tree diagram of the webSphere MIB module

This section provides a summary of the objects defined in the webSphere MIB module. For detailed information on all objects in the module, see the websphere-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## Scalar objects

The webSphere MIB module provides a set of scalar objects for identifying the WebSphere server to be monitored.

Table 304 lists these objects.

Table 304. webSphere scalar objects

Scalar Object	Description
wsPassword	Specifies the password supplied by the subagent when logging on to the monitored WebSphere server.
wsPort	Specifies the port number of the WebSphere server to be monitored.
wsSampleInterval	Sets the interval (in seconds) at which the data is collected from the WebSphere server.
wsServer	Specifies the IP address or DNS name of the WebSphere server to be monitored.
wsUsername	Specifies the username supplied by the subagent when logging on to the monitored WebSphere server.

## Collection table

The webSphere collection table (`wsCollectionTable`) lists the collections available for use in gathering data from the WebSphere server. Control rows in `wsControlTable` reference rows in this table via the collection name.

Table 305 describes the objects defined in `wsCollectionTable`.

*Table 305. Collection table (`wsCollectionTable`)*

Row object	Description
Index	Uniquely identifies this row.
Name	The full name of the collection. This name is referenced by <code>wsControlCollection</code> .

## Control table

The webSphere control table (`wsControlTable`) provides control rows for defining the data to be gathered from the server.

Table 306 describes the objects defined in `wsControlTable`.

*Table 306. Control table (`wsControlTable`)*

Row object	Description
Collection	A regular expression specifying the collections to be monitored. This expression is applied to the <code>wsCollectionName</code> objects in <code>wsCollectionTable</code> when determining the collections to be monitored. This expression need only match a substring in a collection name. For information about using regular expressions, see Appendix D, "Regular expressions," on page 615.
CounterMode	Controls how counter names are created in <code>wsCounterTable</code> :  manual(1) - Counter rows for the required counters must be created manually in <code>wsCounterTable</code> .  automatic(2) - Counter rows for all available counters are automatically created in <code>wsCounterTable</code> . These rows cannot be changed without setting CounterMode to manual(1).
Index	Uniquely identifies this row.
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Status	SNMPv2 Row Status. Controls creation, activation and deletion of the control row.

## Counter table

The webSphere counter table (`wsCounterTable`) stores the names of counters to be monitored. The contents of this table (the counters available for monitoring) are determined by control rows in `wsControlTable`.

Table 307 describes the objects defined in `wsCounterTable`.

*Table 307. Counter table (`wsCounterTable`)*

Row object	Description
Index	Uniquely identifies a row in this table, relative to <code>wsControlIndex</code> .

Table 307. Counter table (wsCounterTable) (continued)

Row object	Description
Name	The name of the counter to be monitored. If wsControlCounterMode is set to manual (1), counter names must be entered manually. Entering an empty (zero-length) string removes a row. If wsControlCounterMode is set to automatic (2), rows in this table are created for every available counter and the Name object of each row is automatically set to the name of the counter.

## Data table

The webSphere data table (wsDataTable) stores data gathered from the monitored counters. Each row contains the data from one counter and is indexed by a combination of the objects wsCollectionIndex, wsControlIndex and wsCounterIndex.

The name of each counter can be determined from the wsCounterName object in the corresponding row of wsCounterTable. Table 308 describes the objects defined in wsDataTable.

Table 308. Data table (wsDataTable)

Row object	Description
Type	Indicates the data type of the counter:  long (1)  load (2)  stat (3)
Value	The data value of the counter.



---

## Chapter 37. Netcool/ASM for WebLogic

The Netcool/ASM for WebLogic provides performance metrics for BEA WebLogic servers. The performance metrics are available as raw bean attribute data, however you can derive composite metrics from this data using a range of built-in functions.

Netcool/ASM for WebLogic consists of the `weblogic` subagent and the `weblogic` MIB module. It provides facilities for gathering performance data from BEA WebLogic application servers.

The subagent utilizes a servlet that loads on the target WebLogic server. The servlet gathers performance data and transfers this data to the subagent in XML format over HTTP.

---

### Component files

Netcool/ASM for WebLogic is comprised of a set of component files.

Table 309 lists the component files of Netcool/ASM for WebLogic.

*Table 309. Netcool/ASM for WebLogic component files*

File	Location	Description
<code>weblogic.dll</code> (Windows)	<code>bin</code>	Binary implementation.
<code>libweblogic.so/.sl</code> (UNIX)		
<code>weblogic-mib.mib</code>	<code>mibs</code>	MIB definition document.
<code>weblogic.oid</code>	<code>config/oid</code>	Object identifier file.
<code>weblogic.cfg</code>	<code>config</code>	Sample configuration file.
<code>wl_setup.cmd</code> (Windows)	<code>bin</code>	Performance monitoring servlet installation files.
<code>wl_setup.sh</code> (UNIX)		
<code>PerfMonitor.ear</code>	<code>bin</code>	WebLogic performance monitoring servlet.

---

### Guidelines

To WebLogic servers, configure the subagent, then create control rows to collect the performance metrics that you wish to monitor.

#### Subagent configuration

Configuring the `weblogic` subagent requires you to install a servlet on the WebLogic servers that you wish to monitor.

The `weblogic` subagent utilizes the servlet `PerfMonitor.ear` to access the performance data on WebLogic servers.

Netcool/ASM for WebLogic provides a script for installing the servlet. Both the installation script and the servlet are located in the `bin` sub-directory of your Netcool/SSM installation. To run the installation script, execute the following command from the `bin` sub-directory of your Netcool/SSM installation:

On Windows systems:

```
wl_setup.cmd weblogic_home username password server ,server, port
```

On UNIX systems:

```
wl_setup.sh weblogic_home username password server ,server, port
```

Table 310 describes the parameters required by `wl_setup`.

*Table 310. wl\_setup parameters*

Parameter	Description
password	The password associated with the admin username.
port	The port used by the WebLogic server. The standard port is 7001.
server	The name of the WebLogic server that you wish to monitor. The PerfMonitor.ear servlet will be installed on this server. To specify more than one server, separate each with a comma.
username	The admin username of the WebLogic server to be monitored.
weblogic_home	The home directory of the WebLogic server that you wish to monitor. For example, <code>c:\bea\weblogic700</code> .

The installation script writes the results of the servlet installation process to the log file `weblogic_install.log` located in the `log` directory of your Netcool/SSM installation.

## Inivars

The `weblogic` subagent provides the `WeblogicURLPort` inivar, which sets the port on which the subagent queries the target WebLogic server.

Normally it is not necessary to set this inivar, however if the subagent is unable to obtain performance data from the server, and the agent log file (`agent.log`) contains errors similar to the following:

```
could not connect over HTTP to server: 'server', port: 'nn'
```

setting the `WeblogicURLPort` inivar to the target port may resolve this issue.

## Monitoring performance data

To monitor Weblogic performance data, load the subagent and create control rows to extract the required performance data.

### About this task

After installing the performance monitoring servlet on the target WebLogic server, monitor performance data by loading the subagent then creating control rows to gather the performance data that you wish to monitor:

### Procedure

1. Load the subagent using the command:  

```
subagent load weblogic
```
2. Configure the `wlServer` and `wlPort` objects with the IP address (or DNS name) and port of the WebLogic server you wish to monitor, and set the `wlUsername` and `wlPassword` objects to the values used by the WebLogic admin user.



The subagent populates the `wlBeanAttrTable`, `wlBeanInstTable` and `wlBeanInstTable` tables with information about the beans identified on the WebLogic server.

3. Examine the tables `wlBeanAttrTable`, `wlBeanInstTable` and `wlBeanInstTable` and identify the type, instance and attribute of the beans that you wish to monitor on the WebLogic server.
4. Create a control row that specifies the type of bean (`wlControlBeanType`), and instances of that type (`wlControlBeanInst`), that you wish to monitor. Select the attributes that represent the performance data you wish to gather (`wlControlBeanAttr1` and, if required, `wlControlBeanAttr2`) and specify a formula for calculating the performance metric (`wlControlFormula`).
5. Activate the control row.

Performance data for the selected bean instances is then logged in `wlDataTable`.

**Note:** The weblogic subagent uses substring matching in regular expression that identifies the bean instances. For more details on regular expressions, see Appendix D, “Regular expressions,” on page 615.

## SNMPv1 compatibility

The weblogic MIB module uses the Counter64 data type, so it is not compatible with SNMPv1. Use Netcool/ASM for WebLogic only with SNMPv2 and later.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Control table

The weblogic commands create rows in the weblogic control table (`wlControlTable`).

The general syntax of these commands is:

```
weblogic property=value
weblogic create [property=value ...]
weblogic reset
```

Table 311 lists the properties supported in these commands.

*Table 311. Configuration command parameters - weblogic*

Property	Type	Description	Sets MIB object
beanattr1	string	The name of the primary bean attribute to be monitored.	BeanAttr1
beanattr2	string	The name of the second bean attribute.	BeanAttr2
beaninst	string	A regular expression specifying the name of one or more bean instances to be monitored.	BeanInst
beantype	string	The type name of the bean to be monitored.	BeanType

Table 311. Configuration command parameters - weblogic (continued)

Property	Type	Description	Sets MIB object
formula	enum	The formula used to derive the performance data from the raw bean attribute values.  delta  deltaratio  percentage  rate  raw	Formula

## Scalars

The weblogicglobal command sets the value of weblogic scalar objects.

The general syntax of this command is:

weblogicglobal *property=value*

Table 312 lists the properties supported in this command.

Table 312. Configuration command parameters - weblogicglobal

Property	Type	Description	Sets MIB object
password	string	The password for the administrative user.	wlPassword
port	int	The port number of the WebLogic server to be monitored.	wlPort
sampleinterval	int	The interval (in seconds) at which the data is collected from the WebLogic server.	wlSampleInterval
server	string	The IP address or DNS name of the WebLogic server to be monitored.	wlServer
username	string	The administrative username for the WebLogic server.	wlUsername

## Examples

These examples demonstrate how to use the weblogic subagent configuration commands to set up simple monitors for a WebLogic server.

### Monitoring a WebLogic server

Configure the weblogic subagent to monitor a WebLogic server named myWLserver on port 7000 at 60-second intervals. The admin username for the server is myWLuser and the password is myWLpass:

```
subagent load weblogic
weblogicglobal server=myWLserver
weblogicglobal port=7000
weblogicglobal sampleinterval=60
weblogicglobal username=myWLuser
weblogicglobal password=myWLpass
```

## Monitoring the number of sessions

Poll the total number of sessions open on a WebLogic server:

```
weblogic reset
weblogic beantype=WebAppComponentRuntime
weblogic beanattr1=OpenSessionsCurrentCount
weblogic formula=raw
weblogic create
```

## Monitoring JVM heap usage

Calculate the current amount of free memory in the JVM heap expressed as a percentage of the current size of the JVM heap. If the usage exceeds 95%, generate an alarm:

```
subagent load rmonc
event reset
event type=snmp-trap
event description="WebLogic alarm"
event create
wlevent = $?

subagent load weblogic
subagent load genalarm
snmp get $wlsampleinterval.0
wlsampleinterval=$?
weblogic reset
weblogic beantype=JVMRuntime
weblogic beanattr1=HeapFreeCurrent
weblogic formula=percentage
weblogic beanattr2=HeapSizeCurrent
weblogic create
controlindex=$?

genalarm reset
genalarm interval=$wlsampleinterval
genalarm type=absolute
genalarm mode=hysteresis
genalarm var=$wldatavalue.$controlindex.*
genalarm startup=falling
genalarm risethresh=5
genalarm fallthresh=5
genalarm fallseverity=warning
genalarm falldescr="Only $$6% of the JVM heap is available"
genalarm fallevent=$wlevent
genalarm vardescr="WebLogic: JVM heap size utilization"
genalarm create
```

---

## MIB module

The weblogic MIB is a subtree of networkharmoni(1977).

The subtree is shown in Figure 66 on page 418.

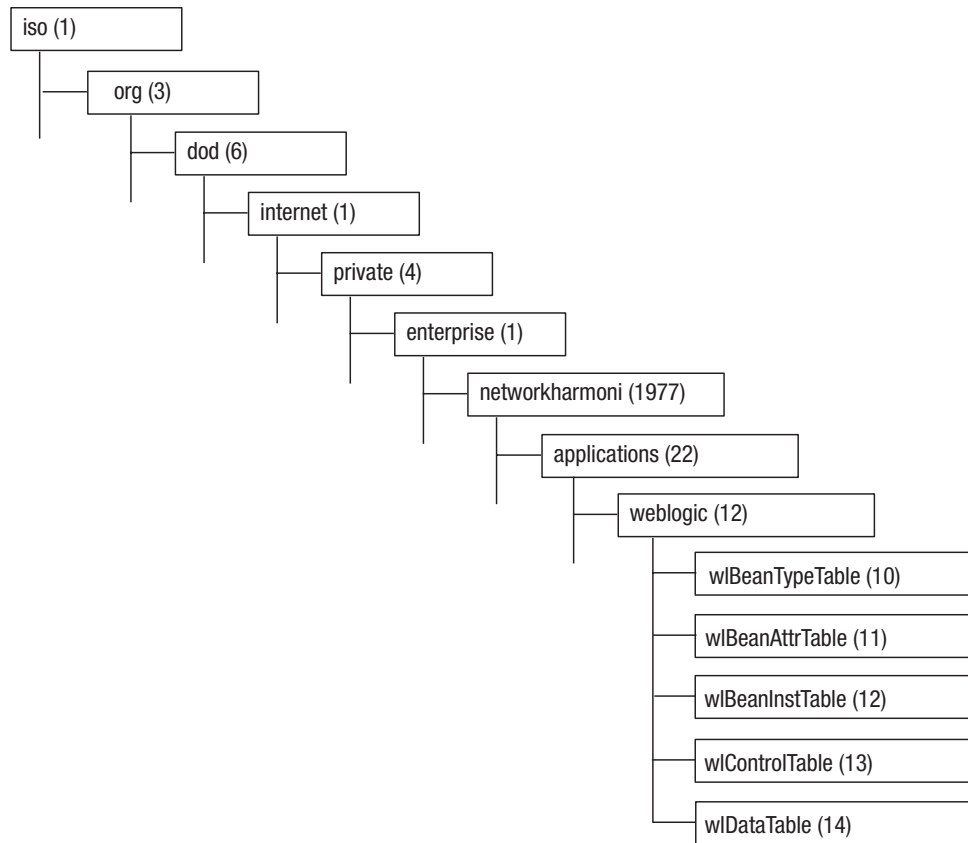


Figure 66. OID Tree Diagram of the weblogic MIB Module

This section provides a summary of the objects defined in the weblogic MIB module. For detailed information on all objects in the module, see the `weblogic-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM installation.

## MIB objects

The weblogic MIB provides a standard SNMP interface.

The module defines the following objects:

- `wlServer`  
Specifies the IP address or DNS name of the WebLogic server to be monitored.
- `wlPort`  
Specifies the port number of the WebLogic server to be monitored.
- `wlSampleInterval`  
Sets the interval (in seconds) at which the data is collected from the WebLogic server.
- `wlUsername`  
The administrative username for the WebLogic server.
- `wlPassword`  
The corresponding password for the administrative user.

In addition to these objects, the weblogic MIB provides the following tables for configuring data collection from WebLogic servers.

### Bean type table

The bean type table (wlBeanTypeTable) lists the bean types available for use in gathering data from the WebLogic server.

Table 313 describes the objects defined in wlBeanTypeTable.

*Table 313. Bean type table (wlBeanTypeTable)*

Row object	Description
Index	Uniquely identifies this row in wlBeanTypeTable.
Name	The class name of the bean type that this row represents.

### Bean attribute table

The bean attribute table (wlBeanAttrTable) lists all available bean attributes. It is provided for reference purposes: use it to locate the name of the bean attribute that you wish to monitor.

Table 314 describes the objects defined in wlBeanAttrTable.

*Table 314. Bean attribute table (wlBeanAttrTable)*

Row object	Description
Index	Uniquely identifies this row in wlBeanAttrTable.
Name	The name of the attribute that this row represents.

### Bean instance table

The bean instance table (wlBeanInstTable) lists all available bean instances on a WebLogic server. It is provided for reference purposes: use it to identify the names of the bean instances that you wish to monitor.

Table 315 describes the objects defined in wlBeanInstTable.

*Table 315. Bean instance table (wlBeanInstTable)*

Row object	Description
Index	Uniquely identifies this row in wlBeanInstTable.
Name	The instance name of the bean represented by this row.

### Control table

The control table (wlControlTable) provides control rows for defining the performance data to be monitored on a WebLogic server. Each control row selects the type of bean and one or more instances of that type to be monitored, and specifies the formula to be used in calculating the performance data.

Table 316 describes the objects defined in wlControlTable.

*Table 316. Control table (wlControlTable)*

Row object	Description
BeanAttr1	The name of the primary bean attribute to be monitored. This must be the name of one of the bean types listed in wlBeanAttrTable.

Table 316. Control table (wlControlTable) (continued)

Row object	Description
BeanAttr2	This object is only used when the formula indicated by wlControlFormula requires a second bean attribute. This object holds the name of the second bean attribute. This must be the name of one of the bean types listed in wlBeanAttrTable.
BeanInst	A regular expression specifying the name of one or more bean instances to be monitored. This expression need only match a substring in a bean instance name.
BeanType	The type name of the bean to be monitored. This must be the name of one of the bean types listed in wlBeanTypeTable.
Formula	Specifies the formula used to derive the performance data gathered by this control row from the raw bean attribute values. The following formulas are available:  raw(1) - The raw, unmodified value of wlControlBeanAttr1  delta(2) - The delta of raw value of wlControlBeanAttr1  percentage(3) - A percentage value calculated as $\text{BeanAttr1} \times 100 / \text{BeanAttr2}$  deltaRatio(4) - Calculated as $\text{delta}(\text{BeanAttr1}) / \text{delta}(\text{BeanAttr2})$ .  rate(5) - Calculated as $\text{delta}(\text{BeanAttr1}) / \text{wlSampleInterval}$  Note: Performance data calculated using delta and deltaRatio is only available after the second sample because a minimum of 2 samples are required to perform these calculations.
Index	Uniquely identifies this row in wlControlTable.
LastUpdate	The time, relative to sysUpTime, when the last sample took place.
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Status	SNMPv2 row status. Controls creation, activation and deletion of the control row.

## Data table

The data table (wlDataTable) stores the performance data gathered by each control row defined in wlControlTable. Each row in the data table represents a performance metric for a particular bean instance.

Table 317 describes the objects defined in wlDataTable.

Table 317. Data table (wlDataTable)

Row object	Description
Value	The current value of the performance metric in 32-bit signed format. If the value has more than 31 significant bits then this object will contain the lower 32-bits of the true value.
Value64	The current value of the performance metric in 64-bit unsigned format.

---

## Chapter 38. Netcool/ASM for Active Directory

The Netcool/ASM for Microsoft Active Directory provides facilities for monitoring the status and performance of Active Directory domain controllers, the connectivity of Active Directory entities and the contents of the Active Directory itself.

The Netcool/ASM for Active Directory consists of the `activedir` subagent and the `activedir` MIB module. It provides facilities for monitoring the status and performance of Microsoft Active Directory domain controllers. In addition to performance counter objects, it provides tables for monitoring the connectivity of Active Directory entities and the contents of the Active Directory itself.

---

### Component files

Netcool/ASM for Active Directory is comprised of a set of component files.

Table 318 lists the component files of Netcool/ASM for Active Directory.

*Table 318. Netcool/ASM for Active Directory component files*

File	Location	Description
<code>activedir.dll</code>	<code>bin</code>	Binary implementation.
<code>activedir-mib.mib</code>	<code>mibs</code>	MIB definition document.
<code>activedir.oid</code>	<code>config/oid</code>	Object identifier file.
<code>activedir.cfg</code>	<code>config</code>	Sample configuration file.

---

### Guidelines

Use the subagent to monitor Active Directory connectivity and the contents of Active Directory objects.

#### About this task

To load the `activedir` subagent, open a command console and execute the command:

```
subagent load activedir
```

**Note:** By default, the subagent does not load on host machines that are not Active Directory domain controllers. To force the subagent to load on a machine that is not a domain controller, set the `ActivedirForceLoad` invar to `yes`.

## Monitoring Active Directory connectivity

Use the `adConnect` group to monitor the connectivity of Active Directory entities.

### Procedure

To monitor an the connectivity of an entity, create a row in `adConnectControlTable`:

1. Set `adConnectControlStatus` to `createAndWait(5)`.
2. Set `adConnectControlConnectTo` to the value appropriate to the type of entity whose connectivity you wish to monitor.
3. Set `adConnectControlConnectionString`, `adConnectControlAuthentication`, `adConnectControlUsername` and `adConnectControlPassword` according to the entity you wish to monitor.
4. Set `adConnectControlInterval` to the interval at which you wish to monitor the entity.
5. Set `adConnectControlNumberBuckets` to the number of data rows you wish to store in the data table.
6. Activate the control row by setting `adConnectControlStatus` to `active(1)`.

### Specifying entity connections

The `adConnectControlConnectTo` object defines the type of entity whose connectivity you wish to monitor. Table 319 lists the allowed values of this object and the entity type that each value corresponds to.

*Table 319. adConnectControlConnectTo object values*

Value	Connection Target
<code>domainMaster(5)</code>	The domain controller that holds the domain naming master role for this forest.
<code>domainMembers(10)</code>	All the domain controllers in the domain specified by the <code>adConnectControlConnectionString</code> object. If this object does not specify a domain, the domain that the machine hosting the <code>activedir</code> subagent belongs to is used.
<code>domainsInSite(9)</code>	If the domain controller is a member of a site, the <code>activedir</code> subagent attempts to connect to all the domains that the site is a member of.
<code>genericConnect(1)</code>	The entity whose LDAP binding string is specified by <code>adConnectControlConnectionString</code> .
<code>globalCatalog(7)</code>	The global catalog server for this domain.
<code>infraMaster(6)</code>	The domain controller that holds the infrastructure master role for this domain.
<code>pdc(2)</code>	The domain controller that holds the PDC emulator role.
<code>ridMaster(3)</code>	The domain controller that allocates RIDs to this domain controller.
<code>schemaMaster(4)</code>	The domain controller that holds the schema master role for this forest.
<code>trustedDomains(8)</code>	The domain controllers of any directly trusted domains.

If the `adConnectControlConnectTo` object is set to one of the FSMO roles `pdc`, `ridMaster`, `schemaMaster`, `domainMaster` or `infraMaster` and the local domain controller holds that role, the subagent will connect to the local domain controller.

To monitor entities when the `adConnectControlConnectTo` object is set to the value `trustedDomains`, `domainsInSite` or `domainMembers`, the subagent requires access to



the relevant domain's DNS records because it uses DNS queries to retrieve the domain controller names.

## **Monitoring specific entities**

### **About this task**

If you wish to manually identify a specific Active Directory entity to connect to, set `adConnectControlConnectTo` to the value `genericConnect` and enter an LDAP binding string for the target entity in `adConnectControlConnectionString` using the following format:

```
LDAP://HostName[:PortNumber][/DistinguishedName]
```

The following guidelines apply to LDAP binding strings:

### **Procedure**

- If the binding string does not commence with the LDAP protocol specifier `LDAP://`, the `activedir` subagent automatically prepends it to the string.
- `HostName` is not a mandatory part of the binding string; however, omitting it from the string is not recommended because there is no guarantee which host the `activedir` subagent connects to.

## **Monitoring specific domains**

### **About this task**

To monitor the members of a specific domain, set `adConnectControlConnectTo` to `domainMembers` and `adConnectControlConnectionString` to the name of the domain you wish to monitor; for example, `support.mysite.com`.

## **Authentication and encryption**

When the `activedir` subagent attempts to connect to the target entity, it authenticates itself using the credentials specified by the `adConnectControlUsername` and `adConnectControlPassword` objects. If these objects are empty, it uses the credentials of the Netcool/SSM installation.

When specifying a username in `adConnectControlUsername`, include the domain that the user belongs to, in the format `domain name\username`.

The `adConnectControlAuthentication` object determines whether data sent in the connection is encrypted. Encryption uses the Secure Sockets Layer (SSL) and requires that the appropriate security infrastructure, such as SSL certificates, be present on both the machine running the Netcool/SSM installation and the target entity.

On Windows platforms, the `activedir` subagent establishes connections using the NTLM protocol; on all other platforms the subagent uses Kerberos.

## Control row activation

The following constraints apply to the activation of control rows in `adConnectControlTable`:

- If a control row's `adConnectControlConnectTo` object has a value other than `genericConnect` or `domainMembers`, the control row may only be activated if its `adConnectControlConnectionString` object contains one or more valid connection strings.
- If a control row's `adConnectControlUsername` object specifies a username, the control row may only be activated if its `adConnectControlPassword` object contains a password.

## Monitoring Active Directory objects

Use the `adMonitor` group to monitor the content of Active Directory objects.

### About this task

#### Procedure

To monitor an object, create a control row in `adMonitorControlTable`:

1. Set `adMonitorControlStatus` to `createAndWait(5)`.
2. Set `adMonitorSearchType` to indicate where the objects you wish to monitor are located, either in the local domain controller or the global catalog.
3. Set `adMonitorSearchString` to the LDAP search string defining the objects that you wish to monitor.
4. If you wish to monitor attributes of those objects, list the attributes in `adMonitorControlAttributes`, with each attribute name separated by a space character.
5. Set `adMonitorControlInterval` to the interval at which you wish to monitor the objects.
6. Set `adConnectControlNumberBuckets` to the number of data rows you wish to store in the data table.
7. Activate the control row by setting `adMonitorControlStatus` to `active(1)`.

### Specifying search strings

The `adMonitorControlSearchString` object specifies the LDAP search string of the objects you wish to monitor, such as `(objectClass=user)`.

All search strings must be contained within parentheses `()`. For more details on LDAP search strings, see RFC 2254.

For example, to monitor the number of collisions in the domain, search for the string that is appended to the common name of an object when the collision occurs by using the LDAP string `(CN=*\0ACNF:*)`.

## Object monitoring and attribute monitoring

Two types of monitoring are available. Object monitoring returns the number of objects matching the search string. Attribute monitoring returns data about the attributes of each object matching the search string.

- For object monitoring, leave the `adMonitorControlAttributes` object empty.  
In the results stored in the data table, the object `adMonitorDataIntegerVal` indicates the total number of objects that match the search string. The objects `adMonitorDataStringVal`, `adMonitorDataValueCount` and `adMonitorDataValueStatus` have no meaning.
- For attribute monitoring, set the value of the `adMonitorControlAttributes` to the name of the attribute or attributes you wish to monitor. Separate each attribute name with a space character.

The results in the data table contain a row for each attribute whose name matches one of those specified by `adMonitorControlAttributes` for every object matching the search string. The row objects `adMonitorDataIntegerVal`, `adMonitorDataStringVal`, `adMonitorDataValueCount` and `adMonitorDataValueStatus` provide details about the attribute and its value.

## Generating events based on attribute values

You can use the `genalarm` to generate events based on the value of an attribute, however monitoring string-valued attributes requires special handling.

In some applications, you may need to generate events based on the attributes of Active Directory objects. The `genalarm` subagent provides facilities for generating events based on the value of an object; however, it is only able to monitor integer-based types. This means that it is not possible to use `genalarm` to generate events according to the value of a monitored attribute because the object that stores the attribute's value, `adMonitorDataStringVal`, is of string type.

However, the `adMonitorDataValueStatus` object indicates whether the value of an attribute has changed from one sample to the next. By configuring a `genalarm` control row to monitor this object, you can generate events in response to changes in an attribute's value.

For example, you might wish to generate an event in response to changes in the email address of any user matching a specific search string. The `lastModified` or `whenChanged` attributes provide an indication of a change to the email address, so either of these objects would provide the basis for generating events in response to a change in email address.

## Inivars

The `activedir` subagent provides an `inivar` for configuring its operation.

The `inivar` is listed in Table 320.

Table 320. *activedir* subagent *inivars*

Inivar	Type	Description
ActivedirForceLoad	string	Forces the subagent to load on host machines that are not Active Directory domain controllers;  no - The subagent is not forced to load.  yes - Forces the subagent to load.

For general information about inivars, see the *Netcool/SSM Administration Guide*.

## SNMPv1 compatibility

The activeDir MIB module uses the Counter64 data type, so it is not compatible with SNMPv1. Only use it with SNMPv2 and later.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Scalars

The activedirglobal command sets the value of various scalar objects.

The general syntax of this commands is:

`activedirglobal property=value`

Table 321 lists the properties supported in these commands.

*Table 321. Configuration command parameters - activedirglobal*

Property	Type	Description	Sets MIB object
collint	unsigned	The collection interval (in seconds).	adCollectionInterval
password	string	The password used for authentication.	adDefaultPassword
username	string	The default username used for authentication.	adDefaultUsername

## Connection control table

The activedirconn commands create rows in the connection control table (adConnectControlTable).

The general syntax of these commands is:

`activedirconn property=value`  
`activedirconn create [property=value ...]`  
`activedirconn reset`

Table 322 lists the properties supported in these commands.

*Table 322. Configuration command parameters - activedirconn*

Property	Type	Description	Sets MIB object
auth	enum	The type of security used in the connection:  authOnly  encrypt	Authentication
buckets	integer	The number of data table rows requested.	NumberBuckets

Table 322. Configuration command parameters - *activedirconn* (continued)

Property	Type	Description	Sets MIB object
connstr	string	The monitored entity.	ConnectionString
connto	enum	Selects an in-built connection string:  domainMaster  domainMembers  domainsInSite  gc  genConnect  infraMaster  pdc  ridMaster  schemaMaster  trustedDomains	ConnectTo
datacontrol	integer	Data control:  0 - Suspends data collection  1 - Enables data collection	DataControl
interval	unsigned	The interval (in seconds) at which the subagent connects to the target entity.	Interval
pass	string	The password used for authentication on the monitored entity.	Password
user	string	The username used for authentication on the monitored entity.	Username

## Monitor control table

The *activedirmon* commands create rows in the monitor control table (*adMonitorControlTable*).

The general syntax of these commands is:

```
activedirmon property=value
activedirmon create [property=value ...]
activedirmon reset
```

Table 323 lists the properties supported in these commands.

Table 323. Configuration command parameters - *activedirmon*

Property	Type	Description	Sets MIB object
attribs	string	The attributes to be monitored in the search results.	Attributes
datacontrol	int	Data control:  0 - Suspends data collection  1 - Enables data collection	DataControl

Table 323. Configuration command parameters - activedirmon (continued)

Property	Type	Description	Sets MIB object
interval	unsigned	The interval (in seconds) at which the search is performed.	Interval
searchstr	string	A string in LDAP format that specifies the name of the objects to be searched for.	SearchString
type	enum	The type of search to perform:  gc  localSearch	SearchType

## Examples

These examples demonstrate how to use the activedir subagent to perform specific tasks.

### Monitoring an LDAP server

Monitor an LDAP server domain controller in the domain mydomain.com and poll it every minute. The LDAP server requires authentication credentials admin\_user/admin\_pass:

```
subagent load activediractivedirconn reset
activedirconn connstr="LDAP://dc.mydomain.com"
activedirconn auth=authOnly
activedirconn user=admin_user
activedirconn pass=admin_pass
activedirconn connnto=genConnect
activedirconn interval=60
activedirconn buckets=5
activedirconn create
```

### Monitoring group policy objects

Monitor all Group Policy Objects in the domain for any changes. Check these objects at five-minute intervals:

```
subagent load activedir
activedirmon reset
activedirmon searchstr="(objectClass=groupPolicyContainer)"
activedirmon interval=300
activedirmon create
```

### Monitoring the number of users

Monitor the number of users in the local domain controller on a 12 hourly basis:

```
subagent load activedir
activedir reset
activedirmon searchstr="(&(objectClass=user)(objectCategory=person))"
activedir type=localSearch
activedirmon interval=43200
activedirmon create
```

### Monitoring invalid passwords

Monitor the number of invalid passwords entered by users in the global catalog on an hourly basis:

```

subagent load activedir
interval=3600
activedir reset
activedirmon searchstr="(&(objectClass=user)(objectCategory=person))"
activedirmon attribs=badPwdCount
activedirmon type=gc
activedirmon interval=$_interval
activedirmon create

```

## MIB module

The activeDir MIB is a subtree of networkharmoni (1977).

The subtree is shown in Figure 67.

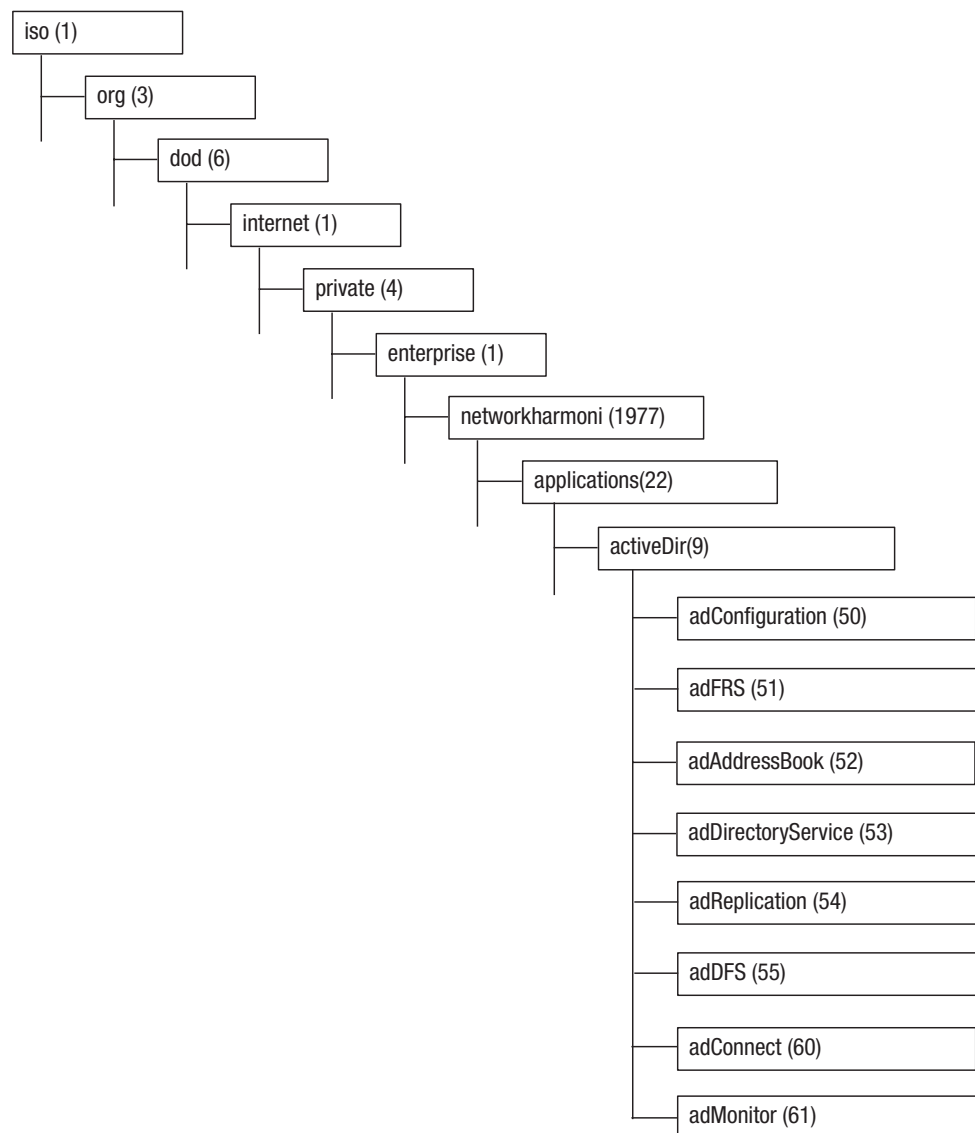


Figure 67. OID tree diagram of the activeDir MIB module

This section provides a summary of the objects defined in the activeDir MIB module. For detailed information on all objects in the module, see the `activedir-mib.mib` document located in the `mibs` subdirectory of the Netcool/SSM

installation.

## MIB tables

The activeDir MIB provides a standard SNMP interface.

The MIB module defines the following groups:

- Domain controller configuration group, adConfiguration
- File replication service group, adFRS
- Address book group, adAddressBook
- Directory service group, adDirectoryService
- Replication management group, adReplication
- Distributed file system group, adDFS
- Connection group, adConnect
- Monitor group, adMonitor

The MIB also contains a number of general configuration objects that define settings used by the subagent to access an active directory domain controller. Table 324 lists these objects.

*Table 324. activeDir configuration scalars*

Object	Description
adCollectionInterval	The interval to use when collecting performance statistics. A lower value increases the accuracy of statistics but increase the CPU load of the subagent. This is only relevant for per second statistics.
adDefaultPassword	Sets the password supplied by the activedir subagent when authenticating itself on the domain controller.
adDefaultUsername	Sets the username supplied by the activedir subagent when authenticating itself on the domain controller. If this object is empty, the subagent attempts authentication using the credentials of the Netcool/SSM installation.

## Domain controller configuration

The domain controller configuration group (adConfiguration) provides scalar objects that store basic configuration information about the Active Directory domain controller.

Table 325 lists the scalar objects in the adConfiguration group.

*Table 325. Active Directory domain controller configuration group scalar objects*

Scalar object	Description
DefaultNamingContext	The default naming context of the domain of which this domain controller is a member.
DomainMaster	The domain naming master for this forest.
DomainName	The name of the domain of which this domain controller is a member.
ForestName	The name of the forest of which this domain controller is a member.
GlobalCatalog	The global catalog for this forest.
InfrastructureMaster	The Infrastructure master for this domain.



Table 325. Active Directory domain controller configuration group scalar objects (continued)

Scalar object	Description
IsDomainMaster	True if this host is the domain naming master for this forest.
IsGlobalCatalog	True if this host is the Global Catalog for this forest.
IsInfrastructureMaster	True if this host is the Infrastructure master for this domain.
IsPDC	True if this host is the Primary Domain Controller emulator for this domain.
IsRIDMaster	True if this host is the RID master for this domain.
IsSchemaMaster	True if this host is the Schema master for this forest.
Mode	Indicates the mode that the domain controller is operating in. If the domain controller supports Windows clients, the mode will be mixed, otherwise the mode will be native.
PDC	The Primary Domain Controller emulator for this domain.
RIDMaster	The RID master for this domain.
RolePlacement	<p>This measures the correctness of FSMO roles the domain controller assumes. Particular combinations of roles are not desirable. This object may not be relevant to particular domain topologies, such as a domain with only one domain controller.</p> <p>noError(1) - The role placement for this server follows recommended practices.</p> <p>infIsGC(2) - This domain controller assumes the Infrastructure master role and is a Global Catalog. This is not a recommended practice.</p> <p>domainIsNotGC(3) - This domain controller assumes the Domain Naming role but is not a Global Catalog. This is not a recommended practice.</p> <p>schemaIsNotDomain(4) - This domain controller is the Schema master for this forest but is not the Domain Naming master. This is not a recommended practice.</p> <p>domainIsNotSchema(5) - This domain controller is the Domain Naming master for this forest but is not the Schema master. This is not a recommended practice.</p>
SchemaMaster	The Schema master for this forest.
ServerName	The distinguished name of the Active Directory domain controller.
SiteName	The name of the site of which this domain controller is a member.
SupportedCapabilities	Supported capabilities of the domain controller.

## File replication service group

The file replication service group (adFRS) provides scalar objects that store information and statistics about the Active Directory domain controller's file replication service (FRS).

Table 326 lists the scalar objects in the adFRS group.

*Table 326. File replica service group scalar objects*

Scalar object	Description
ChangeOrdersReceived	The number of change notifications received from inbound partners.
ChangeOrdersSent	The number of change notifications sent out to outbound partners.
FilesInstalled	The number of file updates and deletes performed on this replica set member.
FilesInstalledErrors	The cumulative number of file updates that were aborted because of a non-recoverable error.
FRSServiceStatus	The status of the File Replication Service process.
PacketsReceived	The number of packets received locally. These packets can be change notifications, file data or other command packets.
PacketsReceivedInError	The cumulative number of FRS data or control packets received that were out of rev or otherwise malformed.
PacketsSent	The number of packets sent. These packets can be change notifications, file data, or other command packets.
PacketsSentInError	The cumulative number of FRS data or control packets not sent to an outbound partner because of an error condition.
StagingSpaceFree	The amount of free space in the staging directory used by FRS to temporarily store files before they are replicated (in KB).
USNRecordsAccepted	The number of records that are accepted for replication. A high rate of change for this value (about one every five seconds) indicates a lot of replication traffic. This can cause replication latency.

## Address book group

The address book group (adAddressBook) provides scalar objects that store information and statistics about the Active Directory domain controller's address book.

Table 327 lists the scalar objects in the adAddressBook group.

*Table 327. Address book group scalar objects*

Scalar object	Description
ABBrowseRate	The rate at which Address Book clients perform browse operations (per second).
ABClientSessions	The number of connected Address Book client sessions.
ABMatchRate	The rate at which Address Book clients perform find operations (per second).
ABPropertyReads	The rate at which Address Book clients perform property read operations (per second).

Table 327. Address book group scalar objects (continued)

Scalar object	Description
ABProxyLookupRate	The rate at which proxy clients perform search operations (per second).
ABSearches	The rate at which Address Book clients perform key search operations (per second).
ANRRate	The rate at which Address Book clients perform Ambiguous Name Resolutions operations (per second).

## Directory service group

The directory service group (adDirectoryService) provides a set of scalar objects that store information and statistics about the Active Directory domain controller's directory service.

adDirectoryService also contains the groups:

- Directory service read group
- Directory service write group
- Directory service search group

Table 328 lists the scalar objects in the adDirectoryService group.

Table 328. Directory service group scalar objects

Scalar object	Description
ClientBindRate	The number of ntdsapi.dll binds per second serviced by this domain controller.
ClientNameTranslations	The number of ntdsapi.dll name translations per second serviced by this domain controller.
DatabaseSize	The size of the Active Directory database on this domain controller (in KB).
DSNameCacheHitRate	The percentage of directory object name component look ups that are satisfied out of the DSA's name cache.
DSThreadsInUse	The current number of threads in use by the directory service (not the directory service process). This value represents the number of threads currently servicing client API calls and can be used to indicate whether additional processors could be of benefit.
IsAdvertised	True if the domain controller is advertised in the DNS records.
ISMServiceStatus	The status of the Inter-Site Messaging process.
KDCServiceStatus	The status of the Kerberos Key Distribution Center process.
KerberosConnections	The number of times per second that clients use a ticket to this domain controller to authenticate to this domain controller.
LDAPClientSessions	The number of connected LDAP client sessions.
NetLogonServiceStatus	The status of the NetLogon process.
NTLMConnections	The number of NTLM authentications per second serviced by this domain controller.
ServerBindRate	The number of domain controller-to-domain controller binds per second serviced by this domain controller.

Table 328. Directory service group scalar objects (continued)

Scalar object	Description
ServerNameTranslations	The number of DC-to-DC name translations per second serviced by this domain controller.
W32TimeServiceStatus	The status of the Windows Time Service process.
XDSCClientSessions	The number of connected Extended Directory Service client sessions. This indicates the number of connections from other Windows services and the Windows Administrator program.

## Directory service read group

The directory service read group (adDSRead) provides a set of scalar objects that store statistics about the Active Directory domain controller's directory service read operations. Table 329 lists the scalar objects in the adDSRead group.

Table 329. Directory service read operations group scalar objects

Scalar Object	Description
DRAReads	% of directory reads coming from DRA (Directory Resource Administrator).
KCCReads	% of directory reads coming from KCC (Knowledge Consistency Checker).
LSAReads	% of directory reads coming from LSA (Local Security Authority).
NSPIReads	% of directory reads coming from NSPI (Named Server Provider Interface).
OtherReads	% of directory reads coming from other sources.
SAMReads	% of directory reads coming from SAM (Security Accounts Manager).
TotalReads	The total number of directory reads per second.
XDSReads	% of directory reads coming from XDS (Extended Directory Service).

## Directory service write group

The directory service write group (adDSWrite) provides a set of scalar objects that store statistics about the Active Directory domain controller's directory service write operations. Table 330 lists the scalar objects in the adDSWrite group.

Table 330. Directory service write operations group scalar objects

Scalar Objects	Description
DRAWrites	% of directory writes coming from DRA (Directory Resource Administrator).
KCCWrites	% of directory writes coming from KCC (Knowledge Consistency Checker).
LDAPWrites	% of directory writes coming from LDAP.
LSAWrites	% of directory writes coming from LSA (Local Security Authority).
NSPIWrites	% of directory writes coming from NSPI (Named Server Provider Interface).
OtherWrites	% of directory writes coming from other sources.
SAMWrites	% of directory writes coming from SAM (Security Accounts Manager).

Table 330. Directory service write operations group scalar objects (continued)

Scalar Objects	Description
TotalWrites	The total number of directory writes per second.
XDSWrites	% of directory writes coming from XDS (Extended Directory Service).

## Directory service search group

The directory service search group (adDSSearch) provides a set of scalar objects that store statistics about the Active Directory domain controller's directory service search operations. Table 331 lists the scalar objects in the adDSSearch group.

Table 331. Directory service search operations group scalar objects

Scalar Object	Description
DRASearches	% of directory searches coming from DRA (Directory Resource Administrator).
KCCSearches	% of directory searches coming from KCC (Knowledge Consistency Checker).
LDAPSearches	% of directory searches coming from LDAP.
LSASearches	% of directory searches coming from LSA (Local Security Authority).
NSPISearches	% of directory searches coming from NSPI (Named Server Provider Interface).
OtherSearches	% of directory searches coming from other sources.
SAMSearches	% of directory searches coming from SAM (Security Accounts Manager).
TotalSearches	The total number of directory searches per second.
XDSSearches	% of directory searches coming from XDS (Extended Directory Service).

## Replication management group

The replication management group (adReplication) provides a set of scalar objects that store information and statistics about database replication on the Active Directory domain controller.

adReplication also contains:

- The replication table
- The inbound statistics group
- The outbound statistics group

Table 332 lists the scalar objects in the adReplication group.

Table 332. Replication management group scalar objects

Scalar Object	Description
HighestCommittedUSN	Highest USN (Update Sequence Number) that this domain controller has committed.
HighestIssuedUSN	Highest USN (Update Sequence Number) that this domain controller has issued.

Table 332. Replication management group scalar objects (continued)

Scalar Object	Description
KCCStatus	Status of the Knowledge Consistency Checker (KCC). It is possible to change the state of KCC via this object. The values are:  enabled(0)  noIntraSite(1)  noInterSite(16)  disabled(17)
PendingSynchronizations	The number of directory synchronizations that are queued for this server but not yet processed.
SyncRequestsRatio	The percentage of sync requests that complete successfully.
SysvolShared	True if there is a valid SYSVOL share on this domain controller. If SYSVOL is not shared, replication issues will result.

## Replication table

The replication table (adReplicationTable) stores replication information about inbound replication partners. Each table row contains information about a specific replication partner. Table 333 lists the row objects in adReplicationTable.

Table 333. Replication table row objects

Row object	Description
LastAttempt	Time of last attempted replication.
LastSuccess	Time of last successful replication.
NumberOfFailures	The number of consecutive failures since the last successful replication.
Partner	The name of the replication partner.

## Inbound DRA group

The inbound DRA group (adReplicationInbound) provides scalar objects that store statistics related to inbound replications monitored by the directory replication agent (DRA). Table 334 lists the scalar objects in the adReplicationInbound group.

Table 334. Inbound DRA group scalar objects

Scalar object	Description
BytesTotal	Total number of inbound replicated bytes. This is determined as the sum of the number of uncompressed bytes (those never compressed) and the number of compressed bytes (after compression).
DNValues	The number of object property values received from inbound replication partners that are Distinguished Names per second. DN-values, such as group or distribution list memberships, are generally more expensive to apply than other kinds of values.

Table 334. Inbound DRA group scalar objects (continued)

Scalar object	Description
FullSyncObjects	The number of objects remaining until the full sync completes.
InterSiteBytesCompressed	Original size in bytes of inbound compressed replication data (size before compression) from DSAs in other sites per second.
InterSiteBytesUncompressed	Compressed size in bytes of inbound compressed replication data from DSAs in other sites per second.
IntraSiteBytes	The number of inbound replicated bytes that were not compressed at the source and originate from DSAs in the same site per second.
ObjectsApplied	The rate at which replication updates received from replication partners are applied by the local directory service per second. This count excludes changes that are received but not applied. This indicates how much replication update activity is occurring on the server as a result of changes generated on other servers.
ObjectsFiltered	The number of objects received from inbound replication partners that contained no updates that needed to be applied per second.
ObjectsTotal	The number of objects received from neighbors through inbound replication per second.
ObjectUpdatesRemaining	The number of object updates received in the current directory replication update packet that have not yet been applied to the local server per second.
PropertiesApplied	The number of properties that are updated per second due to incoming property winning the reconciliation logic.
PropertiesFiltered	The number of property changes that are received per second during the replication.
PropertiesTotal	The number of object properties received from inbound replication partners per second.
ValuesTotal	The number of object property values received from inbound replication partners per second. Each inbound object has one or more properties, and each property has zero or more values. Zero values indicates property removal.

## Outbound DRA group

The outbound DRA group (adReplicationOutbound) provides scalar objects that store statistics related to outbound replications monitored by the directory replication agent (DRA). Table 335 lists the scalar objects in the adReplicationOutbound group.

Table 335. Outbound DRA group scalar objects

Scalar object	Description
BytesTotal	Total number of outbound replicated bytes per second. This is determined as the sum of the number of uncompressed bytes (never compressed) and the number of compressed bytes (after compression).

Table 335. Outbound DRA group scalar objects (continued)

Scalar object	Description
DNValues	Number of object property values containing Distinguished Names sent to outbound replication partners per second. DN-values are generally more expensive to read than other kinds of values.
InterSiteBytesCompressed	Original size in bytes of outbound compressed replication data (size before compression) for DSAs in other sites per second.
InterSiteBytesUncompressed	Compressed size in bytes of outbound compressed replication data (size after compression) for DSAs in other sites per second.
IntraSiteBytes	Number of bytes replicated per second out that were not compressed for DSAs in the same site.
ObjectsFiltered	Number of objects looked at by outbound replication that were determined to have no updates that the outbound partner did not already have (per second).
ObjectsTotal	Number of outbound replicated objects per second.
Properties	Number of outbound replicated properties per second.
Values	Number of object property values sent to outbound replication partners per second.

## Distributed file system group

The distributed file system group (adDFS) contains the distributed file system control and target tables.

### Control table

The distributed file system control table (adDFSControlTable) stores state information about distributed file system (DFS) roots and links. Each row in the table contains information relating to one DFS root or link. Table 336 lists the row objects in adDFSControlTable.

Table 336. Distributed file system control table

Row object	Description
adDFSControlComment	The comment for this DFS root or link.
adDFSControlIndex	Uniquely identifies this row in this table.
adDFSControlPath	The DFS path that this control row will gather statistics from.
adDFSControlState	The state of the DFS root or link.
adDFSControlTargets	The number of targets for this DFS root or link.

### Target table

The distributed file system target table (adDFSTargetTable) stores state information about DFS targets. Each row in the table contains information relating to one DFS target. Table 337 on page 439 lists the row objects in adDFSTargetTable.



Table 337. Distributed file system target table

Row object	Description
Index	Together with adDFSControlIndex, this object uniquely identifies this row in the table.
Server	Name of the target server where this share resides.
Share	Name of the share on the target server.
Status	Status of the share.

## Connection group

The connection group provides facilities for monitoring connectivity to Active Directory entities.

This group contains the following tables:

- Connection control table
- Connection data table

## Connection control table

The connection control table (adConnectControlTable) defines monitors for connections to Active Directory entities. Each row monitors the connection to one or more entities, which it attempts to connect to at regular intervals. If the connection is successful, the time taken for this connection is recorded in the data table; otherwise the error code and any error description available are recorded. Table 338 lists the row objects in adConnectControlTable.

Table 338. Connection control table

Row object	Description
Authentication	Specifies the type of security used in the connection:  auth(1) - The connection is made using the username and password specified by the adConnectControlUsername and adConnectControlPassword objects.  encrypt(2) - The connection is made the same way as in auth(1) mode; however, all data is encrypted using SSL.
ConnectString	Specifies the monitored entity. For more details, see “Monitoring Active Directory connectivity” on page 422.

Table 338. Connection control table (continued)

Row object	Description
ConnectTo	<p>In-built connect string for important entities within the domain: The valid values are:</p> <p>genericConnect(1)</p> <p>pdc(2)</p> <p>ridMaster(3)</p> <p>schemaMaster(4)</p> <p>domainMaster(5)</p> <p>infraMaster(6)</p> <p>globalCatalog(7)</p> <p>trustedDomains(8)</p> <p>domainsInSite(9)</p> <p>domainMembers(10)</p> <p>For more information, see “Monitoring Active Directory connectivity” on page 422.</p>
DataControl	<p>Sets the data collection status of the control row:</p> <p>on(1) - Enables data collection</p> <p>off(2) - Suspends data collection</p>
Index	Uniquely identifies this row in the table.
Interval	Sets the interval (in seconds) at which the subagent attempts to connect to the target entity. If the value of this object is 0 only one connection is attempted. The maximum interval is one day.
NumberBuckets	Sets the number of rows in the data table allocated to this control row.
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
Password	Specifies the password that the subagent uses in combination with adConnectControlUsername to connect to the target entity.
Status	SNMPv2 row status. Controls creation, activation and deletion of the control row. See “Monitoring Active Directory connectivity” on page 422 for more details.
Username	Specifies the username that the subagent uses to connect to the target entity. Use the format domain name\username.

## Connection data table

The connection data table (adConnectDataTable) stores data about the connections to Active Directory entities monitored by control rows in adConnectControlTable. Each data table row stores the data generated by a connection attempt.

The number of data table rows associated with each control row is set by the control row's adConnectControlNumberBuckets object. When the number of data table rows allocated to a control row exceeds the value of this object, the oldest

data table row is deleted each time a new row is created. Depending on the value of `adConnectControlConnectTo`, one control row may monitor several Active Directory entities. In such cases, the number of data table rows specified by `adConnectControlNumberBuckets` is allocated to each entity monitored.

Table 339 lists the row objects in `adConnectDataTable`.

*Table 339. Connection data table*

Row object	Description
ConnectString	Stores the binding string used in this connection attempt, up to a maximum length of 128 characters.
ErrorDescription	A description of the error. If no description is available, the error code from the internal API is displayed as a hexadecimal string.
Index	Uniquely identifies this row in the table, in combination with <code>adConnectControlIndex</code> and <code>adConnectDataConnectString</code> .
Status	The error code returned by the connection. If the connection was not successful, an error description may be retrieved; however, some errors do not have any descriptions.
TimeStamp	The value of <code>sysUpTime</code> when the connection was attempted.
TimeTaken	The total time taken for this connection attempt.

## Monitor group

The monitor group (`adMonitor`) provides facilities for monitoring objects in the local Active Directory domain controller or in the domain global catalog.

The `adMonitor` group contains the following tables:

- Monitor control table
- Monitor data table

## Monitor control table

The monitor control table (`adMonitorControlTable`) defines object monitors. Each control row specifies a search string that identifies the objects to be monitored, the type of monitoring and the location of the objects — either in the local domain controller or the global catalog.

Table 340 lists the row objects in `adMonitorControlTable`.

*Table 340. Monitor control table (adMonitorControlTable)*

Row object	Description
Attributes	Specifies the attributes to be monitored in the search results. See “Object monitoring and attribute monitoring” on page 425 for more details.
DataControl	Sets the monitoring status of the control row:  on(1) - Enables data collection  off(2) - Suspends data collection
Index	Uniquely identifies this row in the table.
Interval	Sets the interval (in seconds) at which the search is performed. If the value of this object is 0, only one search is performed. The maximum value of this interval is 86400 seconds (one day).

Table 340. Monitor control table (adMonitorControlTable) (continued)

Row object	Description
Owner	The entity that configured this entry and is therefore using the resources assigned to it.
SearchString	A string in LDAP format that specifies the objects for which to search. See “Monitoring specific entities” on page 423 for more details.
SearchType	The type of search to perform:  local(1) - Search the local domain controller  globalCatalog(2) - Search the global catalog
Status	SNMPv2 row status. Controls creation, activation and deletion of the control row.

## Monitor data table

The monitor data table (adMonitorDataTable) stores data generated by monitors defined in adMonitorControlTable. The type of data stored in each data table row depends on the type of monitoring performed by the corresponding control row. For object monitoring, the data table rows indicate the number of objects matching the search string; for attribute monitoring, data table rows store data about the attributes of objects matching the search string.

Table 341 lists the row objects in adMonitorDataTable.

Table 341. Monitor data table (adMonitorDataTable)

Row object	Description
Attribute	Indicates the attribute being monitored. This object only contains a value for attribute monitoring.
ErrorDescription	A description of the error. If no description is available, this object contains the error code from the internal API stored as a hexadecimal string.
Index	Along with adMonitorControlIndex and adMonitorDataObject, this object uniquely identifies this row in the table.
IntegerVal	For object monitoring, this object indicates the total number of objects that match the search string. For attribute monitoring, this object indicates the integer value of the attribute if applicable. For multi-value attributes, only the first value is displayed.
Object	The name of the object being monitored. This object applies only to attribute monitoring.
Status	The error code returned from the search process. If the search was not successful, an error description may be retrieved. Note that some errors do not have any descriptions.
StringVal	For attribute monitoring, this object contains a string representation of the current values of this attribute if applicable.
TimeTaken	Time taken for the search.
ValueCount	For attribute monitoring, this object holds the number of values for the specified attribute. This value is only relevant for multi-value attributes; for single-value attributes it is always 1.
ValueStatus	For attribute monitoring, this object indicates whether the value of the attribute has changed during the interval set by adMonitorControlInterval.





---

## Chapter 39. Netcool/ASM for IBM Lotus Notes/Domino Server

The Netcool/ASM for IBM Lotus Notes/Domino Server delivers performance data about IBM Lotus Domino servers by polling metrics provided by the Lotus Domino API. It can monitor multiple Domino server instances running on one machine.

Netcool/ASM for IBM Lotus Notes/Domino Server is also called Netcool/ASM for Lotus. It consists of the `lotus` subagent and the `lotus` MIB module.

---

### Component files

Netcool/ASM for Lotus is comprised of a set of component files.

Table 342 lists the component files of Netcool/ASM for Lotus.

*Table 342. Netcool/ASM for IBM Lotus Domino component files*

File	Location	Description
<code>lotus.dll</code> (Windows) <code>liblotus.so/.sl</code> (UNIX)	<code>bin</code>	Binary implementation of the <code>lotus</code> subagent.
<code>lotus-mib.mib</code>	<code>mibs</code>	<code>lotus</code> MIB definition document.
<code>lotus.oid</code>	<code>config/oid</code>	<code>lotus</code> subagent object identifier file.

---

### Guidelines

Use Netcool/ASM for Lotus to monitor Domino servers running on the local machine.

To load the `lotus` subagent, use the command:

```
subagent load lotus
```

**Note:** Netcool/SSM and Netcool/ASM for Lotus must be running on the same machine as the Domino server that you wish to monitor. You cannot use Netcool/ASM for Lotus to monitor a server remotely.

### Configuring the subagent

The `lotus` subagent provides the inivar `LotusBinPath`, which specifies the directory in which the Lotus Domino API library file is located.

You can set or modify this inivar in the following ways:

- Create or modify the definition of `LotusBinPath` in the Netcool/SSM configuration file `init.cfg`, for example:  
`LotusBinPath=d:\lotus\domino`
- Set the inivar using the `set inivar` command, for example:  
`set inivar LotusBinPath=/opt/lotus/notes/latest/sunspa`

Table 343 on page 446 lists the default name and location of the Domino API library file on each supported platform.

Table 343. Domino API library location and filename by platform

Platform	API Library Location	API Library Filename
AIX	/opt/lotus/notes/latest/ibmpow/	libnotes_r.a
Solaris	/opt/lotus/notes/latest/sunspa/	libnotes.so
Windows	drive:\lotus\domino\	nnotes.dll

For more information about inivars, see the *Netcool/SSM Administration Guide*.

## General monitoring procedure

To monitor Domino performance metrics, create control rows in the lotus MIB.

### Procedure

The general procedure for monitoring Domino server performance metrics is as follows:

1. Set `loSampleInterval` to the desired sample interval (in seconds).
2. Create a row in `loServerTable` for each Lotus Domino server that you wish to monitor and set the `loServerNotesDataDirectory` object to the location of the server's `notes.ini` file, for example `c:\lotus\domino\`.
3. Create a control row in `loControlTable` by setting `loControlStatus` to `createAndWait(5)`.
4. Set `loControlServer` to the index of the `loServerTable` row representing the Domino server instance that you wish to monitor.
5. Set `loControlFacility` to the name of the facility containing the metrics that you wish to monitor.  
Facility names are contained in `loFacilityName` objects in the table `loFacilityTable`.
6. Set `loControlVariable` to the name of the metric that you wish to monitor. Alternatively, you can supply a regular expression representing the name of one or more metrics.  
For more details about regular expressions, see Appendix D, "Regular expressions," on page 615.
7. If appropriate, set the `loControlRealOperator` and `loControlRealOperand` objects to values appropriate for converting real-valued metrics to integer values.
8. Activate the control row by setting `loControlStatus` to `active(1)`.  
The value of each selected metric is stored in the data table, one metric per row, and is updated at the interval set by `loSampleInterval`.

## Signalling changes made to the notes.ini file

If you make changes to the `notes.ini` file on any monitored Domino server, you must signal these changes to the lotus subagent by setting the object `loReSeedTaskTable` to `reseed(1)`.

### About this task

This prompts the subagent to re-scan the `notes.ini` file of each monitored Domino server and update the task table (`loTaskTable`) accordingly.



---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### Control table

The `lotus` commands create rows in the control table (`loControlTable`).

The general syntax of these commands is:

```
lotus property=value
lotus create [property=value ...]
lotus reset
```

Table 344 lists the properties supported in these commands.

*Table 344. Configuration command parameters - lotus*

Property	Type	Description	Sets MIB object
facility	string	The index of the row in the facility table that represents the facility to be monitored.	Facility
operand	int	The operand used in converting metrics to integers.	RealOperand
operator	enum	Specifies the operator applied to metrics to convert them to integers:  divide  multiply  none	RealOperator
server	int	The index of the row in the server table that represents the server to be monitored.	Server
variable	string	A regular expression specifying the name of the metrics to be monitored.	Variable

### Server table

The `lotusserver` command creates rows in the server table (`loServerTable`).

The general syntax of these command is:

```
lotusserver property=value
lotusserver create [property=value]
lotusserver reset
```

Table 345 on page 448 lists the properties supported in this command.

Table 345. Configuration command parameters - lotusserver

Property	Type	Description	Sets MIB object
notesdatadirectory	string	The fully qualified path to the directory containing the notes.ini file of the Domino server to be monitored.	NotesDataDirectory

## Scalars

The `lotusglobal` command sets the value of lotus scalar objects.

The general syntax of this command is:

```
lotusglobal property=value
```

Table 346 lists the properties supported in this command.

Table 346. Configuration command parameters - lotusglobal

Property	Type	Description	Sets MIB object
sampleinterval	int	The interval (in seconds) at which data is collected from the monitored servers.	loSampleInterval

---

## Examples

These examples demonstrate how to use the `lotus` subagent to perform simple monitoring tasks for a Domino server.

### Monitoring LDAP facility metrics

Monitor, at ten-minute intervals, all metrics contained in the LDAP facility on a Domino server whose `notes.ini` file is located at `d:\lotus\domino\`. Truncate any real values to integer values.

The configuration commands for implementing this task are:

```
subagent load lotus
lotusglobal sampleinterval=600
lotusserver notesdatadirectory=d:\lotus\domino\
lotusserver create
serverIndex=$?
lotus server=$serverIndex
lotus facility=LDAP
lotus variable=.*
lotus operator=none
lotus create
```

### Monitoring the NET facility

Monitor any metric in the NET facility whose name contains the string `Bytes`. Truncate any real values to integer values using division by 1000.

The configuration commands for implementing this task are:

```
lotus reset
lotus server=$serverIndex
lotus facility=NET
```

```

lotus variable=.*Bytes.*
lotus operator=divide
lotus operand=1000
lotus create

```

## Monitoring persistent server tasks

Generate an event when any persistent server tasks are not running. Persistent server tasks are those listed in the Domino server's notes.ini file  
d:\lotus\domino\.

The configuration commands for implementing this task are:

```

subagent load rmonc
event type=snmp-trap
event community=public
event descr="Lotus Domino Server Task Event"
event create
ServerTaskFailEvent=$?

subagent load lotus
lotusglobal sampleinterval=60
lotusserver notesdatadirectory=d:\lotus\domino\
lotusserver create
serverIndex=$?

subagent load genalarm
genalarm var="$loTaskState.$serverIndex.0.*"
genalarm vardescr="Lotus Domino Server Task Status"
genalarm interval=60
genalarm type=absolute
genalarm mode=match
genalarm matchvalue=0
genalarm matchduration=120
genalarm matchseverity=severe
genalarm matchdescr="Server task not running on Domino (notes.ini: $$11)"
genalarm matchevent=$SchedManFailEvent
genalarmvb store oid="$loServerNotesDataDirectory.$serverIndex"
genalarm create

```

**Note:** The index of the monitored loTaskState object is formed from the combination of serverIndex, loTaskType and loTaskName objects. The example monitors all persistent server tasks, so the value of this portion of the index is 0. The value of the loTaskName portion of the index is .\*, which selects server tasks with any name.

---

## MIB module

The lotus MIB is a subtree of networkharmoni(1977).

The subtree is shown in Figure 68 on page 450.

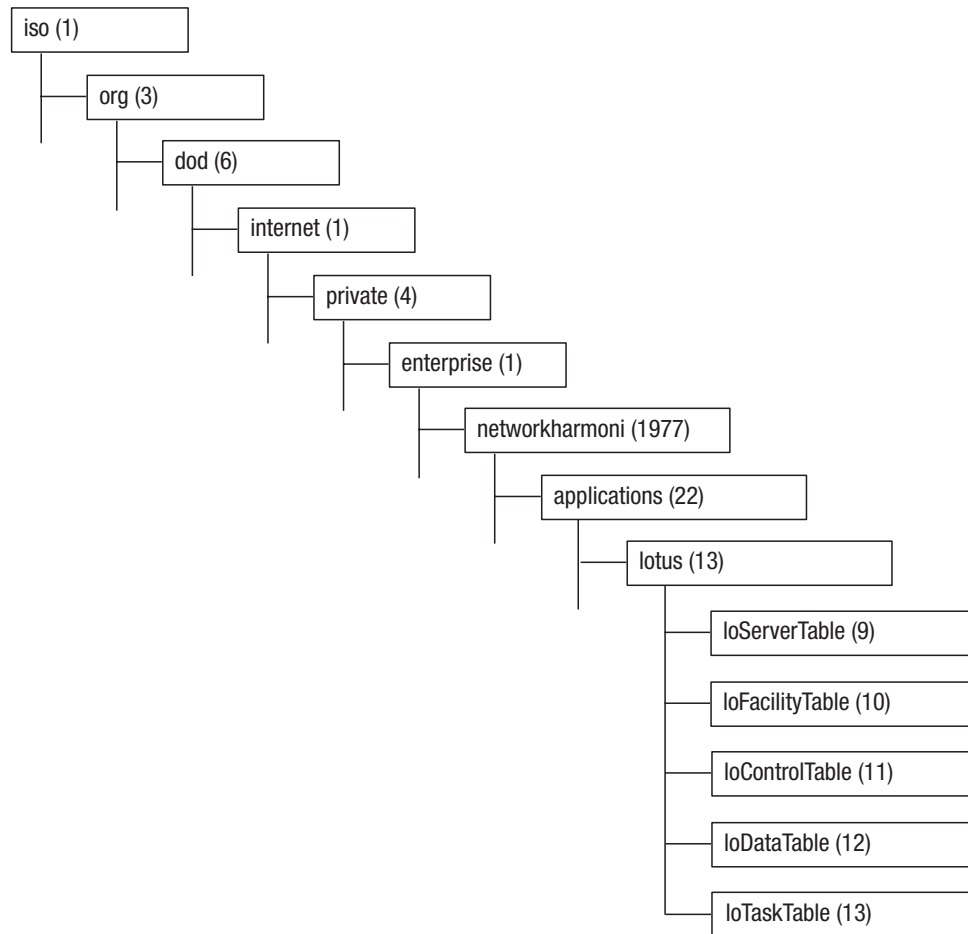


Figure 68. OID tree diagram for lotus MIB module

For detailed information on all the objects provided in this MIB see the MIB document in the mibs subdirectory of the Netcool/SSM installation.

## MIB objects and tables

The lotus MIB provides a standard SNMP interface.

The MIB module defines the following items:

- A group of scalar objects
- A server table, loServerTable
- A facility table, loFacilityTable
- A control table, loControlTable
- A data table, loDataTable
- A task table, loTaskTable

## Scalar objects

The lotus MIB provides several scalar objects that configure the operation of the lotus subagent.

Table 347 lists these objects.

*Table 347. lotus MIB scalar objects*

Object	Description
loReSeedTaskTable	Prompts the subagent to rescan the notes.ini files of all monitored Domino servers:  idle(0) - The passive value of the object.  reseed(1) - When the object is set to this value, the subagent re-scans the notes.ini file on each monitored server
loSampleInterval	Sets the interval (in seconds) at which data is collected from all monitored Domino servers. The minimum allowed sample interval is 60 seconds.

## Server table

The server table (loServerTable) provides information about all Domino server instances running on the host machine.

Table 348 lists the row objects in loServerTable.

*Table 348. Server table (loServerTable)*

Row object	Description
Index	Uniquely identifies this row in the table.
NotesDataDirectory	The fully qualified path to the directory containing the notes.ini file of the monitored Domino server to be monitored.
Owner	Indicates the entity that configured this entry and is therefore using the resources assigned to it.
Status	Indicates the status of the Domino server:  down(0) - Indicates that the server is not running  notMonitoring(1) - Netcool/ASM for Lotus cannot determine the status of the server  up(2) - Indicates that the server is running

## Facility table

The facility table (loFacilityTable) lists the facilities available on each Domino server identified in loServerTable. Each row in the table represents one facility found on a monitored Domino server. Use these rows as a reference when identifying the metrics you wish to monitor.

Table 349 lists the row objects in loFacilityTable.

*Table 349. Facility table (loFacilityTable)*

Row object	Description
Index	Uniquely identifies this row in the table.
Name	The name of the facility that this row represents.

## Control table

The control table (loControlTable) provides a facility for gathering Domino server performance metrics. Each row defines a monitor for one or more metrics identified by name.

Table 350 lists the row objects in loControlTable.

Table 350. Control table (loControlTable)

Row object	Description
Facility	Contains the name of the facility to which the monitored metrics belong.
Index	Uniquely identifies this row in the table.
LastUpdate	The time (relative to sysUpTime) when the last sample took place.
Owner	Indicates the entity that configured this entry and is therefore using the resources assigned to it.
RealOperand	<p>Specifies the operand used to create the integer representation of any real values.</p> <p>If the value of this object is zero and the value of loControlRealOperator is divide(3), the integer representation will be zero.</p>
RealOperator	<p>Selects the operator used to convert any real values monitored by this control row into integer values. The result of the conversion is stored in loDataValueLong. The available operators are:</p> <p>none(1) - No operation; this truncates the real value at the decimal point to produce an integer value</p> <p>multiply(2) - Multiplies the real value contained in loDataValueNumber by the value specified in loControlRealOperand and truncates the result to an integer value</p> <p>divide(3) - Divides the real value contained in loDataValueNumber by the value specified in loControlRealOperand and truncates the result to an integer value</p> <p><b>Note:</b> The value of the loDataValueNumber object is not affected by the operation specified in this object.</p>
Server	References by index value an entry in the server table loServerTable.
Status	SNMPv2 row status. Controls creation, activation and deletion of the control row.
Variable	A regular expression specifying the names of the metrics to be monitored. This expression only needs to match a substring of the name.

## Data table

The data table (loDataTable) stores data values of the performance metrics gathered by control rows. Each row in the data table is associated with one control row and stores the value of one performance metric. This value is updated at the interval set by loSampleInterval.

Table 351 on page 453 lists the row objects in loDataTable.

Table 351. Data table (loDataTable)

Row object	Description
ValueLong	Contains the signed integer value of the metric. If loDataValueType has the value number(3), this object contains an integer value derived from the real value of the metric according to the method specified by the loControlRealOperator object of the corresponding control row.
ValueNumber	Contains the real value of the metric.
ValueString	Contains the string value of the metric.
ValueTime	Contains the date/time value of the metric.
ValueType	Specifies the type of the metric that this data rows contains and indicates which object contains the value of the metric:  long(0) - A signed 32 bit integer value (loDataValueLong)  text(1) - A string value (loDataValueString)  timedate(2) - A time and date value (loDataValueTime)  number(3) - IEEE 64 bit floating value (loDataValueNumber)
VariableInstance	Index of the variable instance.
VariableName	Name of the variable monitored.

## Task table

The task table (loTaskTable) lists the tasks currently running on the Domino server, and those expected to be running.

Table 352 lists the row objects in loTaskTable.

Table 352. Task table (loTaskTable)

Row object	Description
Count	Indicates the total number of processes running under the same service task name.
DisplayName	The name of the task. This object contains the same data as loTaskName, however it does not act as an index, so it is accessible for inclusion in notifications.
Name	The name of the task.
State	Indicates the current status of the task:  notRunning(0) - The task is not currently running  running(1) - The task is currently running
Type	Indicates the task's type:  persistent(0) - The task is loaded automatically at server startup. Persistent tasks are those tasks listed in the ServerTasks setting in the notes.ini file  transient(1) - The task is loaded by some method other than the notes.ini ServerTasks setting





---

## Chapter 40. Netcool/ASM for Sybase ASE

Netcool Application Service Monitor for Sybase Adaptive Server Enterprise (Netcool/ASM for Sybase) provides performance monitoring facilities for a range of Sybase ASE database servers.

Netcool/ASM for Sybase loads and runs as a component of Netcool/SSM. It consists of the sybase subagent and the sybase MIB module.

The subagent collects performance metrics from Sybase ASE by querying internal performance metrics, injecting custom SQL queries, and extracting data from Sybase performance reports. It provides a number of methods for obtaining performance data about Sybase servers:

- Query internal performance data contained in the Sybase internal MDA performance tables.
- Submit SQL queries to Sybase ASE using the generic SQL executer.
- Extract performance data from reports generated by the Sybase `sp_sysmon` stored procedure.

---

### Component files

Netcool/ASM for Sybase is comprised of a set of component files.

Table 353 lists the component files of Netcool/ASM for Sybase.

*Table 353. Netcool/ASM for Sybase component files*

File	Location	Description
sybase.dll (Windows) libsybase.so/.sl (UNIX)	bin	Binary implementation.
sybasemc.dll (Windows) libsybasemc.so/.sl (UNIX)	bin	Support libraries.
sybase-mib.mib	mibs	MIB definition document.
sybase.oid	config/oid	Object identifier file.
sybase.cfg	config	Sample configuration file.

---

### Guidelines

To monitor a Sybase server using the sybase subagent, you must configure both the server and the subagent. Follow the steps provided in this information to complete these tasks.

## Deployment

Netcool/ASM for Sybase must be installed on the machine running the Sybase ASE server that you wish to monitor. If you wish to monitor Sybase ASE servers running on more than one machine, you must install Netcool/SSM on each machine.

## Configuring Sybase

If you wish to use Netcool/ASM for Sybase to monitor performance data contained in the Sybase MDA tables, you must prepare those tables prior to monitoring.

### Procedure

To prepare the tables:

1. Create a loopback server alias
2. Install the monitoring tables
3. Set monitoring configuration parameters
4. Grant monitoring privileges to a database user

### Creating the loopback server alias About this task

To create the loopback server alias, execute the following command using the `isql` utility:

```
sp_addserver loopback, null, servername
```

where *servername* is the name of the target Sybase ASE server.

### Installing the monitoring tables About this task

To install the MDA tables on the target Sybase server, execute the following monitor table installation script using the `isql` utility:

### Procedure

- On UNIX systems, execute `$SYBASE/ASE-12_5/scripts/installmontables`
- On Windows systems, execute `%SYBASE%\ASE-12_5\scripts\installmontables`

### Setting the monitoring configuration parameters About this task

To set the monitoring configuration parameters, execute the following commands using the `isql` utility:

```
sp_configure 'enable monitoring', 1
go
sp_configure 'sql text pipe active', 1
go
sp_configure 'sql text pipe max messages', 100
go
sp_configure 'plan text pipe active', 1
go
sp_configure 'plan text pipe max messages', 100
go
sp_configure 'statement pipe active', 1
go
```

```

sp_configure 'statement pipe max messages', 100
go
sp_configure 'errorlog pipe active', 1
go
sp_configure 'errorlog pipe max messages', 100
go
sp_configure 'deadlock pipe active', 1
go
sp_configure 'deadlock pipe max messages', 100
go
sp_configure 'wait event timing', 1
go
sp_configure 'process wait events', 1
go
sp_configure 'object lockwait timing', 1
go
sp_configure 'SQL batch capture', 1
go
sp_configure 'statement statistics active', 1
go
sp_configure 'per object statistics active', 1
go
sp_configure 'max SQL text monitored', 255
go

```

## Granting monitoring privileges

Netcool/ASM for Sybase requires a database user account to connect to the Sybase server and access the appropriate performance metrics. This user must have the `mon_role` privilege if you wish to monitor MDA performance statistics, and `sa_role` if you wish to extract data from the `sp_sysmon` report.

## About this task

To create a new Sybase database user, execute the following command using the `isql` utility:

```

sp_addlogin "username","password"
go

```

where *username* and *password* are the credentials of the new database user.

To assign the `mon_role` privilege to a Sybase database user, execute the following the command using the `isql` utility:

```

grant role mon_role to username
go

```

where *username* is the name of the database user.

## Locating the Sybase installation

The `SybaseHome` inivar specifies the location of the Sybase ASE installation. The subagent uses this inivar to locate the Sybase ASE installation if the `SYBASE` environment variable is not defined. Set it using the `set inivar` configuration command or by editing its definition in the `init.cfg` file.

To set the inivar from the command console, use the command:

```

set inivar SybaseHome=sybase_install_dir

```

## Loading the subagent

You must load the subagent on Netcool/SSM before monitoring a Sybase server.

### About this task

To load the sybase subagent, use the following configuration command:

```
subagent load sybase
```

## Encrypting passwords

Netcool/ASM for Sybase requires user credentials to access Sybase ASE. When configuring control rows in the sybase MIB, you must supply a Sybase username and password. To keep passwords secure, use the Netcool/SSM password encryption facility rather than storing passwords in clear text.

To encrypt a password, open a Netcool/SSM command console and enter the command:

```
encrypt password
```

where *password* is the password you wish to encrypt. In response, Netcool/SSM displays the encrypted password on the command console. Make note of the encrypted password and use it in place of the original password.

To pass an encrypted password as a property in a configuration command use the @= operator in place of the usual = operator as follows:

```
command password_property@=encrypted_password
```

For example, to supply the encrypted password WMQIa0j0LeVHA when configuring a control row in the MDA group, use the command:

```
sybasemda password@=WMQIa0j0LeVHA
```

## Threshold monitoring and event generation

The sybase subagent does not generate events, however you can use the genalarm subagent to monitor thresholds on objects in the sybase MIB and generate events in response to threshold violations.

The configuration file sybase.cfg located in the Netcool/SSM config directory contains a set of sample threshold monitors for a range of sybase MIB objects. You can edit the sybase.cfg file and configure the monitors and thresholds to suit your monitoring requirements.

To execute sybase.cfg, use the command

```
config execute sybase.cfg
```

**Tip:** If you use the sybase.cfg file, ensure that you set all servername, username, and password configuration command properties to correct values for the target Sybase ASE installation.

---

## Monitoring MDA performance statistics

The sybase subagent uses SQL select statements to retrieve performance data from the Sybase MDA tables, then stores it in the tables in the MDA group (sybaseMDAGroup). Use the sybaseMDAGroup to access low-level performance statistics contained in a Sybase server's internal MDA performance tables.

### General procedure

To monitor Sybase MDA performance data, create a control row in the MDA control table (sybaseMDAControlTable).

#### Procedure

1. Create a row by setting sybaseMDAControlStatus to createAndWait(5).
2. Set the sybaseMDAControlServerName object to the name of the Sybase server that you wish to monitor.
3. Set the sybaseSysMonControlUsername and sybaseSysMonControlPassword objects to the credentials of a database user with mon\_role privileges.  
For information about database user privileges, see “Granting monitoring privileges” on page 457.
4. Set the sybaseMDAControlSampleInterval object to the interval (in seconds) at which you wish the performance data to be updated.
5. Activate the control row by setting sybaseMDAControlStatus to active(1).  
The sybase subagent commences monitoring and updates data in the sybaseMDAGroup tables at the interval set by sybaseMDAControlSampleInterval.

#### Results

### Monitoring engine CPU utilization

Monitor the engine CPU utilization statistics for the Sybase server SYBASE1. Use the genalarm subagent to generate an alarm if the utilization exceeds 60%. Include the engine number and the CPU for which the engine has affinity in the alarm descriptions.

The configuration commands for creating this monitor are:

```
subagent load rmonc
```

```
#Create the warning event for threshold violation
event type=snmp-trap
event description="Sybase Warning"
event create
#Create the status event for threshold return to normal
_sybaseWarnEvtIdx=?
event description="Sybase Status"
event create
_sybaseStatusEvtIdx=?
```

```
subagent load sybase
```

```
#Create the control row for monitoring the server "SYBASE1" every minute
sybasemda servername="SYBASE1"
sybasemda username="SybaseUser"
sybasemda password@="encrypted_password"
sybasemda interval=60
sybasemda create
_ctrlIdx=?
```

```

subagent load genalarm

#Create the varbinds for inclusion in notifications
genalarmvb store oid="$sybaseMDAEngineEngineNumber.*"
genalarmvb store oid="$sybaseMDAEngineAffinitiedToCPU.*"

#Create the alarm for 60% CPU utilization
genalarm var="$sybaseMDAEngineCPUBusyPercent.*"
genalarm vardescr="Sybase: Engine CPU Utilization"
genalarm interval=$_genInterval
genalarm type=absolute
genalarm mode=risingContinuous
genalarm risethresh=60
genalarm riseduration=0
genalarm riseevent=$_sybaseWarnEvtIdx
genalarm riseseverity=severe
genalarm risedescr="Engine $$11 is using $$6% of CPU (with affinity for CPU $$12)"
genalarm fallthresh=60
genalarm fallduration=0
genalarm fallevent=$_sybaseStatusEvtIdx
genalarm fallseverity=info
genalarm falldescr="Engine $$11 has fallen to $$6% of CPU usage"
genalarm create

```

For a list of configuration commands available for the MDA group, see “MDA group” on page 472.

---

## Creating SQL queries

The sybase subagent provides a generic SQL query generator that enables you to submit any SQL query into target Sybase servers. Use the SQL Query group (sybaseSQLQueryGroup) to query the internal performance tables of Sybase ASE servers.

### General procedure

Create and execute SQL queries on a Sybase server using the sybaseSQLQueryGroup.

#### Procedure

The general procedure for creating SQL queries is:

1. Create a row in the SQL query control table (sybaseSQLQueryControlTable) identifying the Sybase server that you wish to monitor:
  - a. Set sybaseSQLQueryControlStatus to createAndWait(5).
  - b. Set sybaseSQLQueryControlServerName to the name of the Sybase server.
  - c. Set the sybaseSysMonControlUsername and sybaseSysMonControlPassword objects to the credentials of a database user with appropriate privileges. For information about database user privileges, see “Granting monitoring privileges” on page 457.
  - d. Set sybaseSQLQueryControlSampleInterval to the interval (in seconds) at which you wish the performance data to be updated.
  - e. Set sybaseSQLQueryInTableRowCount to the number of lines required for the SQL query text.
2. Create rows in the query input table (sybaseSQLQueryInTable) for each line of SQL query text. In each input table row, set sybaseSQLQueryInSQLText to the SQL query text.
3. Activate the control row by setting sybaseSQLQueryControlStatus to active(1).

The sybase subagent executes the query on the target Sybase server at the interval set by `sybaseSysMonControlSampleInterval` and stores the results of the query in the `sybaseSQLQueryResultsTable`.

## Constructing queries

The query input table (`sybaseSQLQueryInTable`) stores the SQL queries submitted to Sybase servers. Each row stores an entire query or part of a query. The `sybaseSQLQueryInSQLText` row objects contain the actual SQL text.

### Query syntax

Queries may contain any valid SQL statements.

The only restriction on queries is that the database user specified by the corresponding `sybaseSQLQueryControlUsername` object must have sufficient database privileges to execute all statements contained in the query, otherwise the query will return an error.

**Attention:** Exercise caution when creating queries because SQL statements such as drop may result in the loss of important data if used inappropriately.

### Long queries

The maximum length of SQL statements permitted in a `sybaseSQLQueryInSQLText` object is 255 characters; however, you may break the query into multiple SQL fragments spanning a set of input table rows. When executing the SQL query, the subagent reconstructs the query by concatenating the set of input rows in order of increasing row index and inserting a space character between each SQL fragment.

### About this task

For example, if the set of input table rows associated with a control row contains the following SQL fragments:

```
select name, schemacnt from sysobjects
where type in ('F', 'P', 'R',
'RI', 'U', 'XP')
```

the subagent would generate the SQL query:

```
select name, schemacnt from sysobjects where type in ('F', 'P', 'R', 'RI',
'U', 'XP')
```

You may break the SQL text at any position, provided that the break does not occur within a name or command word. You may also break the text between parameters in a function call and within comments.

### Temporary tables

If a query creates temporary tables, it must explicitly drop those tables when the query is complete.

Drop temporary table by appending the following command to the query:

```
drop table temp_table_name
```

Explicitly dropping temporary tables is necessary because the subagent uses connection pooling to minimize the number of connections made to the Sybase server. Connection pooling reuses existing sessions but has the side-effect that temporary tables are not cleared until the session eventually closes. Failure to explicitly drop temporary tables may lead to an accumulation of tables over the duration of the monitoring session.

## Query results

The results table (`sybaseSQLQueryResultsTable`) stores all data values returned by SQL queries, one row per value. If a query returns more than one value for a database column, the subagent stores each of the values in a separate row.

The result table also stores the text of each query, which is always located in the last result table row associated with the control row.

### Result formats

Result table rows contain objects for storing results of integer, string and date formats.

These objects are:

- `sybaseSQLQueryResultsIntResult`
- `sybaseSQLQueryResultsStringResult`
- `sybaseSQLQueryResultsDateResult`

The subagent stores query results in the object of appropriate type and, if possible, converts the results to values appropriate for the other two result objects. For example, if a query returns the string "123", the subagent stores this string in `sybaseSQLQueryResultsStringResult` and the equivalent integer value 123 in `sybaseSQLQueryResultsIntResult`. If conversion is not possible, the other objects remain empty.

### Results for multiple columns

If a query retrieves results from more than one database column, the result table stores the results in the same order that the column names appear in the SQL query text. The index object `sybaseSQLQueryResultsColumnIndex` identifies individual column results.

For example, Table 354 lists index values applied to the column names that would be returned by the following query:

```
select ps.SPID, ps.CpuTime, pst.LineNumber from ...
```

*Table 354. Indexing results for multiple columns*

Column name	<code>sybaseSQLQueryResultsColumnIndex</code>
<code>ps.CpuTime</code>	2
<code>ps.SPID</code>	1
<code>pst.LineNumber</code>	3

**Tip:** When using configuration commands to execute SQL queries, create a variable for each result returned by the query and set it to the value of the `sybaseSQLQueryResultsColumnIndex` that points to the result. This simplifies access to query results in subsequent commands.



## Threshold monitoring and event generation

To monitor thresholds and generate events using performance statistics extracted with SQL queries, use the genalarm subagent.

**Tip:** The genalarm subagent only performs threshold monitoring on integer values. When using genalarm to monitor metrics returned by SQL queries, convert all monitored metrics to integer values with the SQL convert() function.

### Monitoring table growth

Monitor the row count per table on the Sybase server SYBASE1 at one-minute intervals and generate an alarm if a table shows a growth rate greater than 10. Use the following SQL query to obtain the row count data:

```
select
  TableName = object_name(id),
  RowCnt = convert (int, rowcnt),
  DBName = db_name ()
from systabstats
where indid = 0
```

The configuration commands required for implementing this monitor are:

```
subagent load rmonc

#Create the warning event
event type=snmp-trap
event description="Sybase Warning"
event create
_sybaseWarnEvtIdx=$?

subagent load sybase

# Configure the control rowsybasesql reset
sybasesql server="SYBASE1"
sybasesql username="SybaseUser"
sybasesql password@="encrypted_password"
sybasesql interval=60

# Construct the query
sybasesqlin reset
sybasesqlin store sqltext="select TableName = object_name(id),"
sybasesqlin store sqltext="RowCnt = convert (int, rowcnt),"
sybasesqlin store sqltext="DBName = db_name () from systabstats
  where indid = 0"

# Store the column indexes for later reference
tabNameCol = 1
rowCntCol = 2
DBNameCol = 3

# Create the control row
sybasesql create
sqlIndex = $?

subagent load genalarm
# Create the delta-value monitor for the row count value
genalarmvb reset
genalarmvb store oid="$sybaseSQLQueryResultsStringResult.$sqlIndex.
  $tabNameCol.*"
genalarmvb store oid="$sybaseSQLQueryResultsStringResult.$sqlIndex.
  $DBNameCol.*"
genalarm reset
genalarm var="$sybaseSQLQueryResultsIntResult.$sqlIndex.$rowCntCol.*"
genalarm vardescr="Sybase: Row Count Delta"
```

```

genalarm interval=60
genalarm type=delta
genalarm mode=risingContinuous
genalarm risethresh=10
genalarm riseduration=0
genalarm riseevent=_sybaseWarnEvtIdx
genalarm riseverity=warning
genalarm risedescr="Table $$11 (DB $$12) has a high rate of row growth
($$6 last sample interval)"
genalarm create

```

For a list of configuration commands available for the SQL Query group, see “SQL query group” on page 473.

## Monitoring database free space

Monitor the free space on the Sybase server SYBASE1 at five-minute intervals and generate an alarm if the free space falls below 50%. Use the following SQL query to obtain and calculate the free space as a percentage value:

```

begin
  select  size = sum(size),
          free = sum(curunreservedpgs(dbid,lstart,unreservedpgs)),
          segmap,
          dbid
  into #free1
  from master..sysusages
  group by dbid, segmap

  select  size = sum(size),
          free = sum(free),
          dbid
  into #free2
  from #free1
  where segmap & 3 = 3
  group by dbid

  select  DB_Name = db_name(dbid),
          Free_KBytes=free*2,
          Total_KBytes=size*2,
          Used_KBytes=(size-free) * 2,
          Used_KBytes=(size-free) * 2,
          Percent_Free = convert(int, (convert(float,free)/size * 100.0))
  from #free2
  order by db_name(dbid)
end

```

The configuration commands required for implementing this monitor are:

```

subagent load rmonc

#Create the warning event
event type=snmp-trap
event description="Sybase Warning"
event create

_sybaseWarnEvtIdx=?
event description="Sybase Information"
event create
_sybaseInfoEvtIdx=?

subagent load sybase

# Configure the control rowsybasesql reset
sybasesql server="SYBASE1"
sybasesql username="SybaseUser"

```

```

sybaseql password@="encrypted_password"
sybaseql interval=60

# Construct the query
sybaseqlin reset
sybaseqlin store sqltext="begin"
sybaseqlin store sqltext="select size = sum(size),"
sybaseqlin store sqltext="free = sum(curunreservedpgs(dbid,lstart,unreservedpgs)),"
sybaseqlin store sqltext="segmap, dbid into #free1 from master..sysusages"
sybaseqlin store sqltext="group by dbid, segmap"
sybaseqlin store sqltext="select size = sum(size),free = sum(free),dbid"
sybaseqlin store sqltext="into #free2 from #free1 where segmap & 3 = 3 group by dbid"
sybaseqlin store sqltext="select DB_Name = db_name(dbid),"
sybaseqlin store sqltext="Free_KBytes=free*2, Total_KBytes=size*2,"
sybaseqlin store sqltext="Used_KBytes=(size-free) * 2,"
sybaseqlin store sqltext="Percent Free = convert (int, "
sybaseqlin store sqltext="(convert(float,free)/size * 100.0))"
sybaseqlin store sqltext="from #free2 order by db_name(dbid)"
sybaseqlin store sqltext="drop table #free1 drop table #free2"
sybaseqlin store sqltext="end"

# Store the column indexes for later reference
dbNameCol = 1
freeCol = 2
percentCol = 5

# Create the control row
sybaseql create
sqlIndex = $?

subagent load genalarm

# Create the variable bindings for notifications
genalarmvb reset
genalarmvb store oid="$sybaseSQLQueryResultsStringResult.$sqlIndex.$dbNameCol.*"
genalarmvb store oid="$sybaseSQLQueryResultsIntResult.$sqlIndex.$freeCol.*"

# Create the delta-value monitor for the row count value
genalarm var="$sybaseSQLQueryResultsIntResult.$sqlIndex.$percentCol.*"
genalarm vardescr="Sybase: Database Space Percentage Free"
genalarm interval=60
genalarm type=absolute
genalarm mode=fallingContinuous
genalarm fallthresh=50
genalarm fallduration=0
genalarm fallevent=$_sybaseWarnEvtIdx
genalarm fallseverity=severe
genalarm falldescr="Database $$11: Free space has dropped below $$7%
(Currently $$6% - $$12 Kb free)"
genalarm risethresh=50
genalarm riseduration=0
genalarm riseevent=$_sybaseInfoEvtIdx
genalarm riseseverity=info
genalarm risedescr="Database $$11: Free space has risen to $$6%"
genalarm create

```

For a list of configuration commands available for the SQL Query group, see “SQL query group” on page 473.

## Executing stored procedures

The SQL Query group enables you to execute Sybase stored procedures.

### Procedure

To execute a stored procedure:

1. Create a query control row and set `sybaseSQLQueryInTableRowCount` to 1.
2. Set `sybaseSQLQueryInSQLText` to the name of the stored procedure.

### Results

**Note:** Executing a stored procedure only returns data in the results table if the stored procedure itself returns a table. If the procedure simply prints output to the command line, the results table will not contain any data.

### Executing the `sp_lock` stored procedure

Execute the `sp_lock` stored procedure every five minutes on the Sybase server SYBASE1.

```
# Configure the control rowsybasesql reset
sybasesql server="SYBASE1"
sybasesql username="SybaseUser"
sybasesql password@="encrypted_password"
sybasesql interval=300

# Construct the query
sybasesqlin reset
sybasesqlin store sqltext="sp_lock"

# Create the control row
sybasesql create
sqlIndex = $?
```

---

## Extracting `sp_sysmon` report data

Use the Sysmon group (`sybaseSysMonGroup`) to extract performance data from reports produced by the Sybase `sp_sysmon` stored procedure.

You can use the sybase subagent to run `sp_sysmon` on the target Sybase server, then extracts specific data from the report it generates.

Figure 69 on page 467 shows an excerpt from an `sp_sysmon` report and identifies the report elements used by the subagent to extract performance data.

Sybase Adaptive Server Enterprise System Performance Report				
Server Version: Adaptive Server Enterprise/12.5.2/EBF 12123/P/Linux Inte				
Server Name: SYBASE_SERVER				
Run Date: Feb 10, 2005				
Statistics Cleared at: Feb 10, 2005 11:23:00				
Statistics Sampled at: Feb 10, 2005 11:23:30				
Sample Interval: 00:00:30				
Kernel Utilization				
Your Runnable Process Search Count is set to 2000 and I/O Polling Process Count is set to 10				
Performance Metrics	Column Headings			
Engine Busy Utilization	CPU Busy	I/O Busy	Idle	
Engine 0	0.0 %	0.0 %	100.0 %	
CPU Yields by Engine	per sec	per xact	count	% of total
Engine 0	49.9	499.0	499	100.0 %
Network Checks				
Non-Blocking	599.4	5994.0	5994	92.3 %
Blocking	49.9	499.0	499	7.7 %
Total Network I/O Checks	649.3	6493.0	6493	
Avg Net I/Os per Check	n/a	n/a	0.00000	n/a
Disk I/O Checks				
Total Disk I/O Checks	649.6	6496.0	6496	n/a
Checks Returning I/O	0.0	0.0	0	0.0 %

Figure 69. Excerpt from an `sp_sysmon` report

## Interaction between `sp_sysmon` and MDA tables

Running `sp_sysmon` has the side-effect of clearing certain MDA performance counters. Never use the `sybaseSysMonGroup` and other MDA table monitors, such as the `sybaseMDAGroup`, simultaneously unless you are certain that resetting MDA counters to zero will not adversely affect those monitors.

To obtain a list of the MDA counters that are reset by `sp_sysmon`, use the `isql` utility to execute the command:

```
select TableName, ColumnName
from monTableColumns
where Indicators & 2=2
```

## General procedure

Use the tables in the `sybaseSysMonGroup` to run `sp_sysmon` and extract the report data.

### Procedure

The general procedure for extracting data from `sp_sysmon` reports is:

1. Create a row in the header table (`sybaseSysMonHeaderTable`) for each set of column headings containing the data values that you wish to extract.

To create each header row:

- a. Set `sybaseSysMonHeaderStatus` to `createAndWait(5)`.
- b. Set `sybaseSysMonHeaderFormat` to a comma-separated list of report column headings.
- c. Activate the row by setting `sybaseSysMonHeaderStatus` to `active(1)`.

2. Create a row in the control table (sybaseSysMonControlTable) identifying the Sybase server that you wish to monitor:
  - a. Set sybaseSysMonControlStatus to createAndWait(5).
  - b. Set sybaseSysMonControlServerName to the name of the Sybase server.
  - c. Set the sybaseSysMonControlUsername and sybaseSysMonControlPassword objects to the credentials of a database user with user with sa\_role privileges.  
For information about database user privileges, see “Granting monitoring privileges” on page 457.
  - d. Set sybaseSysMonControlSampleInterval to the interval (in seconds) at which you wish the performance data to be updated.
  - e. Set sybaseSysMonDuration to the duration (in seconds) that you wish to run sp\_sysmon on the monitored Sybase server.
  - f. Set sybaseSysMonMatchRows to the number of performance metrics that you wish to extract from the report.
3. Create rows in the match table (sybaseSysMonMatchTable) for each performance statistic extracted.  
In each match table row, set sybaseSysMonMatchExpression to a regular expression representing the name of the performance metric that you wish to extract.
4. Activate the control row by setting sybaseSysMonControlStatus to active(1).  
The sybase subagent commences monitoring and updates data in the sybaseSysMonDataTable at the interval set by sybaseSysMonControlSampleInterval.

## Identifying performance metrics

The sybaseSysMonMatchExpression object identifies the name of the performance metric to be extracted from the sp\_sysmon report. To identify the performance metric, construct a regular expression from the name of the report section and any intermediate subsections containing the metric, and the name of the performance metric itself.

For example, to match the first Total Requests metric in the report excerpt below, use the regular expression:

Worker Process Management.\*Worker ProcessRequests.\*Total Requests

### Excerpt from an sp\_sysmon report

```
=====
```

Worker Process Management				
-----				
	per sec	per xact	count	% of total
	-----	-----	-----	-----
Worker Process Requests				
Total Requests	0.0	0.0	0	n/a
Worker Process Usage				
Total Used	0.0	0.0	0	n/a
Max Ever Used During Sample	0.0	0.0	0	n/a
Memory Requests for Worker Processes				
Total Requests	0.0	0.0	0	n/a

```
=====
```

The sybase subagent evaluates regular expressions in a non-greedy manner, returning only one matching row. That is, it matches at most one performance metric, and this metric is always the *last* match in the report. For example, a regular expression matching the statistic Total Requests (located on the last line of the report excerpt is:

```
Worker Process Management.*Total.*
```

For information about regular expression syntax, see the *Netcool/SSM Administration Guide*.

## Generated metrics

The number of data values extracted for each performance metric depends on the number of report column headings listed in the `sybaseSysMonHeaderFormat` object.

For example, if the report contains the column headings `per sec`, `per xact`, `count`, each match on a performance statistic creates three rows in `sybaseSysMonDataTable`. The first row contains the value extracted from the `per sec` report column, the second row contains the value from the `per xact` column, and the third contains the value from `count`.

Data row indexing for the results follows a left-to-right convention: the first data row in `sybaseSysMonDataTable` corresponds to the first column heading in the list, working from left to right. Figure 70 on page 470 illustrates the relationship.

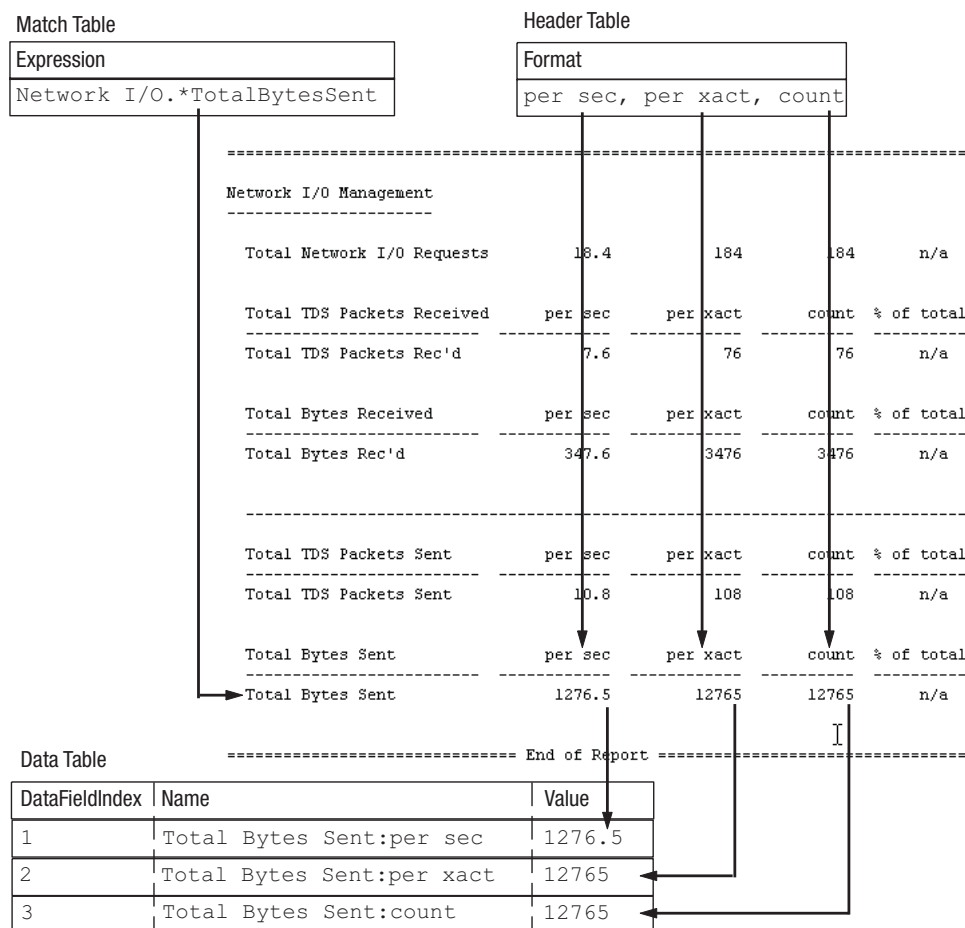


Figure 70. Relationship between report column headings and `sybaseSysMonDataTable`

The mapping between items listed in the `sybaseSysMonHeaderFormat` object and report column headings is fixed. The first item listed always matches the first report column heading, the second item matches the second report column and so on. To list any report column heading, you must also list all column headings to the left of that column in the order that they appear in the report. For example, to extract count metrics from the report shown in Figure 70, the `sybaseSysMonHeaderFormat` object must also list the per sec and per xact column names: per sec, per xact, count.

## Accessing data values

The data extracted from reports is contained in `sybaseSysMonDataValue` objects.

Access these objects using a combination of index values:

`sybaseSysMonDataValue.control_row_index.match_row_index.col_head_num`

where `col_head_num` is determined from the heading's position in the `sybaseSysMonHeaderFormat` string, as shown in Figure 70.



## Threshold monitoring and event generation

To monitor thresholds and generate events using performance statistics extracted from `sp_sysmon` reports, use the `genalarm` subagent.

### Monitoring index scans

This example shows how to use the Sysmon group to monitor the Total Scans performance metric located in the Index Management section of an `sp_sysmon` report. The example also uses the `genalarm` subagent to generate an alarm when the total number of index scans performed by the Sybase server exceeds five per second.

The Total Scans metric is the last entry of the sample report shown below.

```
=====
```

Index Management				
-----				
Nonclustered Maintenance	per sec	per xact	count	% of total
-----				
Ins/Upd Requiring Maint	0.0	0.0	0	n/a
# of NC Ndx Maint	0.0	0.0	0	n/a
Deletes Requiring Maint	0.0	0.0	0	n/a
# of NC Ndx Maint	0.0	0.0	0	n/a
RID Upd from Clust Split	0.0	0.0	0	n/a
# of NC Ndx Maint	0.0	0.0	0	n/a
Upd/Del DOL Req Maint	0.0	0.0	0	n/a
# of DOL Ndx Maint	0.0	0.0	0	n/a
Page Splits	0.0	0.0	0	n/a
Page Shrinks	0.0	0.0	0	n/a
Index Scans	per sec	per xact	count	% of total
-----				
Ascending Scans	0.0	0.0	0	0.0 %
DOL Ascending Scans	0.0	0.0	0	0.0 %
Descending Scans	0.0	0.0	0	0.0 %
DOL Descending Scans	0.0	0.0	0	0.0 %
-----				
Total Scans	0.0	0.0	0	
=====				

The configuration commands required for implementing this monitor are:

```
#Create the events
subagent load rmonc
event type=snmp-trap
event description="Sybase Warning"
event create
_sybaseWarnEvtIdx=$?
event description="Sybase Information"
event create
_sybaseInfoEvtIdx=$?

subagent load sybase

#Define the report column heading format
sysmonheader format="per sec"
sysmonheader create
stdHeaderIdx=$?
```

```

#Identify the "Total Scans" performance metric
sysmonmatch expr="Index Management.*Index Scans.*Total Scans"
sysmonmatch header=$stdHeaderIdx
sysmonmatch store
totalScansIdx=$?

#Create the control row for monitoring the server "SybaseServer"
sysmon servername="MySybaseServer"
sysmon username="SybaseUser"
sysmon password@="encrypted_password"
sysmon sample=60
sysmon duration=30
sysmon create
ctrlIdx=$?

#Configure the alarm
subagent load genalarm

genalarm reset
genalarm var="$sybaseSysMonDataValue.$ctrlIdx.$totalScansIdx.1"
genalarm vardescr="Database Scans"
genalarm interval=60
genalarm mode=singleEdge
genalarm risethresh=5
genalarm riseseverity=critical
genalarm risedescr="Total scans per second is severe ($$6)"
genalarm riseevent=$_sybaseWarnEvtIdx
genalarm fallthresh=5
genalarm fallseverity=info
genalarm falldescr="Total scans per second is nominal ($$6)"
genalarm fallevent=$_sybaseInfoEvtIdx
genalarm create

```

For a list of configuration commands available for the Sysmon group, see “Sysmon group” on page 474.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### MDA group

The sybasemda commands create rows in the MDA control table (sybaseMDAControlTable).

The general syntax of these commands is:

```

sybasemda property=value ...
sybasemda create [property=value ...]
sybasemda reset

```

Table 355 on page 473 lists the properties supported in the commands.

Table 355. Configuration command properties - sybasemda

Property	Type	Description	Sets MIB object
datacontrol	enum	Controls the monitoring status of the control row:  on - Data collection enabled  off - Data collection disabled	DataControl
interval	int	Sets the interval (in seconds) at which data is retrieved from the monitored Sybase server.	SampleInterval
password	string	Sets the password used when logging on to the monitored Sybase server.	Password
servername	string	Specifies the name of the Sybase server to be monitored.	ServerName
username	string	Sets the username supplied when logging on to the monitored Sybase server.	Username

## SQL query group

The configuration commands for the SQL Query group create rows in the group's control and input tables.

### Control table

The sybasesql commands create rows in the SQL Query control table (sybaseSQLQueryControlTable).

The general syntax of these commands is:

```
sybasesql property=value ...
sybasesql create [property=value ...]
sybasesql reset
```

Table 356 lists the properties supported in the commands.

Table 356. Configuration command properties - sybasesql

Property	Type	Description	Sets MIB object
datacontrol	enum	Controls the monitoring status of the control row:  on - Data collection enabled  off - Data collection disabled	DataControl
interval	int	Sets the interval (in seconds) at which data is retrieved from the monitored Sybase server.	SampleInterval
password	string	Sets the password used when logging on to the monitored Sybase server.	Password
servername	string	Specifies the name of the Sybase server to be monitored.	ServerName
username	string	Sets the username supplied when logging on to the monitored Sybase server.	Username

## Input table

The sybasesqlin commands create rows in the SQL Query input table (sybaseSQLQueryInTable) and associate them with the next row created in the SQL Query control table using the sybasesql command.

The general syntax of these commands is:

```
sybasesqlin property=value ...  
sybasesqlin store  
sybasesqlin reset
```

Table 357 lists the properties supported in the commands.

*Table 357. Configuration command properties - sybasesqlin*

Property	Type	Description	Sets MIB object
text	string	The text of the SQL query to run. The query may span multiple lines; each line is limited to 255 characters in length.	QueryInSQLText

## Sysmon group

The configuration commands for the Sysmon group create rows in the group's control, header, and match tables.

### Control table

The sysmon commands create rows in the sysmon control table (sybaseSysmonControlTable).

The general syntax of these commands is:

```
sysmon property=value ...  
sysmon create [property=value ...]  
sysmon reset
```

Table 358 lists the properties supported in the commands.

*Table 358. Configuration command properties - sysmon*

Property	Type	Description	Sets MIB object
datacontrol	enum	Controls the monitoring status of the control row:  on - Data collection enabled  off - Data collection disabled	DataControl
duration	int	Sets the sample duration (in seconds).	Password
password	string	Sets the password used when logging on to the monitored Sybase server.	Password
sample	int	Sets the interval (in seconds) at which data is retrieved from the monitored Sybase server.	SampleInterval
servername	string	Specifies the name of the Sybase server to be monitored.	ServerName
username	string	Sets the username supplied when logging on to the monitored Sybase server.	Username

## Header table

The sysmonheader commands create rows in the sysmon header table (sybaseSysMonHeaderTable).

The general syntax of these commands is:

```
sysmonheader property=value ...  
sysmonheader create [property=value ...]  
sysmonheader reset
```

Table 359 lists the properties supported in the commands.

*Table 359. Configuration command properties - sysmonheader*

Property	Type	Description	Sets MIB object
format	string	A description of the header.	Description
name	string	The name of the header.	Name

## Match table

The sysmonmatch commands create rows in the sysmon match table (sybaseSysMonMatchTable) and associate them with the next row created in the sysmon control table created using the sysmon command.

The general syntax of these commands is:

```
sysmonmatch property=value ...  
sysmonmatch store  
sysmonmatch reset
```

Table 360 lists the properties supported in the commands.

*Table 360. Configuration command properties - sysmonmatch*

Property	Type	Description	Sets MIB object
description	string	A description of the match expression.	Description
expression	string	A regular expression specifying the required report data.	Expression
header	string	The row index of the corresponding entry in sybaseSysMonHeaderTable.	HeaderIndex

---

## MIB module

The MIB provides a standard SNMP interface, and contains groups that correspond to the methods of monitoring Sybase ASE servers.

The groups in the MIB module are:

- MDA group, sybaseMDAGroup
- SQL Query group, sybaseSQLQueryGroup
- Sysmon group, sybaseSysMonGroup

## MIB structure

The sybase MIB module is a subtree of networkharmoni (1977).

The subtree is shown in Figure 71.

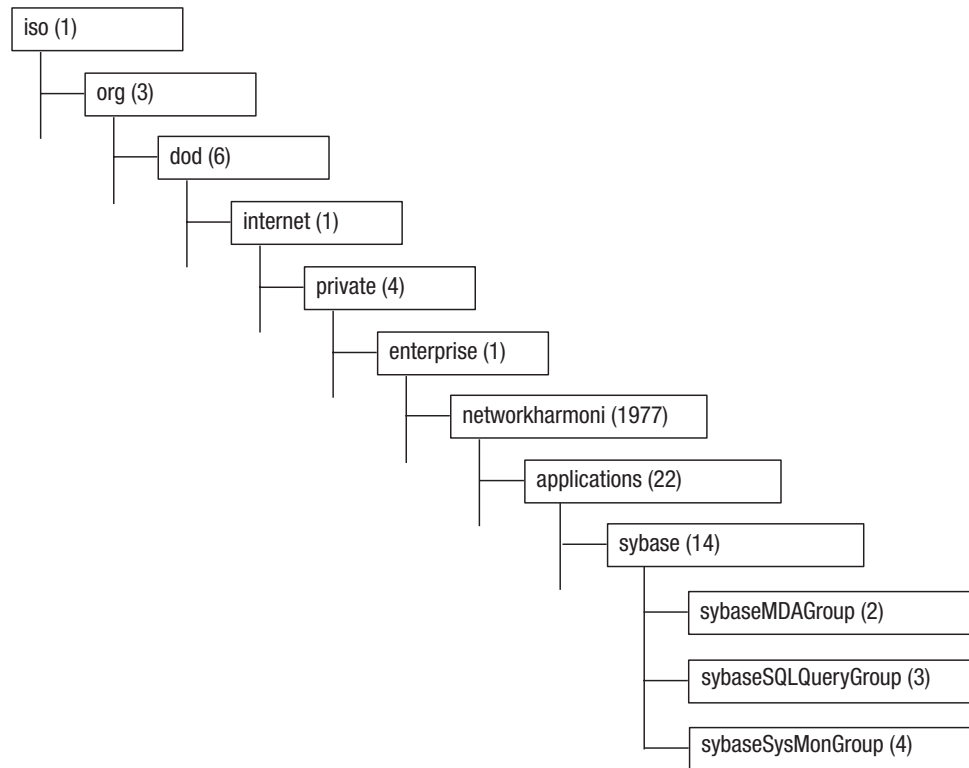


Figure 71. OID tree diagram of the Netcool/ASM for Sybase MIB module

This section provides a summary of the objects defined in the sybase MIB module. For detailed information on all objects in the module, see the sybase-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MDA group

The MDA group (sybaseMDAGroup) provides facilities for retrieving data from the Sybase internal MDA performance tables.

### Control table

The control table (sybaseMDAControlTable) provides objects for configuring monitoring using the Sybase MDA tables.

Table 361 lists the objects in sybaseMDAControlTable.

Table 361. sybaseMDAControlTable row objects

Row object	Description
DataControl	Controls the monitoring status of this control row:  on(1) - Enables data collection off(2) - Disables data collection
Index	Uniquely identifies the row.

Table 361. *sybaseMDAControlTable* row objects (continued)

Row object	Description
LastUpdate	The value of sysUpTime when the row last updated its associated data rows.
MonitorStatus	The current sampling status of the control row:  idle(1) - The control row is in an idle state  sampling(2) - The control row is currently sampling data from the server  loginError(3) - A login error occurred. This usually indicates that the value of one or more of the sybaseMDAControlUsername, sybaseMDAControlPassword and sybaseMDAControlServerName objects are invalid.  systemError(4) - An internal system error occurred, for example the SQL statement could not be executed  noDataError(5) - The subagent was unable to locate the Sybase table monStateTable  unknownError(6) - An error of undetermined nature occurred
Owner	Defines the owner/creator of the control row.
Password	Sets the password used when logging on to the monitored Sybase server.
SampleInterval	Sets the interval (in seconds) at which data is retrieved from the monitored Sybase server. If the value of this object is 0, only one sample is performed.
ServerName	Specifies the name of the Sybase server to be monitored.
Status	SNMPv2 Row Status. Controls creation, activation and deletion of the control row.
TimeForLastUpdate	The time (in seconds) taken to perform the last update.
Username	Sets the username supplied when logging on to the monitored Sybase server.

## State table

The state table (sybaseMDAStateTable) provides data about the state of the Sybase ASE server.

Table 362 lists the objects in sybaseMDAStateTable.

Table 362. *sybaseMDAStateTable* row objects

Row object	Description
CheckPoints	Reports whether any checkpoint is currently running.
Connections	Reports whether sybmon is performing a shared memory dump.
CountersCleared	Date and time that the monitor counters were last cleared.
DaysRunning	Number of days that the Sybase ASE server has been running.
DiagnosticDumps	Reports whether sybmon is performing a shared memory dump.
LockWaits	Number of processes that have waited longer than the lock wait threshold.

Table 362. *sybaseMDAStateTable* row objects (continued)

Row object	Description
LockWaitThreshold	Time (in seconds) that processes must wait for locks in order to be reported.
MaxRecovery	The maximum time (in minutes), per database, that the Sybase ASE server uses to complete its recovery procedures in case of a system failure it contains the current Run Value for recovery interval (in minutes).
NumDeadlocks	Total number of deadlocks that have occurred.
StartDate	Date and time that the Sybase ASE server was started.

## Engine table

The engine table (*sybaseMDAEngineTable*) provides statistics about Sybase ASE engines.

Table 363 lists the objects in *sybaseMDAEngineTable*.

Table 363. *sybaseMDAEngineTable* row objects

Row object	Description
AffinitiedToCPU	The CPU number for which the engine has affinity.
Connections	Number of connections handled.
ContextSwitches	Number of context switches.
CPUBusyPercent	The CPU utilization, expressed as a percentage value, calculated using:  $100 * (\text{sybaseMDAEngineUserCPUTime} + \text{sybaseMDAEngineSystemCPUTime}) / \text{sybaseMDAEngineCPUTime}$
CPUTime	Total time (in seconds) that the engine has been running.
CurrentKPID	Kernel process identifier for the currently executing process.
EngineIndex	Index for the Engine Table.
EngineNumber	Number of the Sybase ASE engine.
IdleCPUTime	Time (in seconds) that the engine has been in idle spin mode.
PreviousKPID	Kernel process identifier for the previously executing process.
ProcessesAffinitied	Number of processes that have affinity for the engine.
StartTime	Date that the engine came online.
Status	Status of the engine (for example, online or offline).
StopTime	Date that the engine went offline.
SystemCPUTime	Time (in seconds) that the engine has been executing system database services.
UserCPUTime	Time (in seconds) that the engine has been executing user commands.



## Data cache table

The data cache table (sybaseMDADataCacheTable) provides statistics relating to Sybase ASE data caches.

Table 364 lists the objects in sybaseMDACacheTable.

*Table 364. sybaseMDADataCacheTable row objects*

Row object	Description
BufferPools	The number of buffer pools within the cache.
CacheID	Unique identifier for the cache.
CacheIndex	Unique identifier for this cache table row.
CacheName	Name of the cache.
CachePartitions	Number of partitions currently configured for the cache.
CacheSearches	Cache searches directed to the cache.
LogicalReads	Number of buffers retrieved from the cache.
PhysicalReads	Number of buffers read into the cache from disk.
PhysicalWrites	Number of buffers written from the cache to disk.
RelaxedReplacement	Indicates whether the cache is using a relaxed cache replacement strategy.
Stalls	Number of 'dirty' buffer retrievals.

## Procedure cache table

The procedure cache table (sybaseMDAProcedureCacheTable) provides statistics relating to Sybase ASE procedure cache.

Table 365 lists the objects in sybaseMDAProcedureCacheTable.

*Table 365. sybaseMDAProcedureCacheTable row objects*

Row object	Description
CacheLoads	Number of stored procedures loaded into the cache.
CacheRequests	Number of stored procedures requested.
CacheStalls	Number of times a process had to wait for a free procedure cache buffer when installing a stored procedure into the cache.
CacheWrites	Number of times a procedure was normalized and the tree was written back to sysprocedures.

## Open databases table

The open databases table (sybaseMDAOpenDatabasesTable) provides state and statistical information about databases that are currently in use.

Table 366 lists the objects in sybaseMDAOpenDatabasesTable.

*Table 366. sybaseMDAOpenDatabasesTable row objects*

Row object	Description
AppendLogRequests	Number of semaphore requests when attempting to append to the database transaction log.
AppendLogWaits	Number of times a task had to wait for the append log semaphore to be granted.

Table 366. *sybaseMDAOpenDatabasesTable* row objects (continued)

Row object	Description
BackupInProgress	Whether a backup is currently in progress for the database.
BackupStartTime	Date that the last backup started for the database.
DatabaseName	Name of the database.
DBID	Uniquely identifies the database.
Index	Uniquely identifies this row.
LastBackupFailed	Whether the last backup of the database failed.
TransactionLogFull	Whether the database transaction log is full.

## Worker thread table

The worker thread table (*sybaseMDASysWorkerThreadTable*) provides statistics about the worker processes running on the Sybase ASE.

Table 367 lists the objects in *sybaseMDASysWorkerThreadTable*.

Table 367. *sybaseMDASysWorkerThreadTable* row objects

Row object	Description
HighWater	The maximum number of worker processes that have ever been in use.
MaxParallelDegree	The maximum degree of parallelism that can be used (the current Run Value for max parallel degree configuration parameter).
MaxScanParallelDegree	The maximum degree of parallelism that can be used for a scan (the current Run Value for max scan parallel degree configuration parameter).
ParallelQueries	Number of parallel queries that were attempted.
PlansAltered	Number of plans altered due to worker processes not being available.
ThreadsActive	Number of worker processes active.
TotalWorkerMemory	The amount of memory configured for use by worker processes.
TotalWorkerThreads	Configured maximum number of worker processes.
WorkerMemory	The amount of memory currently in use by worker processes.
WorkerMemoryHWM	The maximum amount of memory ever used by worker processes.

## Network I/O table

The network I/O table (*sybaseMDANetworkIOTable*) provides statistics about network input and output.

Table 368 lists the objects in *sybaseMDANetworkIOTable*.

Table 368. *sybaseMDANetworkIOTable* row objects

Row object	Description
BytesReceived	Number of bytes received.
BytesSent	Number of bytes sent.

Table 368. *sybaseMDANetworkIOTable* row objects (continued)

Row object	Description
PacketsReceived	Number of packets received.
PacketsSent	Number of packets sent.

## Error log table

The error log table (*sybaseMDAErrorLogTable*) stores the most recent error messages from the Sybase ASE error log.

Table 369 lists the objects in *sybaseMDAErrorLogTable*.

Table 369. *sybaseMDAErrorLogTable* row objects

Row object	Description
EngineNumber	Engine on which the process was running.
ErrorMessage	Text of the error message.
ErrorNumber	Error message number.
FamilyID	SPID of the parent process.
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
Severity	Severity of error.
SPID	Session process identifier.
Time	Time stamp when error occurred.

## Locks table

The Locks table (*sybaseMDALocksTable*) lists all locks that are being held, and those that have been requested, by any process, for every object.

Table 370 lists the objects in *sybaseMDALocksTable*.

Table 370. *sybaseMDALocksTable* row objects

Row object	Description
Context	Lock context (bit field). These values are the same as those of the context column in <i>syslocks</i> .
DBID	Unique identifier for the database.
Index	Unique identifier for each entry in the MDA Locks table.
KPID	Kernel process identifier.
LockID	Lock object ID.
LockLevel	The type of object for which the lock was requested (for example, PAGE or ROW).
LockState	Indicates whether the lock has been granted:  Granted  Requested
LockType	The type of lock (for example, exclusive table, or shared page).
ObjectID	Unique identifier for the object.
PageNumber	Page that is locked when LockLevel has the value PAGE.

Table 370. *sybaseMDALocksTable* row objects (continued)

Row object	Description
ParentSPID	Parent process ID.
RowNumber	Row that is locked when LockLevel has the value ROW.
SPID	Session process identifier.
WaitTime	The time (in seconds) that the lock request has not been granted.

## Deadlock table

The deadlock table (*sybaseMDADeadLockTable*) provides data about the most recent deadlocks that have occurred in Sybase ASE.

Table 371 lists the objects in *sybaseMDADeadLockTable*.

Table 371. *sybaseMDADeadLockTable* row objects

Row object	Description
DeadLockID	Unique identifier for the deadlock.
DeadLockIndex	Unique identifier for this row.
HeldAppName	Name of the application holding the lock.
HeldBatchID	Unique batch identifier for the SQL code being executed by the process holding the lock when it was blocked by another process (not when it acquired the lock).
HeldCommand	Command being executed that caused the lock to be held.
HeldContextID	Unique context identifier for the process holding the lock when it was blocked by another process (not when it acquired the lock).
HeldFamilyId	SPID of the parent process of the process holding the lock.
HeldKPID	KPID of the process holding the lock.
HeldLineNumber	Line number within the batch of the statement being executed by the process holding the lock when it was blocked by another process (not when it acquired the lock).
HeldLockType	Type of lock being held.
HeldProcDBID	Unique identifier for the database where the stored procedure that caused the lock to be held resides, if applicable.
HeldProcedureID	Unique object identifier for the stored procedure that caused the lock to be held, if applicable.
HeldSPID	SPID of the process holding the lock.
HeldTranName	Name of the transaction in which the lock was acquired.
HeldUserName	Name of the user for whom the lock is being held.
ObjectDBID	Unique database identifier for the database where the object resides.
ObjectName	Name of the object.
PageNumber	Page number for which the lock was requested, if applicable.
ResolveTime	Time at which the deadlock was resolved.
RowNumber	Row number for which the lock was requested, if applicable.
VictimKPID	KPID of the victim process for the deadlock.
WaitFamilyId	SPID of the parent process of the process waiting for the lock.
WaitKPID	KPID of the process waiting for the lock.

Table 371. *sybaseMDADeadLockTable* row objects (continued)

Row object	Description
WaitLockType	Type of lock requested.
WaitSPID	SPID of the process waiting for the lock.
WaitTime	Amount of time in milliseconds that the waiting process was blocked before the deadlock was resolved.
WaitUserName	Name of the user for whom the lock is being requested.

## Wait class information table

The wait class information table (*sybaseMDAWaitClassInfoTable*) provides a description of each wait class (for example, waiting for a disk read to complete). Wait classes group wait events according the type of event that a process is waiting for.

Table 372 list the objects in *sybaseMDAWaitClassInfoTable*.

Table 372. *sybaseMDAWaitClassInfoTable* row objects

Row object	Description
Description	Description of the wait event class.
WaitClassID	Unique identifier for the wait event class.
WaitClassIndex	Uniquely identifies the row.

## Wait event information table

The wait event information table (*sybaseMDAWaitEventInfoTable*) provides a description of every possible situation in which a process is forced to wait within Sybase ASE, such as waiting for a buffer read to complete.

Table 373 list the objects in *sybaseMDAWaitEventInfoTable*.

Table 373. *sybaseMDAWaitEventInfoTable* row objects

Row object	Description
Description	Description of the wait event type.
Index	Uniquely identifies the row.
WaitClassID	Unique identifier for the wait event class.
WaitEventID	Unique identifier for the wait event type.

## Cached object table

The cached object table (*sybaseMDACachedObjectTable*) stores statistics for all objects and indexes with pages currently in a data cache.

Table 374 lists the objects in *sybaseMDACachedObjectTable*.

Table 374. *sybaseMDACachedObjectTable* row objects

Row object	Description
CachedKB	Number of kilobytes of the cache that the object is occupying.
CacheID	Unique identifier for the cache.
CacheName	Name of the cache.
DBID	Unique identifier for the database.

Table 374. *sybaseMDACachedObjectTable* row objects (continued)

Row object	Description
DBName	Name of the database.
Index	Uniquely identifies the row.
IndexID	Unique identifier for the index.
ObjectID	Unique identifier for the object.
ObjectName	Name of the object.
ObjectType	Object type.
OwnerName	Name of the object owner.
OwnerUserID	Unique identifier for the database owner.
ProcessesAccessing	Number of processes that are currently accessing the object.

## Cache pool table

The cache pool table (*sybaseMDACachePoolTable*) provides statistics about all pools allocated in all caches.

Table 375 lists the objects in *sybaseMDACachePoolTable*.

Table 375. *sybaseMDACachePoolTable* row objects

Row object	Description
AllocatedKB	Number of bytes that have been allocated for the pool.
BuffersToLRU	Number of buffers that were fetched and replaced in the least recently used portion of the pool.
BuffersToMRU	Number of buffers that were fetched and replaced in the most recently used portion of the pool.
CacheID	Unique identifier for the cache.
CacheName	Name of the cache.
Index	Uniquely identifies the row.
IOBufferSize	Size (in bytes) of the I/O buffer for the pool.
PagesRead	Number of pages read into the pool.
PagesTouched	Number of bytes that are currently being used within the pool.
PhysicalReads	Number of buffers that have been read from disk into the pool.
Stalls	Number of dirty buffer retrievals.

## Open object activity table

The open object activity table (*sybaseMDAOpenObjectActivityTable*) provides statistics for all open objects.

Table 376 lists the objects in *sybaseMDAOpenObjectActivityTable*.

Table 376. *sybaseMDAOpenObjectActivityTable* row objects

Row object	Description
APFReads	Number of APF buffers read.
DBID	Database ID.
Index	Uniquely identifies the row.
IndexID	Index ID.

Table 376. *sybaseMDAOpenObjectActivityTable* row objects (continued)

Row object	Description
LockRequests	Number of requests for a lock on the object.
LockWaits	Number of times a task waited for a lock for the object.
LogicalReads	Total number of buffers read.
ObjectID	Object ID.
Operations	Number of times that the object was accessed.
PagesRead	Total number of pages read.
PagesWritten	Total number of pages written to disk.
PhysicalReads	Number of buffers read from disk.
PhysicalWrites	Total number of buffers written to disk.
RowsDeleted	Number of rows deleted.
RowsInserted	Number of rows inserted.
RowsUpdated	Number of updates.

## I/O queue table

The I/O queue table (*sybaseMDAIOQueueTable*) provides device I/O statistics broken down into data and log I/O for normal and temporary databases on each device.

Table 377 lists the objects in *sybaseMDAIOQueueTable*.

Table 377. *sybaseMDAIOQueueTable* row objects

Row object	Description
Index	Uniquely identifies the row.
I/Os	Total number of I/O operations.
IOTime	Amount of time (in milliseconds) spent waiting for I/O requests to be satisfied.
IOType	Category for grouping I/O (user data, User log, Tempdb Data, or Tempdb log).
LogicalName	Logical name of the device.

## Device I/O table

The device I/O table (*sybaseMDADeviceIOTable*) provides statistical information about devices.

Table 378 lists the objects in *sybaseMDADeviceIOTable*.

Table 378. *sybaseMDADeviceIOTable* row objects

Row object	Description
APFReads	Number of APF reads from the device.
DevSemaphoreRequests	Number of I/O requests.
DevSemaphoreWaits	Number of tasks forced to wait for synchronization of an I/O request.
Index	Uniquely identifies the row.
IOTime	Total amount of time (in milliseconds) spent waiting for I/O requests to be satisfied.

Table 378. *sybaseMDADeviceIOTable* row objects (continued)

Row object	Description
LogicalName	Logical name of the device.
PhysicalName	Full hierarchic file name of the device.
Reads	Number of reads from the device (excluding APF).
Writes	Number of writes to the device.

## System waits table

The system waits table (*sybaseMDASysWaitsTable*) provides a server-wide view of processes that are waiting for an event.

Table 379 lists the objects in *sybaseMDASysWaitsTable*.

Table 379. *sybaseMDASysWaitsTable* row objects

Row object	Description
Index	Uniquely identifies the row.
WaitEventID	Unique identifier for the wait event.
Waits	Number of times tasks have waited for the event.
WaitTime	Amount of time (in milliseconds) that tasks have spent waiting for the event.

## Processes group

The processes group contains tables that provide information about Sybase ASE process. The tables contain information such as process activity and wait times, and SQL queries currently executing.

### Process table

The process table (*sybaseMDAProcessTable*) provides detailed statistics about processes that are currently executing or waiting. Table 380 lists the objects in *sybaseMDAProcessTable*.

Table 380. *sybaseMDAProcessTable* row objects

Row object	Description
Application	Application name.
BatchID	Unique identifier for the SQL batch containing the statement being executed.
BlockingSPID	If the process is waiting for a lock, this object holds the session process identifier of the process holding the lock that the process has requested.
BlockingXLOID	If the process is waiting for a lock, this object holds the unique lock identifier for the lock that this process has requested.
Command	Category of process or command that the process is currently executing.
ContextID	The stack frame of the procedure, if it exists.
DBID	Unique identifier for the database being used by the current process.



Table 380. *sybaseMDAProcessTable* row objects (continued)

Row object	Description
DBName	Name of process for the database being used by the current process.
EngineGroupName	Engine group for the process.
EngineNumber	Unique identifier of the engine on which the process is executing.
ExecutionClass	Execution class for the process.
FamilyID	The SPID of the parent process, if it exists.
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
LineNumber	Line number of the current statement within the SQL batch.
Login	Login user name.
MasterTransactionID	If the process is in a transaction, this object holds the unique transaction identifier for the current transaction.
NumChildren	Number of child processes, if executing a parallel query.
Priority	Priority at which the process is executing.
SecondsConnected	Number of seconds since this connection was established.
SecondsWaiting	Amount of time in seconds that the process has been waiting, if the process is currently in a wait state.
SPID	Session process identifier.
WaitEventID	if the process is currently in a wait state, this object holds the unique identifier for the event that the process is waiting for.

## Process lookup table

The process lookup table (*sybaseMDAProcessLookupTable*) provides information for tracking processes to an application, user, and client machine. Table 381 lists the objects in *sybaseMDAProcessLookupTable*.

Table 381. *sybaseMDAProcessLookupTable* row objects

Row object	Description
Application	Application name.
ClientHost	Host name of client.
ClientIP	IP address of client.
ClientOSPID	OS process identifier of the client application.
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
Login	Login user name.
SPID	Session process identifier.

## Process activity table

The process activity table (*sybaseMDAProcessActivityTable*) provides detailed statistics about process activity. Table 382 on page 488 lists the objects in *sybaseMDAProcessActivityTable*.

Table 382. *sybaseMDAProcessActivityTable* row objects

Row object	Description
Commits	Number of transactions committed by the process.
CPUTime	CPU time (in milliseconds) used by the process.
Index	Uniquely identifies the row.
IndexAccesses	Number of pages where data was retrieved using an index.
KPID	Kernel process identifier.
LocksHeld	Number of locks that the process currently holds.
LogicalReads	Number of buffers read from cache.
MemUsageKB	Amount of memory (in bytes) allocated to the process.
PagesRead	Number of pages read.
PagesWritten	Number of pages written.
PhysicalReads	Number of buffers read from disk.
PhysicalWrites	Number of buffers written to disk.
Rollbacks	Number of transactions rolled back by the process.
SPID	Session process identifier.
TableAccesses	Number of pages where data was retrieved without an index.
TempDbAccess	Number of temporary tables accessed.
Transactions	Number of transactions started by the process.
ULCBytesWritten	Number of bytes written to the user log cache for the process.
ULCCurrentUsage	The current usage (in bytes) of the User log cache by the process.
ULCFlushes	Total number of times that the user log cache was flushed.
ULCFlushFull	Number of times that the user log cache was flushed because it was full.
ULCMaxUsage	The maximum ever usage (in bytes) of the user log cache by the process.
WaitTime	Time (in milliseconds) the process has spent waiting.

## Process net I/O table

The process network I/O table (*sybaseMDAProcessNetIOTable*) provides network I/O activity information for each process. Table 383 lists the objects in *sybaseMDAProcessNetIOTable*.

Table 383. *sybaseMDAProcessNetIOTable* row objects

Row object	Description
BytesRecieved	Number of bytes received.
BytesSent	Number of bytes sent.
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
NetworkPacketSize	Network packet size the session is currently using.
PacketSent	Number of packets sent.
PacketsReceived	Number of packets received.
SPID	Session process identifier.

## Process object table

The process object table (sybaseMDAProcessObjectTable) provides statistics about objects that have been accessed by processes. Table 384 lists the objects in sybaseMDAProcessObjectTable.

*Table 384. sybaseMDAProcessObjectTable row objects*

Row object	Description
DBID	Unique identifier for the database where the object resides.
DBName	Name of the database.
Index	Uniquely identifies the row.
IndexID	Unique identifier for the index.
KPID	Kernel process identifier.
LogicalReads	Number of buffers read from the cache.
ObjectID	Unique identifier for the object.
ObjectName	Name of the object.
ObjectType	Type of object.
OwnerUserID	User identifier for the object owner.
PhysicalAPFReads	Number of APF buffers read from disk.
PhysicalReads	Number of buffers read from disk.
SPID	Session process identifier.

## Process waits table

The process waits table (sybaseMDAProcessWaitsTable) provides a server-wide view of the processes waiting for an event. Table 385 lists the objects in sybaseMDAProcessWaitsTable.

*Table 385. sybaseMDAProcessWaitsTable row objects*

Row object	Description
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
SPID	Session process identifier.
WaitEventID	Unique identifier for the wait event.
Waits	Number of times the process has waited for the event.
WaitTime	Amount of time (in milliseconds) that the process has waited for the event.

## Process statement table

The process statement table (sybaseMDAProcessStatementTable) provides data about statements that are currently executing. Table 386 lists the objects in sybaseMDAProcessStatementTable.

*Table 386. sybaseMDAProcessStatementTable row objects*

Row object	Description
BatchID	Unique identifier for the SQL batch containing the statement.

Table 386. *sybaseMDAProcessStatementTable* row objects (continued)

Row object	Description
ContextID	Stack frame of the procedure, if a procedure.
CPUTime	Number of milliseconds of CPU used by the statement.
DBID	Unique identifier for the database.
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
LineNumber	Line number of the statement within the SQL batch.
LogicalReads	Number of buffers read from the cache.
MemUsageKB	Number of kilobytes of memory used for execution of the statement.
NetworkPacketSize	Size (in bytes) of the network packet currently configured for the session.
PacketSent	Number of network packets sent by Sybase ASE.
PacketsReceived	Number of network packets received by Sybase ASE.
PagesModified	Number of pages modified by the statement.
PhysicalReads	Number of buffers read from disk.
PlanID	Unique identifier for the stored plan for the procedure.
PlansAltered	Number of plans altered at execution time.
ProcedureID	Unique identifier for the procedure.
SPID	Session process identifier.
StartTime	Date when the statement began execution.
WaitTime	Number of milliseconds the task has waited during execution of the statement.

## Process SQL text table

The process SQL text table (*sybaseMDAProcessSQLTextTable*) provides the SQL text that is currently being executed. Table 387 lists the objects in *sybaseMDAProcessSQLTextTable*.

Table 387. *sybaseMDAProcessSQLTextTable* row objects

Row object	Description
BatchID	Unique identifier for the SQL batch containing the SQL text.
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
LineNumber	Line number in SQL batch.
SequenceInLine	If the line of SQL text exceeds the size of the SQL text column, the text is split over multiple rows. Each row has a unique SequenceInLine value that increments by one with each row.
SPID	Unique identifier for this row in the table.
SQLText	SQL text.

## Process procedures table

The process procedures table (sybaseMDAProcessProceduresTable) lists all procedures that are being executed by processes. Table 388 lists the objects in sybaseMDAProcessProceduresTable.

*Table 388. sybaseMDAProcessProceduresTable row objects*

Row object	Description
CompileDate	Compile date of the procedure.
ContextID	Stack frame of the procedure.
DBID	Unique identifier for the database.
DBName	Name of the database that contains the procedure.
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
MemUsageKB	Number of kilobytes of memory used by the procedure.
ObjectID	Unique identifier for the procedure.
ObjectName	Name of the procedure.
ObjectType	Type of procedure.
OwnerName	Name of the object owner.
OwnerUID	Unique identifier for the object owner.
PlanID	Unique identifier for the query plan.
SPID	Session process identifier.

## System plan text table

The system plan text table (sybaseMDASysPlanTextTable) contains data about the most recently generated text plan.

### About this task

Table 389 lists the objects in sybaseMDASysPlanTextTable.

*Table 389. sybaseMDASysPlanTextTable row objects*

Row object	Description
BatchID	Unique identifier for the SQL batch for which the plan was created.
ContextID	If the plan is for a procedure, this object holds the stack frame of the procedure.
DBID	if the plan is for a stored procedure, this object holds the unique identifier for the database where the procedure is stored.
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
PlanID	Unique identifier for the plan.
PlanText	Plan text output.
ProcedureID	if the plan is for a stored procedure, this object holds the unique identifier for the procedure.
SequenceNumber	A monotonically increasing number indicating the position of the PlanText column within the entire plan text.
SPID	Session process identifier.

## System statement table

The system statement table (sybaseMDASysStatementTable) provides statistics about the most recently executed statements.

Table 390 lists the objects in sybaseMDASysStatementTable.

*Table 390. sybaseMDASysStatementTable row objects*

Row object	Description
BatchID	Unique identifier for the SQL batch containing the statement.
ContextID	Current procedure nesting level when executing the statement.
CPUTime	Number of milliseconds of CPU used by the statement.
DBID	Unique identifier for the database.
EndTime	Date when the statement finished execution.
Index	Uniquely identifies the row.
KPID	Kernel process identifier.
LineNumber	Line number of the statement within the SQL batch.
LogicalReads	Number of buffers read from the cache.
MemUsageKB	Number of kilobytes of memory used for execution of the statement.
NetworkPacketSize	Size (in bytes) of the network packet currently configured for the session.
PacketSent	Number of network packets sent by Sybase ASE.
PacketsReceived	Number of network packets received by Sybase ASE.
PagesModified	Number of pages modified by the statement.
PhysicalReads	Number of buffers read from disk.
PlanID	Unique identifier for the stored plan for the procedure.
PlansAltered	The number of plans altered at execution time.
ProcedureID	Unique identifier for the procedure.
SPID	Session process identifier.
StartTime	Date when the statement began execution.
WaitTime	Number of milliseconds the task has waited during execution of the statement.

## Cached procedures table

The cached procedures table (sybaseMDACachedProceduresTable) provides statistics on all procedures currently stored in the procedure cache.

Table 391 lists the objects in sybaseMDACachedProceduresTable.

*Table 391. sybaseMDACachedProceduresTable row objects*

Row object	Description
CompileDate	Date that the procedure was compiled.
DBID	Unique identifier for the database.
DBName	Name of the database.
Index	Unique identifier for this entry in the Cached Procedures Table.
MemUsageKB	Number of kilobytes of memory used by the procedure.

Table 391. *sybaseMDACachedProceduresTable* row objects (continued)

Row object	Description
ObjectID	Unique identifier for the procedure.
ObjectName	Name of the procedure.
ObjectType	Type of procedure.
OwnerName	Number of the object owner.
OwnerUID	Unique identifier for the database owner.
PlanID	Unique identifier for the query plan.

## System SQL text table

The system SQL text table (*sybaseMDASysSQLTextTable*) contains the most recently executed SQL text or the text that is currently executing.

Table 392 lists the objects in *sybaseMDASysSQLTextTable*.

Table 392. *sybaseMDASysSQLTextTable* row objects

Row object	Description
BatchID	Unique identifier for the SQL batch containing the SQL text.
Index	Unique identifier for this row in the table.
KPID	Kernel process identifier.
SequenceInBatch	Indicates the position of this portion of the SQL text within a batch.
SPID	Session process identifier.
SQLText	SQL text.

## SQL query group

The SQL Query group (*sybaseSQLQueryGroup*) provides facilities for defining SQL queries and injecting them into the Sybase ASE server.

### Control table

The control table (*sybaseSQLQueryControlTable*) contains a list of control information for Sybase generic SQL queries.

Table 393 lists the objects in *sybaseSQLQueryControlTable*.

Table 393. *sybaseSQLQueryControlTable* row objects

Row object	Description
DataControl	Controls the monitoring status of this control row:  on(1) - Enables data collection  off(2) - Disables data collection
Index	Uniquely identifies the row.
InTableRowCount	Number of rows to create in the <i>sybaseSQLQueryInTable</i> .
LastUpdate	The value of <i>sysUpTime</i> when the control row last updated its associated data rows.
Owner	Defines the owner/creator of the control row.
Password	Sets the password used when logging on to the monitored Sybase server.

Table 393. *sybaseSQLQueryControlTable* row objects (continued)

Row object	Description
ResultTableRowCount	Number of rows that are present in the sybaseSQLQueryResultsTable. If the query fails, this object contains the value -1.
SampleInterval	Sets the interval (in seconds) at which data is retrieved from the monitored Sybase server. If the value of this object is 0, only one sample is performed.
ServerName	Specifies the name of the Sybase server to be monitored.
ServerStatus	Indicates the current sampling status of the control row:  idle(1) - The row is in an idle state  sampling(2) - The row is currently sampling data from the Sybase server  logonError(3) - The subagent was unable to connect to Sybase server  error(4) - The SQL query failed or was invalid
Status	SNMPv2 row status. Controls creation, activation, and deletion of the row.
TimeForLastUpdate	The time taken to perform the last update.
Username	Sets the username supplied when logging on to the monitored Sybase server.

## Input table

The input table (sybaseSQLQueryInTable) contains the lines of the SQL query to run.

Table 394 lists the objects in sybaseSQLQueryInTable.

Table 394. *sybaseSQLQueryInTable* row objects

Row object	Description
Index	Uniquely identifies the row.
SQLText	The text of the SQL query to run. This query may span multiple lines; each line is limited to 255 characters in length.

## Results table

The results table (sybaseSQLQueryResultsTable) contains the results from the latest SQL query.

Table 395 lists the objects in sybaseSQLQueryResultsTable.

Table 395. *sybaseSQLQueryResultsTable* row objects

Row object	Description
ColumnIndex	The column number of the table returned from the query.
ColumnName	The column name for the returned query (if it exists). If there is no column name, then an integer value (stored as a string) is used instead.
DateResult	The result for this row, if it is a date-and-time value.
IntResult	The result for this row, if it is an integer value.



Table 395. *sybaseSQLQueryResultsTable* row objects (continued)

Row object	Description
RowIndex	The row number of the table returned from the query.
StringResult	The result for this row, if it is a string value.

## Sysmon group

The Sysmon group (`sybaseSysMonGroup`) provides facilities for extracting performance data from reports generated by the Sybase `sp_sysmon` stored procedure.

### Header table

The header table (`sybaseSysMonHeaderTable`) defines the metric types extracted from lines in `sp_sysmon` reports; each metric type is equivalent to one column heading in the report.

Table 396 lists the objects in `sybaseSysMonHeaderTable`.

Table 396. *sybaseSysMonHeaderTable* row objects

Row object	Description
Description	A description of the row.
Format	A comma-separated list of metric types (report column headings). For each heading in the list, the subagent extracts a performance data value and creates a row in <code>sybaseSysMonDataTable</code> .
Index	Uniquely identifies the row.
Owner	Defines the owner/creator of the control row.
Status	SNMPv2 row status. Controls creation, activation and deletion of the row.

### Control table

The control table (`sybaseSysMonControlTable`) contains information for identifying the Sybase servers monitored using `sp_sysmon` stored procedure.

Table 397 lists the objects in `sybaseSysMonControlTable`.

Table 397. *sybaseSysMonControlTable* row objects

Row object	Description
DataControl	Controls the monitoring status of this control row:  on(1) - Enables data collection  off(2) - Disables data collection
Duration	Sets the duration (in seconds) for which the subagent runs the <code>sp_sysmon</code> stored procedure on the monitored Sybase server.
Index	Uniquely identifies the row.
LastUpdate	The value of <code>sysUpTime</code> when this control row last updated its associated data rows.
MatchRows	Specifies the number of rows to create in the match table <code>sybaseSysMonMatchTable</code> .
Owner	Defines the owner/creator of the control row.
Password	Sets the password used when logging on to the monitored Sybase server.

Table 397. *sybaseSysMonControlTable* row objects (continued)

Row object	Description
SampleInterval	Sets the interval (in seconds) at which data is retrieved from the monitored Sybase server. If the value of this object is 0, only one sample is performed.
ServerName	Specifies the name of the Sybase server to be monitored.
ServerStatus	Indicates the current sampling status of the control row:  idle(1) - The row is in an idle state  sampling(2) - The row is currently sampling data from the Sybase server  logonError(3) - The subagent was unable to connect to Sybase server  error(4) - The SQL query failed or was invalid
Status	SNMPv2 row status. Controls creation, activation, and deletion of the row.
TimeForLastUpdate	The time taken to perform the last update.
Username	Sets the username supplied when logging on to the monitored Sybase server.

## Match table

The match table (*sybaseSysMonMatchTable*) defines the performance metrics extracted from *sp\_sysmon* reports.

Table 398 lists the objects in *sybaseSysMonMatchTable*.

Table 398. *sybaseSysMonMatchTable* row objects

Row object	Description
Description	A description of the monitor.
Expression	A regular expression defining the name of the performance metric extracted from the <i>sp_sysmon</i> report.  For instructions on using regular expressions, see the <i>Netcool/SSM Administration Guide</i> .
HeaderIndex	Indexes the row in <i>sybaseSysMonHeaderTable</i> that determines the types of data value extracted for the performance metric specified by <i>sybaseSysMonMatchExpression</i> .
Index	Uniquely identifies the row.

## Data table

The data table (*sybaseSysMonDataTable*) contains the data extracted from *sp\_sysmon* reports. Rows in this table are indexed by a combination of control index, match table index, and report heading column.

Table 399 lists the objects in *sybaseSysMonDataTable*.

Table 399. *sybaseSysMonDataTable* row objects

Row object	Description
FieldIndex	Uniquely identifies the row.

Table 399. *sybaseSysMonDataTable* row objects (continued)

Row object	Description
Name	The name and type of the performance metric extracted from the <code>sp_sysmon</code> report.
Value	The value of the performance metric.



---

## Chapter 41. Netcool/ASM for MSCS

Netcool/ASM for Microsoft Cluster Service Control (MSCS) monitors and controls MSCS server clusters. It monitors and controls the status of nodes, groups, resources, networks and network interfaces, and generate events in response to changes in status.

Netcool/ASM for MSCS consists of the `mscs` subagent and MIB module, which provide the following facilities for monitoring and controlling Microsoft Cluster Server (MSCS) server clusters:

- Monitoring the status of nodes, groups, resources, networks and network interfaces.
- Generating events in response to changes in the state of nodes, groups, resources, networks and network interfaces.
- Controlling cluster nodes, groups, and resources.
- Generating events in response to changes in cluster group or resource ownership as a result of server failover.

---

### Component files

Netcool/ASM for MSCS is comprised of set of component files.

Table 400 lists the `mscs` component files.

*Table 400. mscs component files*

File	Location	Description
<code>mscs.dll</code>	<code>bin</code>	Binary implementation of the <code>mscs</code> subagent.
<code>mscs-mib.mib</code>	<code>mibs</code>	MIB definition document.
<code>mscs.oid</code>	<code>config/oid</code>	<code>mscs</code> subagent object identifier file.

---

### Guidelines

When you load the `mscs` subagent, it automatically gathers cluster metrics.

#### Procedure

To set up the `mscs` subagent for monitoring an MSCS cluster server:

1. Load the `mscs` subagent on the Netcool/SSM installations running on the cluster nodes that you wish to monitor.
  - a. To load the subagent, use the command:

```
subagent load mscs
```

The subagent automatically populates the node, group, resource, network and network interface tables with data from the MSCS cluster.

2. Set the `mscsNodeEvent`, `mscsGroupEvent`, `mscsResourceEvent`, `mscsNetworkEvent` and `mscsNetworkInterfaceEvent` objects to generate events in response to changes in the operational status of nodes, groups, resources, networks and network interfaces as required.

3. Set the `mscsNodeState`, `mscsGroupState` and `mscsResourceState` objects to control the state of nodes, groups and resources as required.

## Restarting groups and resources automatically

The `mscs` subagent is able to restart groups and resources automatically if they go off line.

The `mscsGroupOnlineAttempts` and `mscsResourceOnlineAttempts` objects determine the number of times the subagent will attempt to restart a group or resource whenever it goes off line. If the value of a `mscsGroupOnlineAttempts` or `mscsResourceOnlineAttempts` object is 0, the subagent does not attempt to restart the corresponding group or resource if it goes off line.

If you wish to take a group or resource off line using the `mscs` subagent or some other MSCS administration tool, before doing so ensure that the value of the `mscsGroupOnlineAttempts` or `mscsResourceOnlineAttempts` object for that group or resource has the value 0, otherwise the subagent will attempt to restart it automatically after you take it off line.

## Inivars

The `mscs` subagent provides inivars for configuring its operation.

The inivars are listed in Table 401.

*Table 401. MSCS subagent inivars*

Inivar	Type	Description
<code>MscsResyncTime</code>	int	Sets the amount of time (in seconds) that the subagent listens to an idle cluster. If the subagent does not receive any notifications within the time specified by this variable, it then reconnects to the cluster. This ensures that the subagent is able to resynchronize itself in the event of cluster hang or malfunction.
<code>MscsRetryTime</code>	int	Sets the amount of time (in seconds) between attempts to connect to a cluster. If the subagent cannot connect to a cluster, it logs an error message and waits for the amount of time specified by this variable before attempting to connect again.

For general information about inivars, see the *Netcool/SSM Administration Guide*.

---

## Examples

This example shows how to use the subagent to perform simple monitoring tasks.

### Monitoring changes in status

Monitor an MSCS server cluster and generate events in response to changes in the status of the nodes, groups, resources, networks and network interfaces in the cluster:

```
subagent load rmonc
subagent load mscs

event type=snmp-trap
event descr="Cluster Node status change event"
event create
```

```

snmp set $mscsNodeEvent.* i $?
snmp set $mscsNodeEventStatus.* i 3
event type=snmp-trap
event descr="Cluster Group status change event"
event create
snmp set $mscsGroupEvent.* i $?
snmp set $mscsGroupEventStatus.* i 3
event type=snmp-trap
event descr="Cluster Resource status change event"
event create
snmp set $mscsResourceEvent.* i $?
snmp set $mscsResourceEventStatus.* i 3
event type=snmp-trap
event descr="Cluster Network status change event"
event create
snmp set $mscsNetworkEvent.* i $?
snmp set $mscsNetworkEventStatus.* i 3
event type=snmp-trap
event descr="Cluster Network Interface status change event"
event create
snmp set $mscsNetworkInterfaceEvent.* i $?
snmp set $mscsNetworkInterfaceEventStatus.* i 3

```

## MIB module

The mscs MIB is a subtree of networkharmoni (1977).

The subtree is shown in Figure 72.

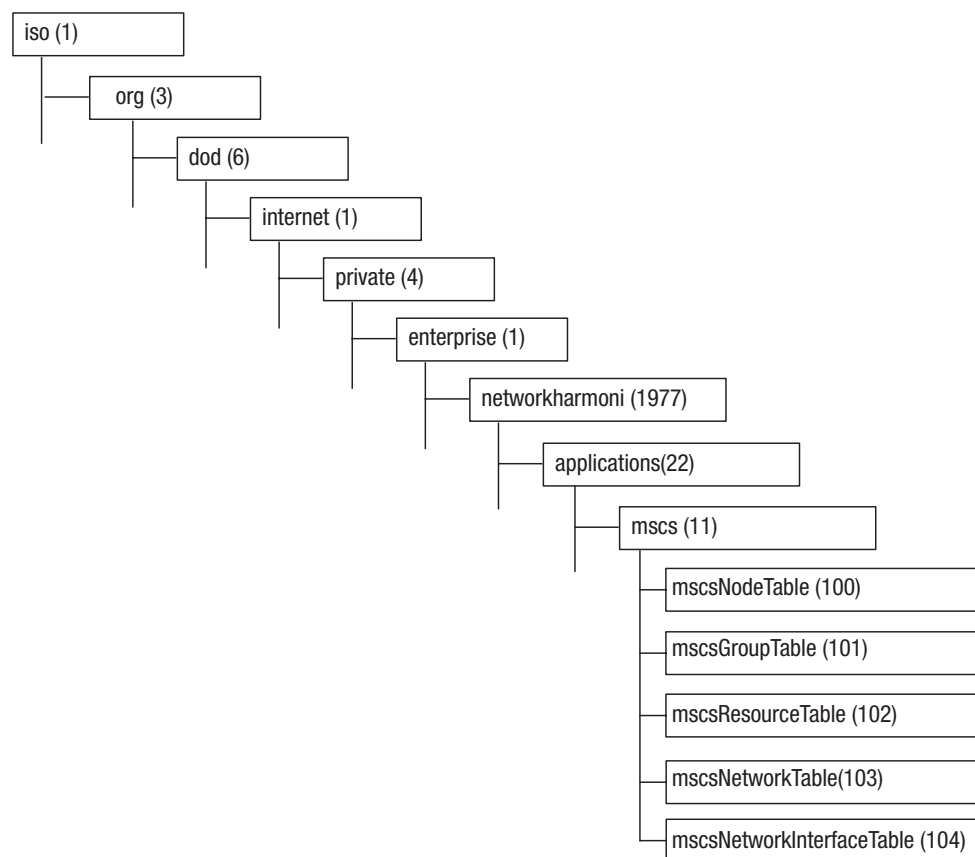


Figure 72. OID tree diagram for mscs

This section provides a summary of the objects defined in the mscs MIB module. For detailed information on all objects in the module, see the mscs-mib.mib document located in the mibs subdirectory of the Netcool/SSM installation.

## MIB objects

The mscs MIB provides a standard SNMP interface.

The MIB module defines:

- A group of scalar objects.
- A node table.
- A group table.
- A resource table.
- A network table.
- A network interface table.

### Scalars

The scalar objects define details about the monitored MSCS server cluster.

Table 402 describes these objects.

*Table 402. mscs scalar objects*

Row object	Description
Description	The description of the cluster.
MixedMode	Indicates the type of cluster services on cluster nodes:  True - The cluster nodes are operating with different versions of the cluster service.  False - All nodes are running identical versions of the cluster service.
Monitored	Controls monitoring of the cluster identified by mscsClusterName:  yes(1) - Monitoring in progress  no(2) - Monitoring suspended  unavailable(3) - Indicates that the subagent is unable to connect to the cluster service
Name	The name of the cluster that this host is attached to.
QuorumDevice	The path name of the quorum log files.
QuorumLogSize	The maximum size of the quorum log files in kilobytes.
QuorumResource	The name of the cluster's quorum resource.



## Node table

The node table (`mcsNodeTable`) contains objects that provide information about cluster nodes, as well as objects for configuring event generation based on changes in a node's status and for controlling a node's operational state.

Table 403 describes the objects defined in `mcsNodeTable`.

Table 403. Node table (`mcsNodeTable`)

Row object	Description
Description	A description of the node.
Event	The index of an event in the RMON event table that is fired whenever the value of <code>mcsNodeState</code> changes. If the value is zero, no event is fired.
EventStatus	RMON-style event throttle control.
EventTime	The value of <code>sysUpTime</code> when the event was last fired.
Name	The name of the node.
State	Indicates the current state of the node:  unknown(1) - The subagent was unable to determine the state of the node  up(2) - The node is active  down(3) - The node is inactive  paused(4) - The node is currently suspended  joining(5) - The node is in the process of joining the cluster  If the value of this object is up(2), setting it to paused(4) suspends the node.  If the value of this object is paused(4), setting it to up(2) reactivates the node.  Setting this object to a value other than up(2) or paused(4) returns a value of inconsistent.

## Group table

The group table (`mcsGroupTable`) contains objects that provide information about cluster groups, as well as objects for configuring event generation based on changes in a group's status and for controlling a group's operational state.

Table 404 describes the objects defined in `mcsGroupTable`.

Table 404. Group table (`mcsGroupTable`)

Row object	Description
Description	A description of the group.
Event	The index of an event in the RMON event table that is fired whenever the value of <code>mcsGroupState</code> changes. If the value is zero, no event is fired.
EventStatus	RMON-style event throttle control.
EventTime	The value of <code>sysUpTime</code> when the event indexed by <code>mcsGroupEvent</code> was last fired.
Name	The name of the group.

Table 404. Group table (mscsGroupTable) (continued)

Row object	Description
OnlineAttempts	Sets the maximum number of attempts to bring a group online. If the value of this object is 0, no attempts will be made.
Owner	If the group is online, this object indicates the name of the node which is the current owner of the group.
OwnerEvent	The index of an event in the RMON event table that is fired whenever the value of mscsGroupOwner changes. If a value of zero is specified, no event is fired.
OwnerEventStatus	RMON-style event throttle control.
OwnerEventTime	The value of sysUpTime when the event indexed by mscsGroupOwnerEvent was last fired.
State	<p>Indicates the current state of the cluster group:</p> <p>unknown(1) - The agent was unable to determine the state of the group</p> <p>online(2) - All the resources in the group are online</p> <p>offline(3) - All the resources in the group are offline, or there are no resources in the group</p> <p>failed(4) - At least one resource in the group is in the failed state</p> <p>partialOnline(5) - At least one resource in the group is online, and no resources are in the pending or failed states</p> <p>pending(6) - At least one resource in the group is in a pending state</p> <p>If the value of this object is online(2) setting it to offline(3) places the group offline.</p> <p>If the value of this object is offline(3), setting it to online(2) brings the group online.</p> <p>Setting this object to a value other than online(2) or offline(3) returns a value of inconsistent.</p>

## Resource table

The resource table (mscsResourceTable) contains objects that provide information about cluster resources, as well as objects for configuring event generation based on changes in a resource's status and for controlling a resource's operational state.

Table 405 describes the objects defined in mscsResourceTable.

Table 405. Resource table (mscsResourceTable)

Row object	Description
Description	A description of the resource.
Event	The index of an event in the RMON event table that is fired whenever the value of mscsResourceState changes. If the value of this object is 0, no event is fired.
EventStatus	RMON-style event throttle control.
EventTime	The value of sysUpTime when the event indexed by mscsResourceEvent was last fired.

Table 405. Resource table (*mcsResourceTable*) (continued)

Row object	Description
Group	Indicates the group that this resource is assigned to. If the resource has not been assigned to a group, this object is empty.
Name	The name of the resource.
OnlineAttempts	Sets the maximum number of attempts to bring a resource online. If the value of this object is 0, no attempts will be made.
Owner	The node that this currently owns this resource.
State	<p>Indicates the current state of the resource:</p> <p>unknown(1) - The subagent was unable to determine the state of the resource</p> <p>initializing(2) - The resource is initializing</p> <p>online(3) - The resource is operational</p> <p>offline(4) - The resource is not operational</p> <p>failed(5) - The resource has failed</p> <p>pending(6) - The resource is in the process of going offline, or coming online</p> <p>onlinePending(7) - The resource is in the process of coming online</p> <p>offlinePending(8) - The resource is in the process of going offline.</p> <p>If the value of this object is online(3) setting it to offline(4) places the resource offline.</p> <p>If the value of this object is offline(4), setting it to online(3) brings the resource online.</p> <p>Setting this object to a value other than online(3) or offline(4) returns a value of inconsistent.</p>

## Network table

The network table (*mcsNetworkTable*) contains objects that provide information about the networks available to the cluster, as well as objects for configuring event generation based on changes in a network's operational state.

Table 406 describes the objects defined in *mcsNetworkTable*.

Table 406. Network table (*mcsNetworkTable*)

Row object	Description
Description	A description of the network.
Event	The index of an event in the RMON event table that is fired whenever the value of <i>mcsNetworkState</i> changes. If the value of this object is 0, no event is fired.
EventStatus	RMON-style event throttle control.
EventTime	The value of <i>sysUpTime</i> when the event indexed by <i>mcsNetworkEvent</i> was last fired.
Name	The name of the network.

Table 406. Network table (mcsNetworkTable) (continued)

Row object	Description
Role	<p>The role of the network in the cluster:</p> <p>none(1) - The network is not used by the cluster</p> <p>clientAccess(2) - The network is used to connect clients to the cluster</p> <p>internalUse(3) - The network is used to carry cluster information only</p> <p>internalAndclient(4) - The network is used for both clients and cluster information</p>
State	<p>The current state of the network:</p> <p>unknown(1)</p> <p>unavailable(2)</p> <p>down(3)</p> <p>partitioned(4)</p> <p>up(5)</p>

## Network interface table

The interface table (mcsNetworkInterfaceTable) contains objects that provide information about the network interfaces available to the cluster as well as objects for configuring event generation based on changes in a network interface's operational state.

Table 407 describes the objects defined in mcsNetworkInterfaceTable.

Table 407. Network interface table (mcsNetworkInterfaceTable)

Row object	Description
Address	The IP address of the network interface.
Description	A description of the network interface.
Event	The index of an event in the RMON event table that is fired whenever the value of mcsNetworkInterfaceState changes. If the value of this object is 0, no event is fired.
EventStatus	RMON-style event throttle control.
EventTime	The value of sysUpTime when the event indexed by mcsNetworkInterfaceEvent was last fired.
Name	The name of the network interface.
Node	The node that owns the network interface.

Table 407. Network interface table (*mcsNetworkInterfaceTable*) (continued)

Row object	Description
State	<p>The current state of the network interface:</p> <p>unknown(1) - The agent was unable to determine the state of the interface</p> <p>unavailable(2) - The node which owns the interface is down</p> <p>failed(3) - The interface is unable to communicate with other interfaces</p> <p>unreachable(4) - The interface cannot communicate with at least 1 other interface</p> <p>up(5) - The interface can communicate with all other interfaces</p> <p>removed(6) - The interface indicated removal, its cable has most likely become unplugged</p>

## Notification types

The mcs MIB module defines notification types for events generated by the subagent.

These types are shown in Figure 73 and listed in Table 408 on page 508.

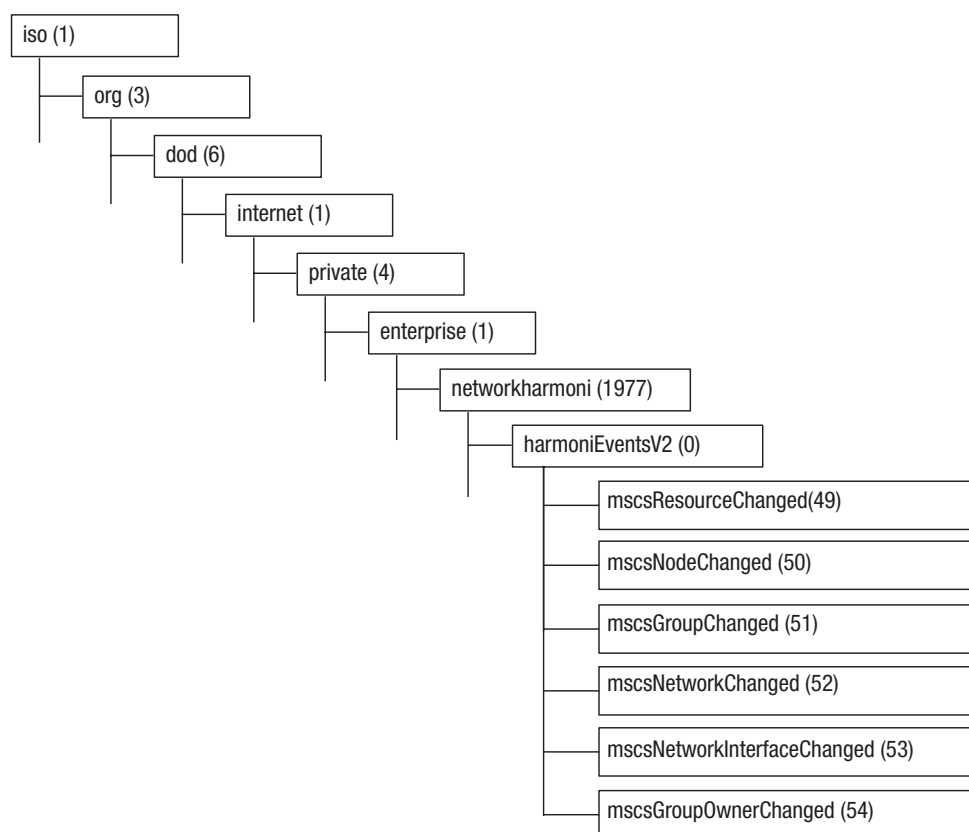


Figure 73. OID tree diagram for mcs notification types

Table 408. Notification types for mscs MIB

Notification type	Description
mscsGroupChanged	Generated when the specified group changes state.  Variable bindings:  mscsGroupName  mscsGroupState
mscsGroupOwnerChanged	Generated when the owner of the specified group changes.  Variable bindings:  mscsGroupName  mscsGroupOwner
mscsNetworkChanged	Generated when the specified network changes state.  Variable bindings:  mscsNetworkName  mscsNetworkState
mscsNetworkInterfaceChanged	Generated when the specified interface changes state.  Variable bindings:  mscsNetworkInterfaceName  mscsNetworkInterfaceState
mscsNodeChanged	Generated when the specified node changes state.  Variable bindings:  mscsNodeName  mscsNodeState
mscsResourceChanged	Generated when the specified resource changes state.  Variable bindings:  mscsResourceName  mscsResourceState

---

## Appendix A. MIB-2 group

The objects in the MIB-2 group are defined in the MIB-2 standard (RFC 1213).

Figure 74 shows the structure of the MIB-2 group.

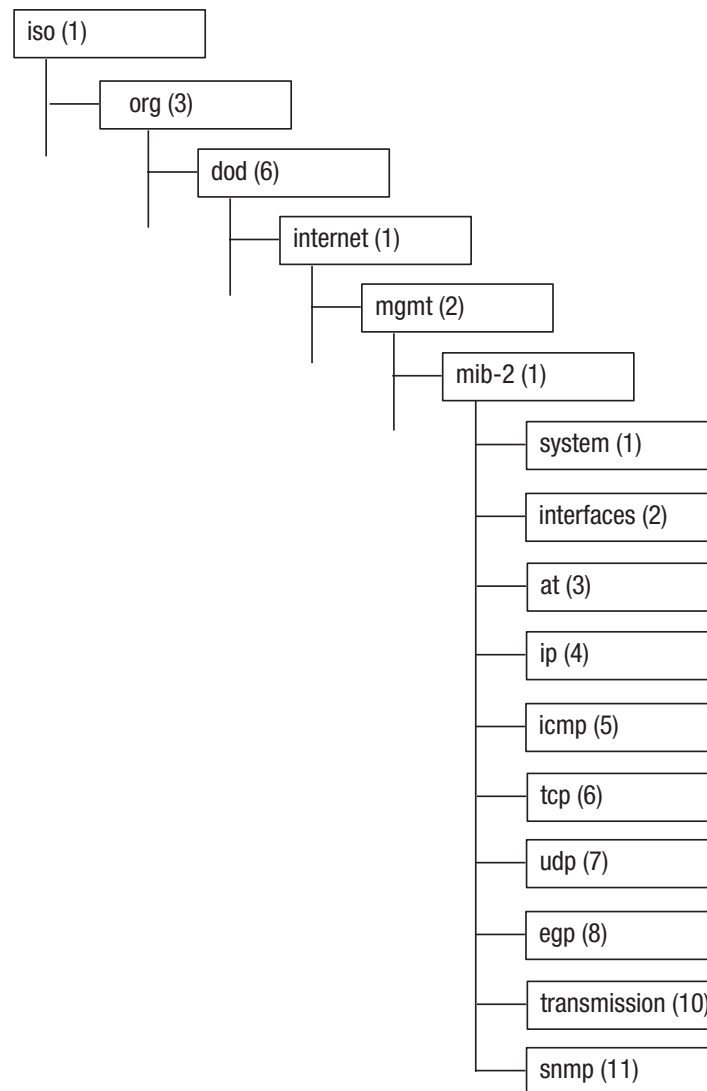


Figure 74. Location of the MIB-2 group

## system

The objects in the system group are shown in Figure 75 and listed in Table 409.

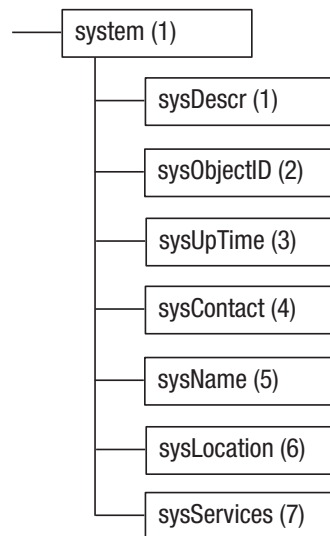


Figure 75. Structure of system group

Table 409. Description of system group objects

Sub- OID	Object	Description
(1)	Desc	<p>A textual description of the entity. This value should include the full name and version identification of the system's hardware type, software operating-system and networking software. It is mandatory that this only contains printable ASCII characters.</p> <p>If required, override this value with the SysDescrOverrideValue inivar. To override sysDescr, stop the agent and edit SysDescrOverrideValue in init.cfg.</p>
(2)	ObjectID	<p>The vendor's authoritative identification of the network management subsystem contained in the entity. This value is allocated within the SMI enterprises subtree (1.3.6.1.4.1) and provides an easy and unambiguous means for determining `what kind of box' is being managed. For example, if vendor `Flintstones, Inc.' was assigned the subtree 1.3.6.1.4.1.4242, it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its `Fred Router'.</p>
(3)	UpTime	<p>The time (in hundredths of a second) since the network management portion of the system was last re-initialized.</p>
(4)	Contact	<p>The textual identification of the contact person for this managed node, together with information on how to contact this person.</p>
(5)	Name	<p>An administratively assigned name for this managed node. By convention, this is the node's fully qualified domain name.</p>
(6)	Location	<p>The physical location of this node (for example, "telephone closet, 3rd floor").</p>



Table 409. Description of system group objects (continued)

Sub- OID	Object	Description
(7)	Services	<p>A value that indicates the set of services that this entity primarily offers. The value is a sum, which initially takes the value zero. Then, for each layer, L, in the range 1 through 7, this node performs a transactions for, 2 raised to (L-1) is added to the sum. For example, a node that performs primarily routing functions would have a value of 4 (<math>2^{(3-1)}</math>). In contrast, a node that is a host offering application services would have a value of 72 (<math>2^{(4-1)}+2^{(7-1)}</math>).</p> <p><b>Note:</b> In the context of the Internet suite of protocols, values should be calculated accordingly:</p> <ul style="list-style-type: none"> <li>1 - physical (for example, repeaters)</li> <li>2 - datalink/subnetwork (for example, bridges)</li> <li>3 - internet (for example, IP gateways)</li> <li>4 - end-to-end (for example, IP hosts)</li> <li>7 - applications (for example, mail relays)</li> </ul> <p>For systems including OSI protocols, layers 5 and 6 may also be counted.</p>

## interfaces

The objects in the interfaces group are shown in Figure 76 and listed in Table 410.

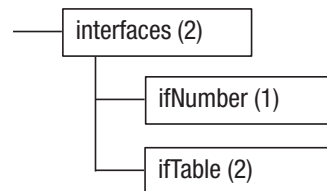


Figure 76. Structure of interfaces group

Table 410. Description of interfaces group objects

Sub- OID	Object	Description
(1)	ifNumber	The number of network interfaces (regardless of their current state) present on this system.
(2)	ifTable	A list of interface entries. The number of entries is given by the value of ifNumber.

## ifTable

The objects in ifTable are shown in Figure 77 and listed in Table 411.

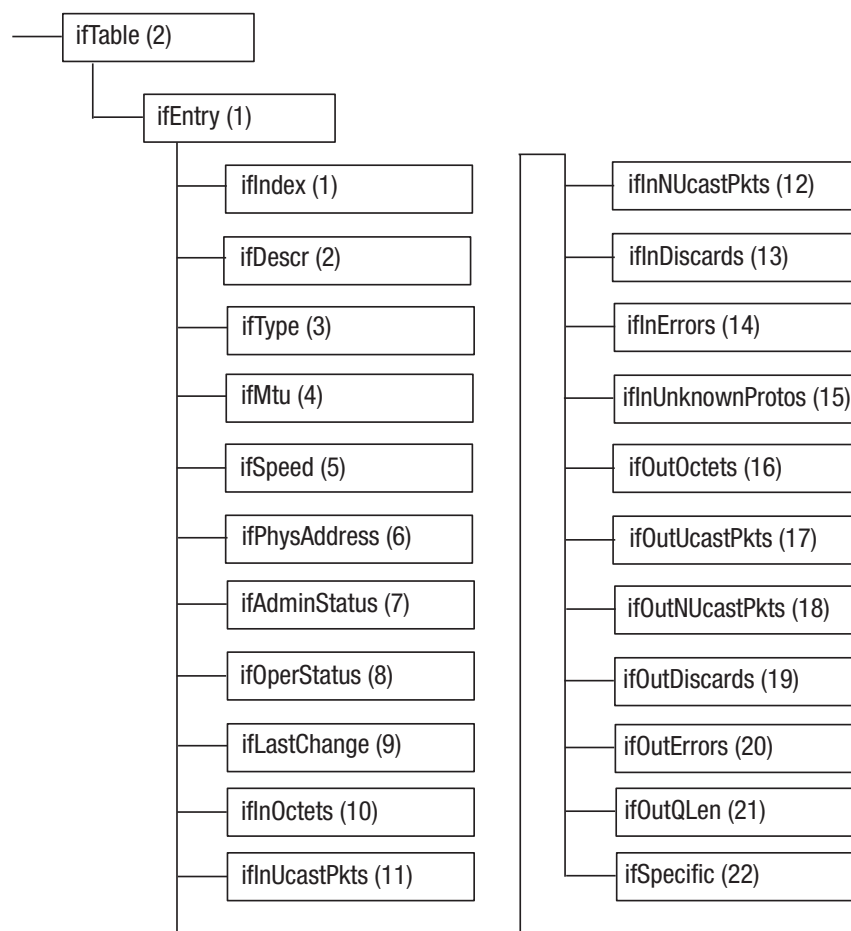


Figure 77. Structure of IfTable

Table 411. Description of IfTable objects

Sub-OID	Object	Description
(1)	Index	A unique value for each interface. Its value ranges between 1 and the value of ifNumber. The value for each interface must remain constant at least from one re-initialization of the entity's network management system to the next re-initialization.
(2)	Descr	A textual string containing information about the interface. This string should include the name of the manufacturer, the product name and the version of the hardware interface.
(3)	Type	The type of interface, distinguished according to the physical/link protocol(s) immediately 'below' the network layer in the protocol stack.

Table 411. Description of IfTable objects (continued)

Sub- OID	Object	Description
(4)	Mtu	The size of the largest datagram that can be sent/received on the interface, specified in octets. For interfaces that are used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface.
(5)	Speed	An estimate of the interface's current bandwidth in bits per second. For interfaces that do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth.
(6)	PhysAddress	The interface's address at the protocol layer immediately 'below' the network layer in the protocol stack. For interfaces that do not have such an address (for example, a serial line), this object should contain an octet string of zero length.
(7)	AdminStatus	The desired state of the interface. The testing(3) state indicates that no operational packets can be passed.
(8)	OperStatus	The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed.
(9)	LastChange	The value of sysUpTime at the time the interface entered its current operational state. If the current state was entered prior to the last re- initialization of the local network management subsystem, then this object contains a zero value.
(10)	InOctets	The total number of octets received on the interface, including framing characters.
(11)	InUcastPkts	The number of subnetwork-unicast packets delivered to a higher-layer protocol.
(12)	InNUcastPkts	The number of non-unicast, that is, subnetwork-broadcast or subnetwork-multicast packets delivered to a higher-layer protocol.
(13)	InDiscards	The number of inbound packets that were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space.
(14)	InErrors	The number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol.
(15)	InUnknownProtos	The number of packets received via the interface that were discarded because of an unknown or unsupported protocol.
(16)	OutOctets	The total number of octets transmitted out of the interface, including framing characters.
(17)	OutUcastPkts	The total number of packets that higher-level protocols requested be transmitted to a subnetwork-unicast address, including those that were discarded or not sent.
(18)	OutNUcastPkts	The total number of packets that higher-level protocols requested be transmitted to a non- unicast, that is, a subnetwork-broadcast or subnetwork-multicast address, including those that were discarded or not sent.
(19)	OutDiscards	The number of outbound packets that were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space.

Table 411. Description of IfTable objects (continued)

Sub- OID	Object	Description
(20)	OutErrors	The number of outbound packets that could not be transmitted because of errors.
(21)	OutQLen	The length of the output packet queue (in packets).
(22)	Specific	A reference to MIB definitions specific to the particular media being used to realize the interface. For example, if an Ethernet realizes the interface, then the value of this object refers to a document defining objects specific to Ethernet. If this information is not present, its value should be set to the OBJECT IDENTIFIER {0 0}, which is a syntactically valid object identifier, and any conforming implementation of ASN.1 and BER must be able to generate and recognize this value.

## at

The objects in atTable are shown in Figure 78 and listed in Table 412.

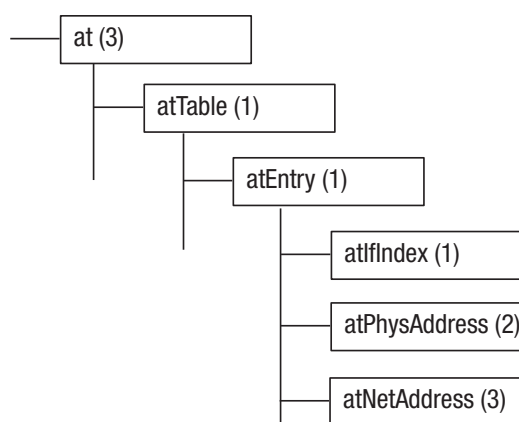


Figure 78. Structure of atTable

Table 412. Description of atTable objects

Sub- OID	Object	Description
(1)	IfIndex	The interface on which this entry's equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.
(2)	PhysAddress	<p>The media-dependent 'physical' address. Setting this object to a null string (one of zero length) has the effect of invalidating the corresponding entry in the atTable object. That is, it effectively disassociates the interface identified with said entry from the mapping identified with said entry.</p> <p>It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant atPhysAddress object.</p>

Table 412. Description of *atTable* objects (continued)

Sub- OID	Object	Description
(3)	NetAddress	The NetworkAddress, for example the IP address, corresponding to the media-dependent 'physical' address.

## ip group

The objects in the *ipGroup* group are shown in Figure 79 and listed in Table 413.

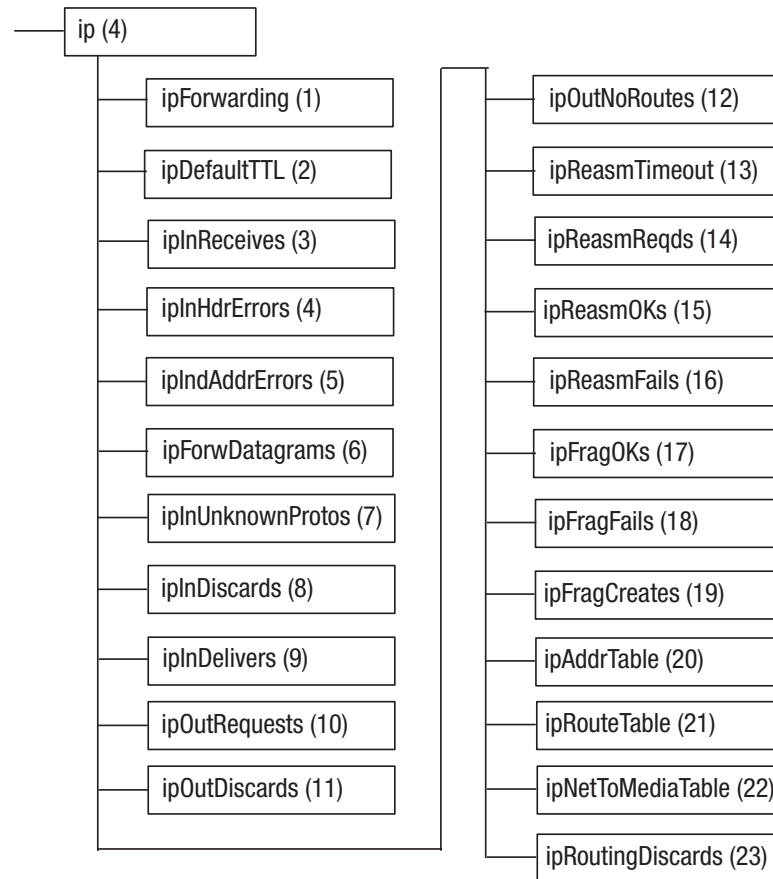


Figure 79. Structure of *ip* group

Table 413. Description of *ip* group objects

Sub- OID	Object	Description
(1)	Forwarding	The indication of whether this entity is acting as an IP gateway in respect to the forwarding of datagrams received by, but not addressed to, this entity. IP gateways forward datagrams. IP hosts do not (except those source-routed via the host).
(2)	DefaultTTL	The default value inserted into the Time-To-Live field of the IP header of datagrams originated at this entity, whenever a TTL value is not supplied by the transport layer protocol.

Table 413. Description of ip group objects (continued)

Sub- OID	Object	Description
(3)	InReceives	The total number of input datagrams received from interfaces, including those received in error.
(4)	InHdrErrors	The number of input datagrams discarded due to errors in their IP headers, including bad checksums, version number mismatch, other format errors, time-to-live exceeded, errors discovered in processing their IP options, etc.
(5)	InAddrErrors	The number of input datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity. This count includes invalid addresses (for example, 0.0.0.0) and addresses of unsupported Classes (for example, Class E). For entities which are not IP Gateways and therefore do not forward datagrams, this counter includes datagrams discarded because the destination address was not a local address.
(6)	ForwDatagrams	The number of input datagrams for which this entity was not their final IP destination, as a result of which an attempt was made to find a route to forward them to that final destination. In entities which do not act as IP Gateways, this counter will include only those packets which were Source-Routed via this entity, and the Source-Route option processing was successful.
(7)	InUnknownProtos	The number of locally addressed datagrams received successfully but discarded because of an unknown or unsupported protocol.
(8)	InDiscards	The number of input IP datagrams for which no problems were encountered to prevent their continued processing, but which were discarded (for example, for lack of buffer space). Note that this counter does not include any datagrams discarded while awaiting re-assembly.
(9)	InDelivers	The total number of input datagrams successfully delivered to IP user-protocols (including ICMP).
(10)	OutRequests	The total number of IP datagrams, which local IP user-protocols (including ICMP) supplied to IP in requests for transmission. Note that this counter does not include any datagrams counted in ipForwDatagrams.
(11)	OutDiscards	The number of output IP datagrams for which no problem was encountered to prevent their transmission to their destination, but which were discarded (for example, for lack of buffer space). Note that this counter would include datagrams counted in ipForwDatagrams if any such packets met this (discretionary) discard criteria.
(12)	OutNoRoutes	The number of IP datagrams discarded because no route could be found to transmit them to their destination. Note that this counter includes any packets counted in ipForwDatagrams, which meet this 'no-route' criterion. Note: this includes any datagrams that a host cannot route because all of its default gateways are down.
(13)	ReasmTimeout	The maximum number of seconds that received fragments are held while they are awaiting reassembly at this entity.
(14)	ReasmReqds	The number of IP fragments received which needed to be reassembled at this entity.
(15)	ReasmOKs	The number of IP datagrams successfully re-assembled.

Table 413. Description of ip group objects (continued)

Sub- OID	Object	Description
(16)	ReasmFails	The number of failures detected by the IP re-assembly algorithm (for whatever reason: time out or error). Note that this is not necessarily a count of discarded IP fragments since some algorithms (notably the algorithm in RFC 815) can lose track of the number of fragments by combining them as they are received.
(17)	FragOKs	The number of IP datagrams that have been successfully fragmented at this entity.
(18)	FragFails	The number of IP datagrams that have been discarded because they needed to be fragmented at this entity but could not be, for example, because their Don't Fragment flag was set.
(19)	FragCreates	The number of IP datagram fragments that have been generated as a result of fragmentation at this entity.
(20)	AddrTable	The table of addressing information relevant to this entity's IP addresses.
(21)	RouteTable	This entity's IP Routing table.
(22)	NetToMediaTable	The IP Address Translation table used for mapping from IP addresses to physical addresses.
(23)	RoutingDiscards	The number of routing entries that were chosen to be discarded even though they are valid. One possible reason for discarding such an entry could be to free-up buffer space for other routing entries.

## ipAddrTable

The objects in ipAddrTable are shown in Figure 80 and listed in Table 414 on page 518.

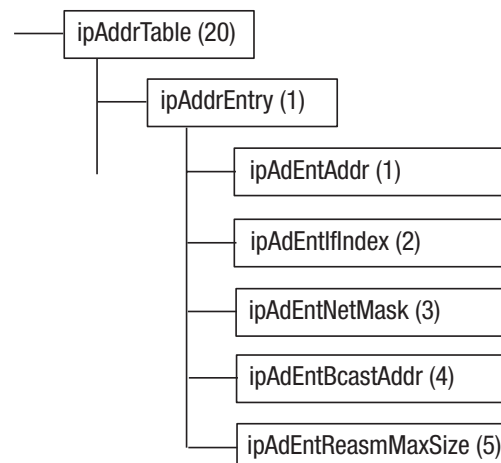


Figure 80. Structure of ipAddrTable

Table 414. Description of ipAddrTable Objects

Sub- OID	Object	Description
(1)	EntAddr	The IP address to which this entry's addressing information pertains.
(2)	EntIfIndex	The index value that uniquely identifies the interface to which this entry is applicable. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.
(3)	EntNetMask	The subnet mask associated with the IP address of this entry. The value of the mask is an IP address with all the network bits set to 1 and all the hosts bits set to 0.
(4)	EntBcastAddr	The value of the least-significant bit in the IP broadcast address used for sending datagrams on the (logical) interface associated with the IP address of this entry. For example, when the Internet standard all-ones broadcast address is used, the value will be 1. This value applies to both the subnet and network broadcasts addresses used by the entity on this (logical) interface.
(5)	EntReasmMaxSize	The size of the largest IP datagram that this entity can re-assemble from incoming IP fragmented datagrams received on this interface.

## ipRouteTable

The objects in ipRouteTable are shown in Figure 81 and listed in Table 415 on page 519.

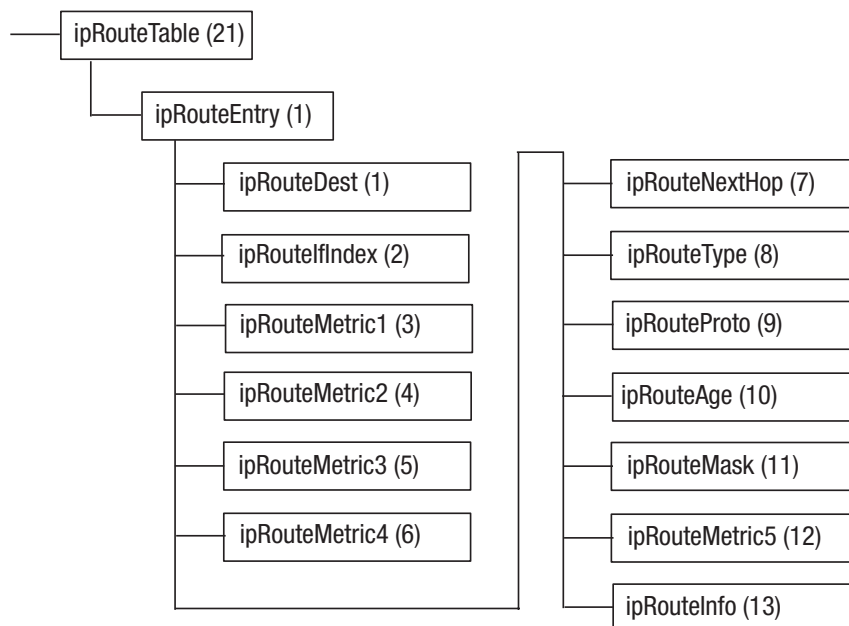


Figure 81. Structure of ifRouteTable



Table 415. Description of *ifRouteTable* Objects

Sub- OID	Object	Description
(1)	Dest	The destination IP address of this route. An entry with a value of 0.0.0.0 is considered a default route. Multiple routes to a single destination can appear in the table, but access to such multiple entries is dependent on the table- access mechanisms defined by the network management protocol in use.
(2)	IfIndex	The index value that uniquely identifies the local interface through which the next hop of this route should be reached. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.
(3)	Metric1	The primary routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.
(4)	Metric2	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.
(5)	Metric3	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.
(6)	Metric4	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.
(7)	NextHop	The IP address of the next hop of this route. In the case of a route bound to an interface that is realized via a broadcast media, the value of this field is the agent's IP address on that interface.
(8)	Type	The type of route. Note: the values direct(3) and indirect(4) refer to the notion of direct and indirect routing in the IP architecture. Setting this object to the value invalid (2) has the effect of invalidating the corresponding entry in the ipRouteTable object. That is, it effectively disassociates the destination identified with said entry from the route identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ipRouteType object.
(9)	Proto	The routing mechanism via which this route was learned. Inclusion of values for gateway routing protocols is not intended to imply that hosts should support those protocols.
(10)	Age	The number of seconds since this route was last updated or otherwise determined to be correct. Note: no semantics of 'too old' can be implied except through knowledge of the routing protocol by which the route was learned.

Table 415. Description of ifRouteTable Objects (continued)

Sub- OID	Object	Description
(11)	Mask	Indicate the mask to be logical-ANDed with the destination address before being compared to the value in the ipRouteDest field. For those systems that do not support arbitrary subnet masks, an agent constructs the value of the ipRouteMask by determining whether the value of the correspondent ipRouteDest field belong to a class-A, B, or C network, and then using one of 255.0.0.0, 255.255.0.0, or 255.255.255.0 masks, respectively. If the value of the ipRouteDest is 0.0.0.0 (a default route), then the mask value is also 0.0.0.0. It should be noted that all IP routing subsystems implicitly use this mechanism.
(12)	Metric5	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.
(13)	Info	A reference to MIB definitions specific to the particular routing protocol which is responsible for this route, as determined by the value specified in the route's ipRouteProto value. If this information is not present, its value should be set to the OBJECT IDENTIFIER {0 0}, which is a syntactically valid object identifier, and any conforming implementation of ASN.1 and BER must be able to generate and recognize this value.

## ipNetToMediaTable

The objects in ipNetToMediaTable are shown in Figure 82 and listed in Table 416.

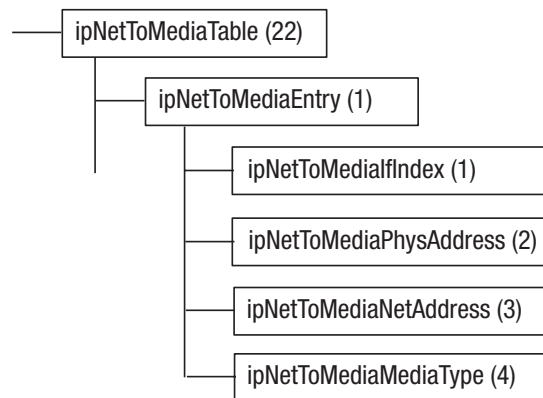


Figure 82. Structure of ipNetToMediaTable

Table 416. Description of ipNetToMediaTable Objects

Sub- OID	Object	Description
(1)	IfIndex	The interface on which this entry's equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.
(2)	PhysAddress	The IP address corresponding to the media-dependent 'physical' address.

Table 416. Description of ipNetToMediaTable Objects (continued)

Sub- OID	Object	Description
(3)	NetAddress	The media-dependent 'physical' address.
(4)	Type	The type of mapping. Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the ipNetToMediaTable. That is, it effectively disassociates the interface identified with said entry from the mapping identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ipNetToMediaType object.

## icmp

The objects in ipRouteTable are shown in Figure 83 and listed in Table 417 on page 522.

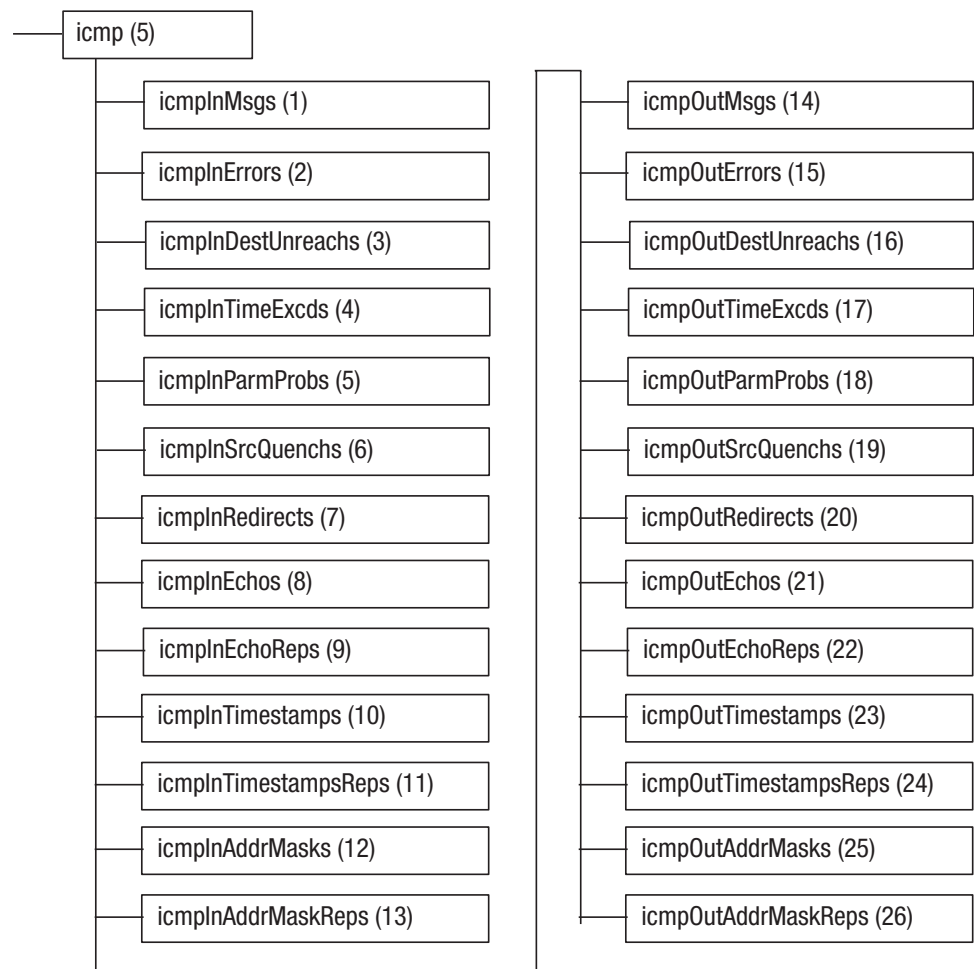


Figure 83. Structure of the icmp group

Table 417. Description of the icmp Group Objects

Sub- OID	Object	Description
(1)	InMsgs	The total number of ICMP messages which the entity received. Note that this counter includes all those counted by icmpInErrors.
(2)	InErrors	The number of ICMP messages that the entity received but determined as having ICMP-specific errors (bad ICMP checksums, bad length, etc.).
(3)	InDestUnreachs	The number of ICMP Destination Unreachable messages received.
(4)	InTimeExcds	The number of ICMP Time Exceeded messages received.
(5)	InParmProbs	The number of ICMP Parameter Problem messages received.
(6)	InSrcQuenchs	The number of ICMP Source Quench messages received.
(7)	InRedirects	The number of ICMP Redirect messages received.
(8)	InEchos	The number of ICMP Echo (request) messages received.
(9)	InEchoReps	The number of ICMP Echo Reply messages received.
(10)	InTimestamps	The number of ICMP Timestamp (request) messages received.
(11)	InTimestampReps	The number of ICMP Timestamp Reply messages received.
(12)	InAddrMasks	The number of ICMP Address Mask Request messages received.
(13)	InAddrMaskReps	The number of ICMP Address Mask Reply messages received.
(14)	OutMsgs	The total number of ICMP messages that this entity attempted to send. Note: this counter includes all those counted by icmpOutErrors.
(15)	OutErrors	The number of ICMP messages that this entity did not send due to problems discovered within ICMP such as a lack of buffers. This value should not include errors discovered outside the ICMP layer such as the inability of IP to route the resultant datagram. In some implementations there may be no types of error that contribute to this counter's value.
(16)	OutDestUnreachs	The number of ICMP Destination Unreachable messages sent.
(17)	OutTimeExcds	The number of ICMP Time Exceeded messages sent.
(18)	OutParmProbs	The number of ICMP Parameter Problem messages sent.
(19)	OutSrcQuenchs	The number of ICMP Source Quench messages sent.
(20)	OutRedirects	The number of ICMP Redirect messages sent. For a host, this object will always be zero, since hosts do not send redirects.
(21)	OutEchos	The number of ICMP Echo (request) messages sent.
(22)	OutEchoReps	The number of ICMP Echo Reply messages sent.
(23)	OutTimestamps	The number of ICMP Timestamp (request) messages sent.
(24)	OutTimestampReps	The number of ICMP Timestamp Reply messages sent.
(25)	OutAddrMasks	The number of ICMP Address Mask Request messages sent.
(26)	OutAddrMaskReps	The number of ICMP Address Mask Reply messages sent.

## tcp

The objects in the tcpGroup are shown in Figure 84 and listed in Table 418.

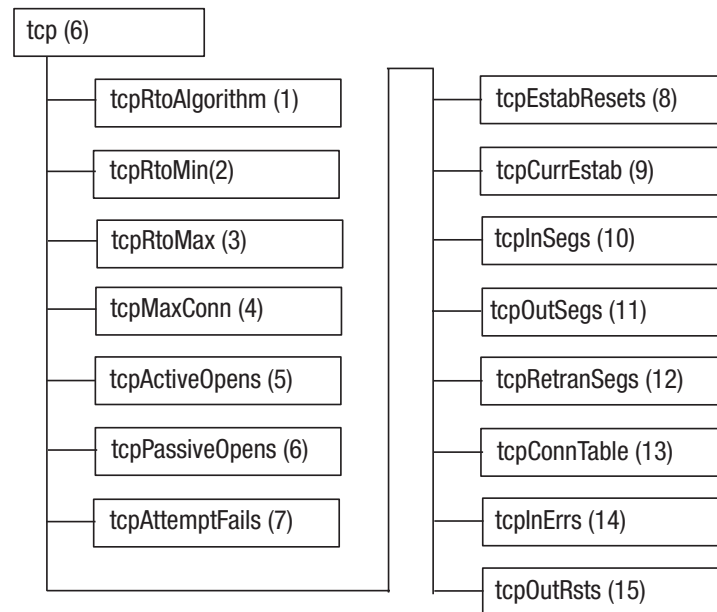


Figure 84. Structure of tcp group

Table 418. Description of tcp group objects

Sub- OID	Object	Description
(1)	RtoAlgorithm	The algorithm used to determine the timeout value used for retransmitting unacknowledged octets.
(2)	RtoMin	The minimum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the LBOUND quantity described in RFC 793.
(3)	RtoMax	The maximum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the UBOUND quantity described in RFC 793.
(4)	MaxConn	The limit on the total number of TCP connections the entity can support. In entities where the maximum number of connections is dynamic, this object should contain the value -1.
(5)	ActiveOpens	The number of times TCP connections have made a direct transition to the SYN-SENT state from the CLOSED state.
(6)	PassiveOpens	The number of times TCP connections have made a direct transition to the SYN-RCVD state from the LISTEN state.

Table 418. Description of tcp group objects (continued)

Sub- OID	Object	Description
(7)	AttemptFails	The number of times TCP connections have made a direct transition to the CLOSED state from either the SYN-SENT state or the SYN-RCVD state, plus the number of times TCP connections have made a direct transition to the LISTEN state from the SYN-RCVD state.
(8)	EstabResets	The number of times TCP connections have made a direct transition to the CLOSED state from either the ESTABLISHED state or the CLOSE-WAIT state.
(9)	CurrEstab	The number of TCP connections for which the current state is either ESTABLISHED or CLOSE-WAIT.
(10)	InSegs	The total number of segments received, including those received in error. This count includes segments received on currently established connections.
(11)	OutSegs	The total number of segments sent, including those on current connections but excluding those containing only retransmitted octets.
(12)	RetransSegs	The total number of segments retransmitted - that is, the number of TCP segments transmitted containing one or more previously transmitted octets.
(13)	ConnTable	A table containing TCP connection-specific information.
(14)	InErrs	The total number of segments received in error (for example, bad TCP checksums).
(15)	OutRsts	The number of TCP segments sent containing the RST flag.

## tcpConnTable

The objects in tcpConnTable are shown in Figure 85 and listed in Table 419 on page 525.

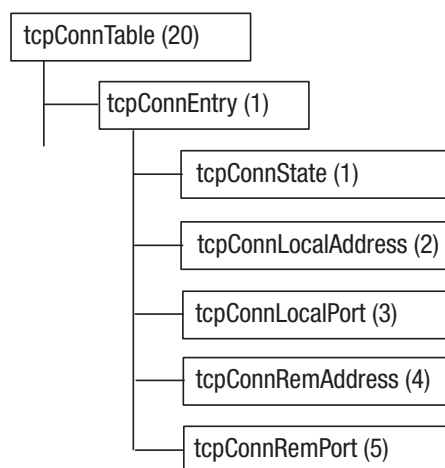


Figure 85. Structure of tcpConnTable

Table 419. Description of tcpConnTable objects

Sub- OID	Object	Description
(1)	State	The state of this TCP connection. The only value that may be set by a management station is deleteTCB(12). Accordingly, it is appropriate for an agent to return a 'badValue' response if a management station attempts to set this object to any other value. If a management station sets this object to the value deleteTCB(12), then this has the effect of deleting the TCB (as defined in RFC 793) of the corresponding connection on the managed node, resulting in immediate termination of the connection. As an implementation-specific option, a RST segment may be sent from the managed node to the other TCP endpoint (note however that RST segments are not sent reliably).
(2)	LocalAddress	The local IP address for this TCP connection. In the case of a connection in the listen state that is willing to accept connections for any IP interface associated with the node, the value 0.0.0.0 is used.
(3)	LocalPort	The local port number for this TCP connection.
(4)	RemAddress	The remote IP address for this TCP connection.
(5)	RemPort	The remote port number for this TCP connection.

## udp group

The objects in the udp group are shown in Figure 86 and listed in Table 420.

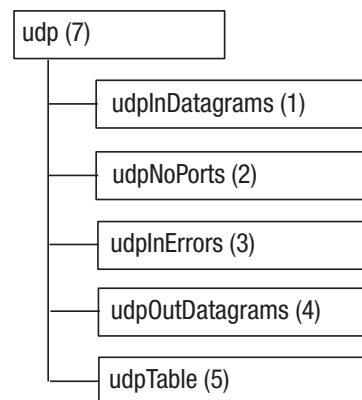


Figure 86. Structure of the udp group

Table 420. Description of udp Group Objects

Sub- OID	Object	Description
(1)	InDatagrams	The total number of UDP datagrams delivered to UDP users.
(2)	NoPorts	The total number of received UDP datagrams for which there was no application at the destination port.
(3)	InErrors	The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port.

Table 420. Description of udp Group Objects (continued)

Sub-OID	Object	Description
(4)	OutDatagrams	The total number of UDP datagrams sent from this entity.
(5)	Table	A table containing UDP listener information.

## udpTable

The objects in udpTable are shown in Figure 87 and listed in Table 421.

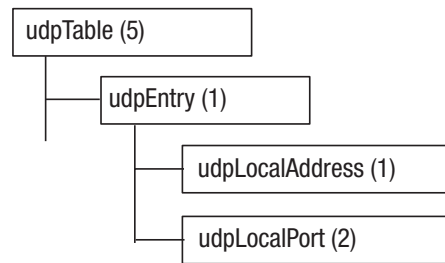


Figure 87. Structure of udpTable

Table 421. Description of udpTable Objects

Sub-OID	Object	Description
(1)	LocalAddress	The local IP address for this UDP listener. In the case of a UDP listener that is willing to accept datagrams for any IP interface associated with the node, the value 0.0.0.0 is used.
(2)	LocalPort	The local port number for this UDP listener.

---

## egpGroup

The objects in egpGroup are shown in Figure 88 on page 527 and listed in Table 422 on page 527.



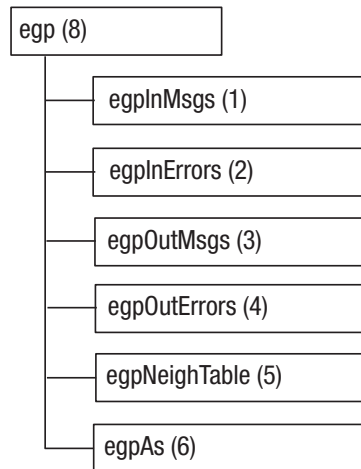


Figure 88. Structure of *egpGroup*

Table 422. Description of *egpGroup* objects

Sub- OID	Object	Description
(1)	InMsgs	The number of EGP messages received without error.
(2)	InErrors	The number of EGP messages received that proved to be in error.
(3)	OutMsgs	The total number of locally generated EGP messages.
(4)	OutErrors	The number of locally generated EGP messages not sent due to resource limitations within an EGP entity.
(5)	NeighTable	The EGP neighbor table.
(6)	As	The autonomous system number of this EGP entity.

## egpNeighTable

The objects in *egpNeighTable* are shown in Figure 89 on page 528 and listed in Table 423 on page 528.

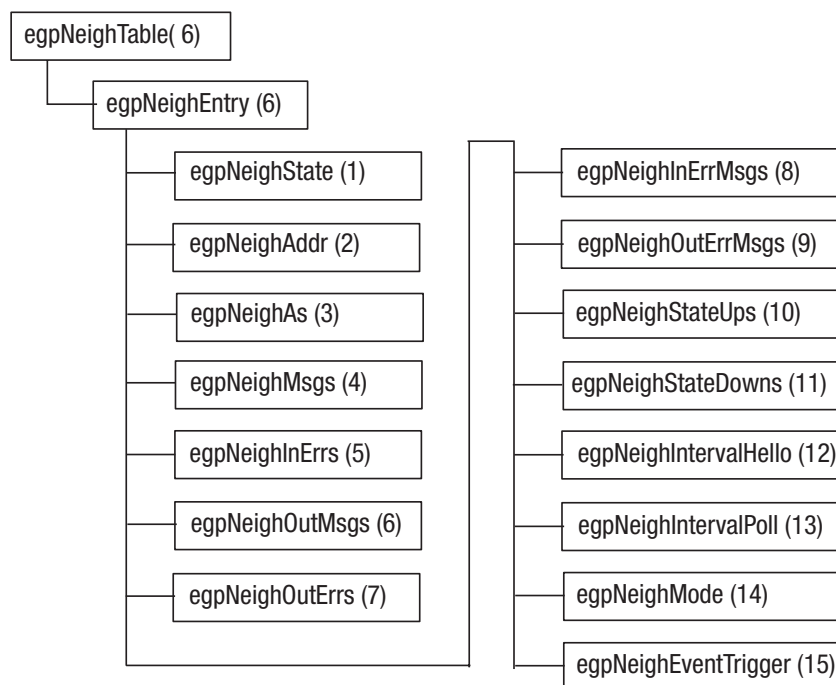


Figure 89. Structure of egpNeighTable

Table 423. Description of egpNeighTable objects

Sub- OID	Object	Description
(1)	State	The IP address of this entry's EGP neighbor.
(2)	Addr	The EGP state of the local system with respect to this entry's EGP neighbor. Each EGP state is represented by a value that is one greater than the numerical value associated with said state in RFC 904.
(3)	As	The autonomous system of this EGP peer. Zero should be specified if the autonomous system number of the neighbor is not yet known.
(4)	InMsgs	The number of EGP messages received without error from this EGP peer.
(5)	InErrs	The number of EGP messages received from this EGP peer that proved to be in error (for example, bad EGP checksum).
(6)	OutMsgs	The number of locally generated EGP messages to this EGP peer.
(7)	OutErrs	The number of locally generated EGP messages not sent to this EGP peer due to resource limitations within an EGP entity.
(8)	InErrMsgs	The number of EGP-defined error messages received from this EGP peer.
(9)	OutErrMsgs	The number of EGP-defined error messages sent to this EGP peer.
(10)	StateUps	The number of EGP state transitions to the UP state with this EGP peer.
(11)	StateDowns	The number of EGP state transitions from the UP state to any other state with this EGP peer.

Table 423. Description of *egpNeighTable* objects (continued)

Sub- OID	Object	Description
(12)	IntervalHello	The interval between EGP Hello command retransmissions (in hundredths of a second). This represents the t1 timer as defined in RFC 904.
(13)	IntervalPoll	The interval between EGP poll command retransmissions (in hundredths of a second). This represents the t3 timer as defined in RFC 904.
(14)	Mode	The polling mode of this EGP entity, either passive or active.
(15)	EventTrigger	A control variable used to trigger operator-initiated Start and Stop events. When read, this variable always returns the most recent value that <i>egpNeighEventTrigger</i> was set to. If it has not been set since the last initialization of the network management subsystem on the node, it returns a value of 'stop'. When set, this variable causes a Start or Stop event on the specified neighbor, as specified on pages 8-10 of RFC 904. Briefly, a Start event causes an Idle peer to begin neighbor acquisition and a non-Idle peer to reinitiate neighbor acquisition. A stop event causes a non-Idle peer to return to the Idle state until a Start event occurs, either via <i>egpNeighEventTrigger</i> or otherwise.

---

## transmission group

The structure of the transmission group is shown in Figure 90.

transmission (10)

Figure 90. Structure of transmission group

---

## snmp group

The objects in the *snmp* group are shown in Figure 91 on page 530 and listed in Table 424 on page 530.

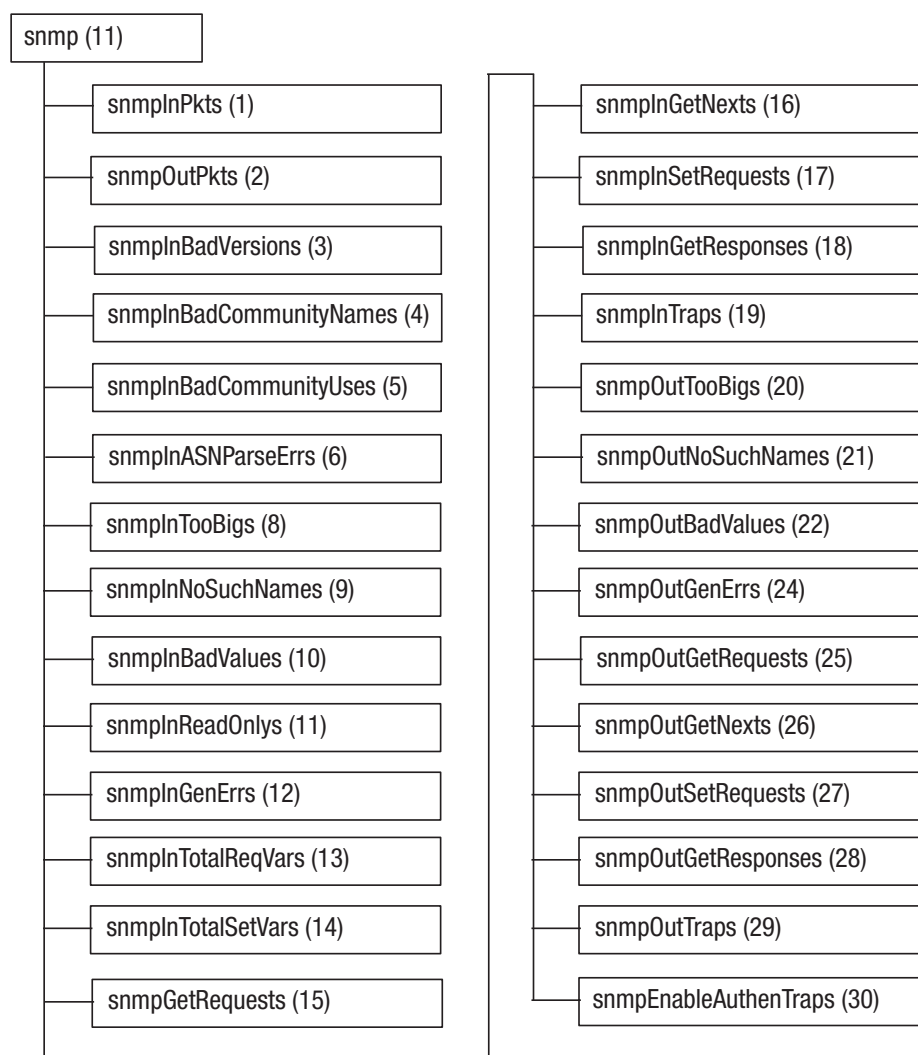


Figure 91. Structure of snmp group

Table 424. Description of snmp group objects

Sub- OID	Object	Description
(1)	InPkts	The total number of Messages delivered to the SNMP entity from the transport service.
(2)	OutPkts	The total number of SNMP Messages that were passed from the SNMP protocol entity to the transport service.
(3)	InBadVersions	The total number of SNMP Messages that were delivered to the SNMP protocol entity and were for an unsupported SNMP version.
(4)	InBadCommunityNames	The total number of SNMP Messages delivered to the SNMP protocol entity that used an SNMP community name not known to said entity.
(5)	InBadCommunityUses	The total number of SNMP Messages delivered to the SNMP protocol entity that represented an SNMP operation, which was not allowed by the SNMP community named in the Message.

Table 424. Description of snmp group objects (continued)

Sub- OID	Object	Description
(6)	InASNParseErrs	The total number of ASN.1 or BER errors encountered by the SNMP protocol entity when decoding received SNMP Messages.
(8)	InTooBigs	The total number of SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is tooBig.
(9)	InNoSuchNames	The total number of SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is noSuchName.
(10)	InBadValues	The total number of SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is badValue.
(11)	InReadOnlys	The total number valid SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is readOnly. It should be noted that it is a protocol error to generate an SNMP PDU, which contains the value readOnly in the error-status field, as such this object is provided as a means of detecting incorrect implementations of the SNMP.
(12)	InGenErrs	The total number of SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is genErr.
(13)	InTotalReqVars	The total number of MIB objects which have been retrieved successfully by the SNMP protocol entity as the result of receiving valid SNMP Get-Request and Get-Next PDUs.
(14)	InTotalSetVars	The total number of MIB objects that have been altered successfully by the SNMP protocol entity as the result of receiving valid SNMP Set-Request PDUs.
(15)	InGetRequests	The total number of SNMP Get-Request PDUs that have been accepted and processed by the SNMP protocol entity.
(16)	InGetNexts	The total number of SNMP Get-Next PDUs that have been accepted and processed by the SNMP protocol entity.
(17)	InSetRequests	The total number of SNMP Set-Request PDUs that have been accepted and processed by the SNMP protocol entity.
(18)	InGetResponses	The total number of SNMP Get-Response PDUs that have been accepted and processed by the SNMP protocol entity.
(19)	InTraps	The total number of SNMP Trap PDUs that have been accepted and processed by the SNMP protocol entity.
(20)	OutTooBigs	The total number of SNMP PDUs that were generated by the SNMP protocol entity and for which the value of the error-status field is tooBig.
(21)	OutNoSuchNames	The total number of SNMP PDUs that were generated by the SNMP protocol entity and for which the value of the error-status is noSuchName.

Table 424. Description of snmp group objects (continued)

Sub- OID	Object	Description
(22)	OutBadValues	The total number of SNMP PDUs that were generated by the SNMP protocol entity and for which the value of the error-status field is badValue.
(24)	OutGenErrs	The total number of SNMP PDUs that were generated by the SNMP protocol entity and for which the value of the error-status field is genErr.
(25)	OutGetRequests	The total number of SNMP Get-Request PDUs that have been generated by the SNMP protocol entity.
(26)	OutGetNexts	The total number of SNMP Get-Next PDUs that have been generated by the SNMP protocol entity.
(27)	OutSetRequests	The total number of SNMP Set-Request PDUs that have been generated by the SNMP protocol entity.
(28)	OutGetResponses	The total number of SNMP Get-Response PDUs that have been generated by the SNMP protocol entity.
(29)	OutTraps	The total number of SNMP Trap PDUs that have been generated by the SNMP protocol entity.
(30)	EnableAuthenTraps	<p>Indicates whether the SNMP agent process is permitted to generate authentication-failure traps. The value of this object overrides any configuration information; as such, it provides a means whereby all authentication-failure traps may be disabled.</p> <p><b>Note:</b> It is strongly recommended that this object be stored in non-volatile memory so that it remains constant between re-initializations of the network management system.</p>

---

## Appendix B. RMON1 MIB group

RMON probes listen to traffic on the network and record statistics or capture traffic according to their configuration. The statistics and saved traffic retrieved by the network management station, using the SNMP protocol, are stored in a MIB.

Using the ASN.1 standard, the managed objects comprising the RMON1 MIB groups, represented by leaves, are grouped into a tree structure, as show in Figure 92. The rmon group specifies the objects relating to the RMON1 MIB objects (RMON2 MIB objects are described in Appendix C, “RMON2 MIB group,” on page 585).

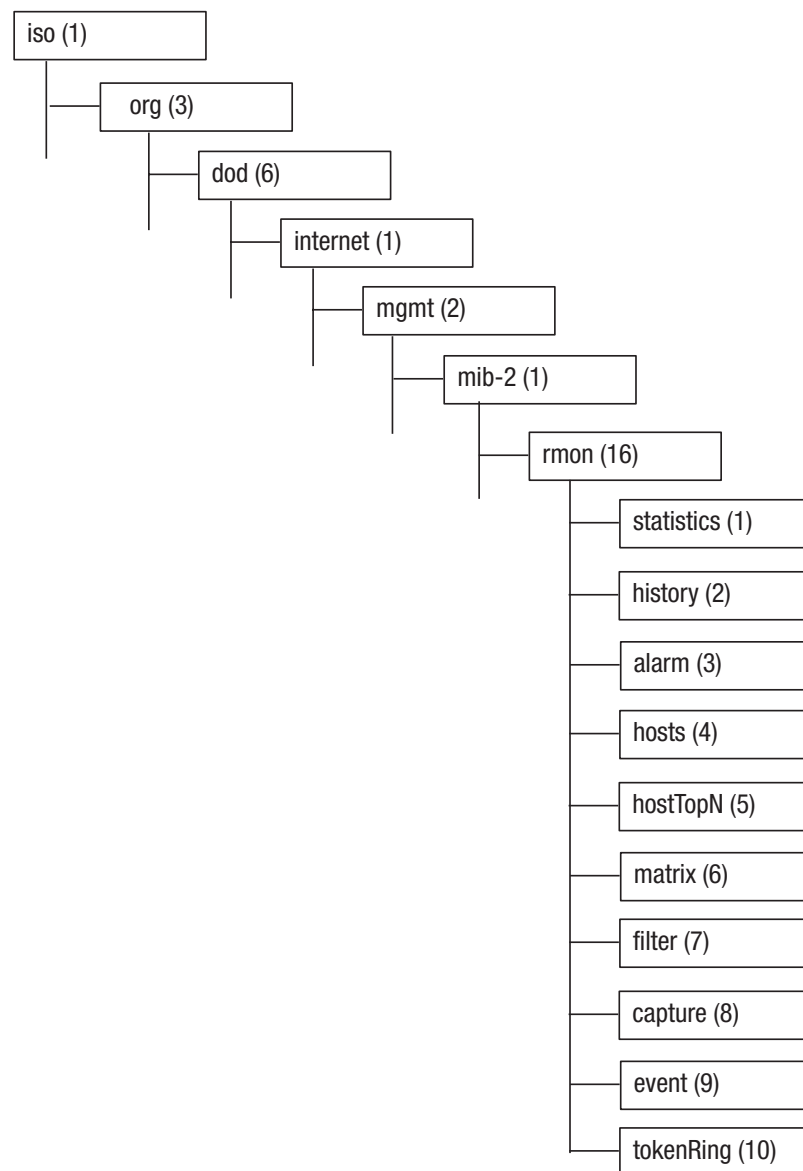


Figure 92. RMON1 MIB group structure

## statistics group

This group maintains low-level utilization and error statistics for each sub-network monitored by Netcool/SSM.

Statistics are in the form of counters that start from zero when a valid control entry is created. Counters for CRC alignment errors, collisions, undersized packets and oversized packets provide useful information about the load and overall health of the sub-network. It should be noted that the statistics group collects statistics on promiscuous traffic across an interface while the interface group collects statistics on total traffic into and out of the agent's interface. This results in some overlap but the statistics group provides much more detail about the network behavior.

### etherStatsTable

etherStatsTable contains statistics for Ethernet interfaces. Each row pertains to one interface. The objects in this table are shown in Figure 93 and listed in Table 425.

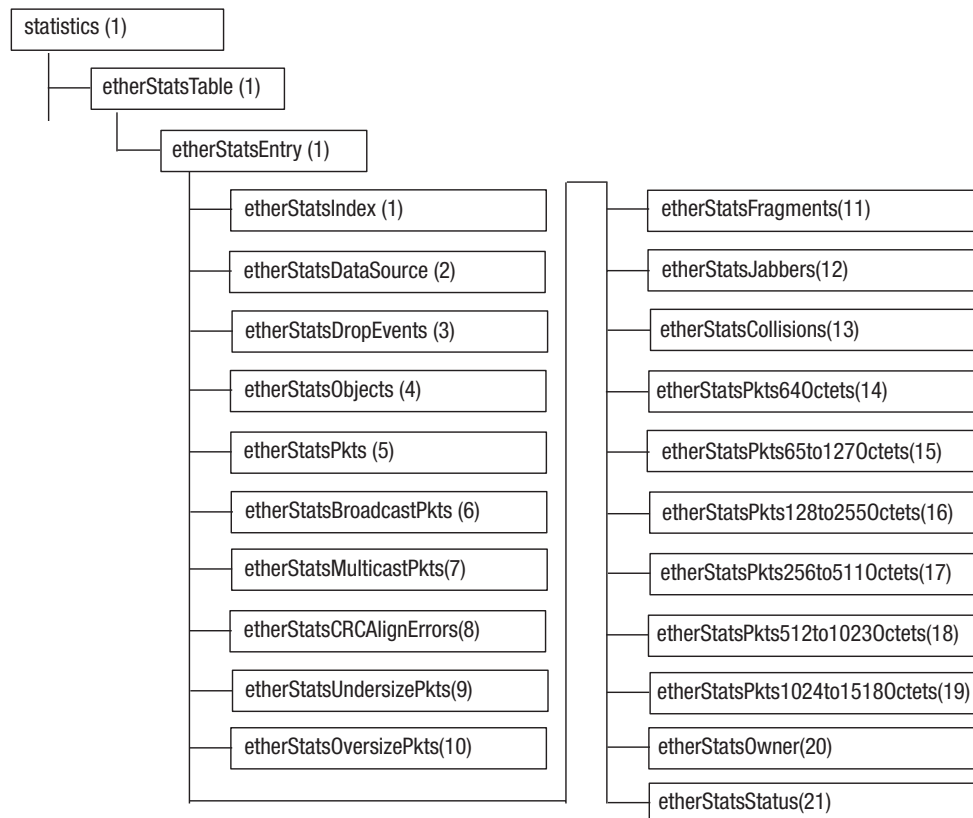


Figure 93. Structure of etherStatsTable

Table 425. Descriptions of etherStatsTable objects

Sub- OID	Object	Description
(1)	Index	Integer index that uniquely identifies this row.



Table 425. Descriptions of etherStatsTable objects (continued)

Sub- OID	Object	Description
(2)	DataSource	Identifies the Ethernet sub network that is the source of the data in this row.
(3)	DropEvents	Number of events detected in which packets were dropped due to lack of resources.
(4)	Octets	Number of received octets of data, including those in bad packets, that can be used to obtain a reasonable estimate for utilization.
(5)	Pkts	Number of received packets, including bad, broadcast and multicast packets.
(6)	BroadcastPkt	Number of good broadcast packets received, not including multicast packets.
(7)	MulticastPkts	Number of good multicast packets received.
(8)	CRCAlignErrors	Number of packets received of the proper size (64-1518 octets) but with either a CRC error or an alignment error.
(9)	UndersizePkts	Number of packets received that were well formed but < 64 octets long.
(10)	OversizePkts	Number of packets received that were well formed but > 1518 octets long.
(11)	Fragments	Number of packets received that were < 64 octets long with either a CRC or an alignment error.
(12)	Jabbers	Number of packets received that were > 1518 octets long with either a CRC or an alignment error; a jabber is the condition where any packet exceeds 20 milliseconds.
(13)	Collisions	Best estimate of the total number of collisions on this Ethernet segment.
(14)	Pkts64Octets	Number of packets, including bad packets, that were 64 octets long.
(15)	Pkts65to127Octets	Number of packets that were between 65 and 127 octets long.
(16)	Pkts128to255Octets	Number of packets between 128 and 255 octets long.
(17)	Pkts256to511Octets	Number of packets between 256 and 511 octets long.
(18)	Pkts512to1023Octets	Number of packets between 512 and 1023 octets long.
(19)	1024to1518Octets	Number of packets between 1024 and 1518 octets long.
(20)	Owner	String that represents the entity that configured this entry and is therefore using the resources assigned to it.
(21)	Status	Current status of this entry, which can take on the values:  1 - valid  2 - createRequest  3 - underCreation  4 - invalid

## tokenRingMLStatsTable

tokenRingMLStatsTable is part of the RMON extension and includes counts of the various MAC-level control packets that manage the token ring. The objects in this table are shown in Figure 94 and listed in Table 426.

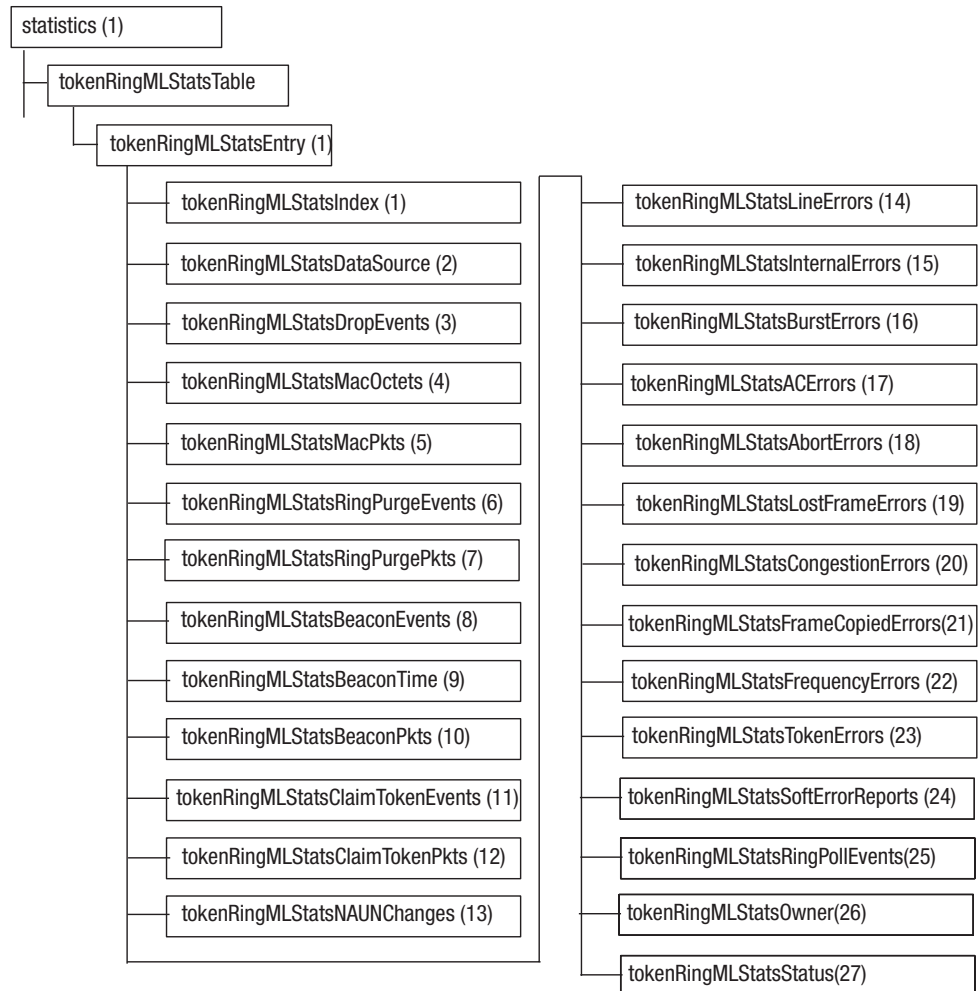


Figure 94. Structure of tokenRingMLStatsTable

Table 426. Descriptions of tokenRingMLStatsTable objects

Sub- OID	Object	Description
(1)	Index	Index that uniquely identifies an entry in the tokenRingStatsTable.
(2)	DataSource	Identifies the source of the data that this entry is configured to analyze.
(3)	DropEvents	Number of events in which packets were dropped due to lack of resources.
(4)	MacOctets	Number of octets of data in MAC packets excluding those in bad frames.
(5)	MacPkts	Number of MAC packets, excluding those in bad frames.

Table 426. Descriptions of tokenRingMLStatsTable objects (continued)

Sub- OID	Object	Description
(6)	RingPurgeEvents	Number of times that the ring enters the ring purge state from the normal ring state.
(7)	RingPurgePkts	Number of ring purge MAC packets detected by the probe.
(8)	BeaconEvents	Number of times the ring enters a beaconing state from a non-beaconing state.
(9)	BeaconTime	Length of time that ring was in a beaconing state.
(10)	BeaconPkts	Number of beacon MAC packets detected by the probe.
(11)	ClaimTokenEvents	Number of times that the ring enters the claim token state from either a normal ring state or a ring purge state.
(12)	ClaimTokenPkts	Number of claim token packets detected by the probe.
(13)	NAUNChanges	Number of NAUN changes detected by the probe.
(14)	LineErrors	Number of line errors reported in error reporting packets detected by the probe.
(15)	InternalErrors	Number of adapter internal errors reported in error reporting packets.
(16)	BurstErrors	Number of burst errors reported in error reporting packets.
(17)	ACErrors	Number of address copied errors reported in error reporting packets.
(18)	AbortErrors	Number of abort delimiters errors reported in error reporting packets.
(19)	LostFrameErrors	Number of lost frame errors reported in error reporting packets.
(20)	CongestionErrors	Number of receive congestion errors reported in error reporting packets.
(21)	FrameCopiedErrors	Number of frame copied errors reported in error reporting packets.
(22)	FrequencyErrors	Number of frequency errors reported in error reporting packets.
(23)	TokenErrors	Number of token errors reported in error reporting packets.
(24)	SoftErrorReports	Number of soft error report frames detected by the probe.
(25)	RingPollEvents	Number of ring poll events detected by the probe.
(26)	Owner	Owner of this entry.
(27)	Status	Status of this entry.

## tokenRingPStatsTable

tokenRingPStatsTable is part of the RMON extension and accumulates statistics on data packets, regardless of their destination address. The objects in this table are shown in Figure 95 on page 538 and listed in Table 427 on page 538.

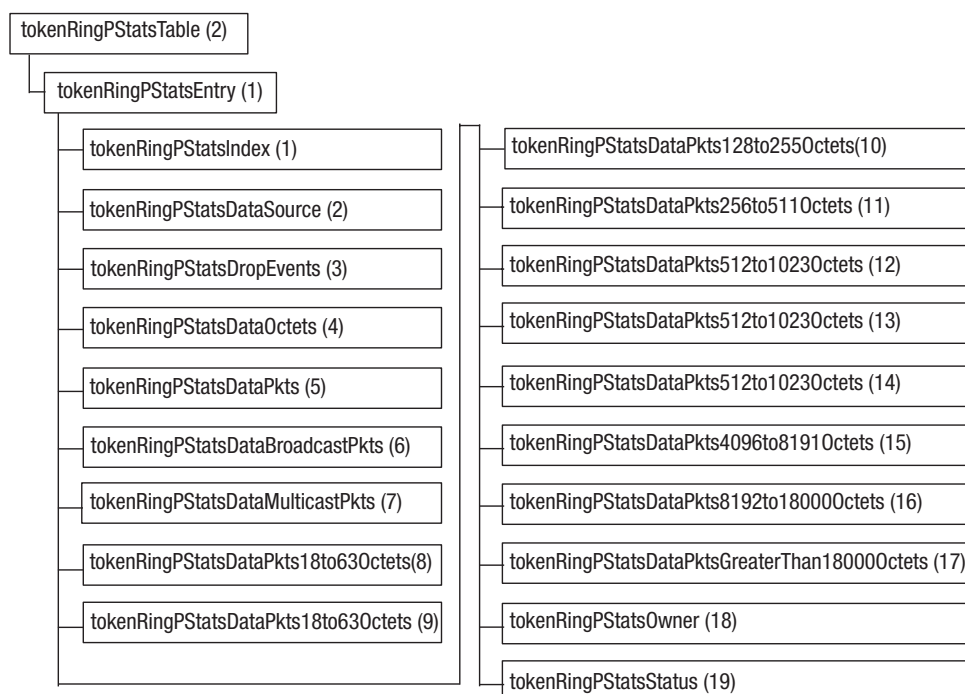


Figure 95. Structure of tokenRingPStatsTable

Table 427. Descriptions of tokenRingPStatsTable objects

Sub- OID	Object	Description
(1)	Index	Index that uniquely identifies this entry.
(2)	DataSource	Identifies the source of the data that this entry is configured to analyze.
(3)	DropEvents	Number of events in which packets were dropped due to lack of resources.
(4)	DataOctets	Number of octets of data in good frames received on the network in non-MAC packets.
(5)	DataPkts	Number of non-MAC packets in good frames received.
(6)	DataBroadcastPkts	Number of good non-MAC frames received that were directed to an LLC broadcast address.
(7)	DataMulticastPkts	Number of good non-MAC frames received that were directed to a local or global multicast or functional address.
(8)	DataPkts18to63octets	Number of good non-MAC frames received between 18 and 63 octets in length inclusive, excluding framing bits but including FCS octets.
(9)	DataPkts64to127octets	Number of good non-MAC frames received between 64 and 127 octets in length inclusive, excluding framing bits but including FCS octets.

Table 427. Descriptions of tokenRingPStatsTable objects (continued)

Sub- OID	Object	Description
(10)	DataPkts128to255octets	Number of good non-MAC frames received between 128 and 255 octets in length inclusive, excluding framing bits but including FCS octets.
(11)	DataPkts256to511octets	Number of good non-MAC frames received between 256 and 511 octets in length inclusive, excluding framing bits but including FCS octets.
(12)	DataPkts512to1023octets	Number of good non-MAC frames received between 512 and 1023 octets in length inclusive, excluding framing bits but including FCS octets.
(13)	DataPkts1024to2047octets	Number of good non-MAC frames received between 1024 and 2047 octets in length inclusive, excluding framing bits but including FCS octets.
(14)	DataPkts2048to4095octet	Number of good non-MAC frames received between 2048 and 4095 octets in length inclusive, excluding framing bits but including FCS octets.
(15)	DataPkts4096to8191octets	Number of good non-MAC frames received between 4096 and 8191 octets in length inclusive, excluding framing bits but including FCS octets.
(16)	DataPkts8192to18000octets	Number of good non-MAC frames received between 8192 and 18000 octets in length inclusive, excluding framing bits but including FCS octets.
(17)	DataPktsGreaterThanOrEqualTo18000octets	Number of good non-MAC frames received > 18000 octets in length inclusive, excluding framing bits but including FCS octets.
(18)	Owner	Owner of this entry.
(19)	Status	Status of this entry.

## etherStats2Table

etherStats2Table is part of the RMON extension and accumulates statistics on data packets, regardless of their destination address. The objects in this table are shown in Figure 96 on page 540 and listed in Table 428 on page 540.

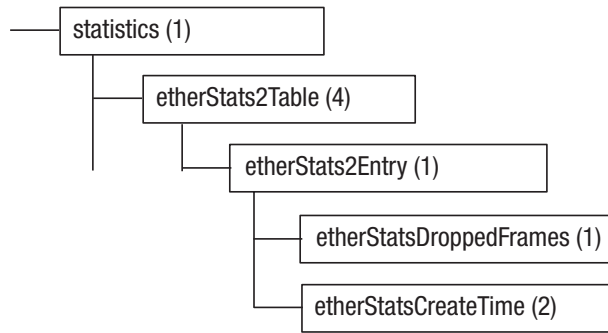


Figure 96. Structure of etherStats2Table

Table 428. Descriptions of etherStats2Table objects

Sub-OID	Object	Description
(1)	DroppedFrames	Number of frames received by the probe and therefore not accounted for in the statsDropEvents, but for which the probe chose not to count for this entry for whatever reason. <b>Note:</b> This is equal to the exact number of frames dropped.
(2)	CreateTime	Value of sysUpTime when this control entry was last activated.

## tokenRingMLStats2Table

tokenRingMLStats2Table is part of the RMON extension and gives additional information of dropped frames not counted in the statsDropEvents. The objects in this table are shown in Figure 97 and listed in Table 429.

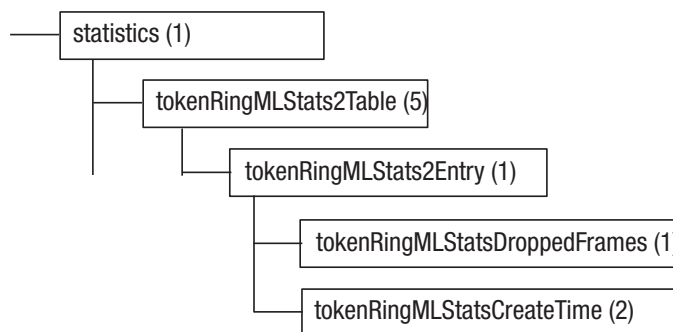


Figure 97. Structure of tokenRingMLStats2Table

Table 429. Descriptions of tokenRingMLStats2Table objects

Sub-OID	Object	Description
(1)	DroppedFrames	Number of frames received by the probe and therefore not accounted for in the statsDropEvents, but which the probe chose not to count for this entry for whatever reason.
(2)	CreateTime	Value of sysUpTime when this control entry was last activated.

## tokenRingPStats2Table

tokenRingPStats2Table is part of the RMON extension and gives additional information of dropped frames not counted in the statsDropEvents. The objects in this table are shown in Figure 98 and listed in Table 430.

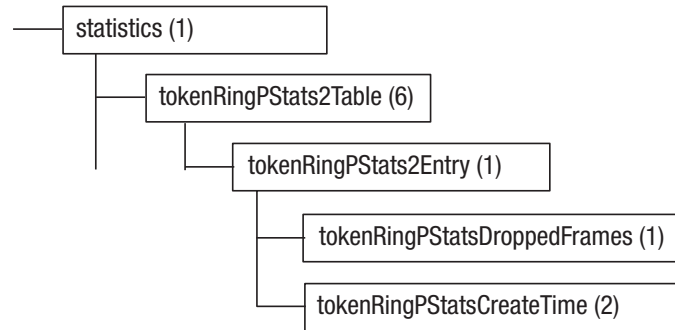


Figure 98. Structure of tokenRingStats2Table

Table 430. Descriptions of tokenRingStats2Table objects

Sub- OID	Object	Description
(1)	DroppedFrames	Number of frames received by the probe and therefore not accounted for in the statsDropEvents, but for which the probe chose not to count for this entry for whatever reason.
(2)	CreateTime	Value of sysUpTime when this control entry was last activated.

---

## history group

The history group records periodic statistical samples from information available in the statistics group. It defines sampling functions for one or more of the interfaces of the monitor.

Each row in the historyControlTable defines a set of samples at a particular interface. As each sample is collected, it is stored in a new row of etherHistoryTable.

## historyControlTable

historyControlTable specifies the interface and the details of the sampling function used to construct the history. The objects in this table are shown in Figure 99 on page 542 and listed in Table 431 on page 542.

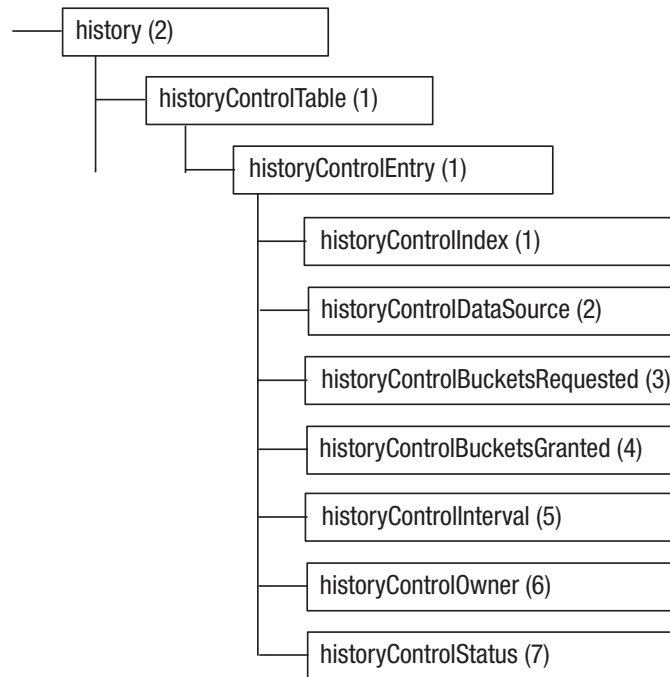


Figure 99. Structure of historyControlTable

Table 431. Descriptions of historyControlTable objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies the row in the historyControlTable; each entry defines a set of samples at a particular interval for an interface on this device.
(2)	DataSource	Identifies the interface, and hence the sub network that is the source for data in this row.
(3)	BucketsRequested	Requested number of discrete sampling intervals over which data is to be saved; when this entry is created or modified, the probe should set historyControlBucketsGranted as close as possible to the value of this object.
(4)	BucketsGranted	Actual number of sampling intervals over which to save data.
(5)	Interval	Interval, in seconds, over which data is sampled for each bucket; may be set to a value between 1 and 3600 seconds.
(6)	Owner	Owner of this entry.
(7)	Status	Status of this entry.



## EtherHistoryTable

The objects in EtherHistoryTable are shown in Figure 100 and listed in Table 432.

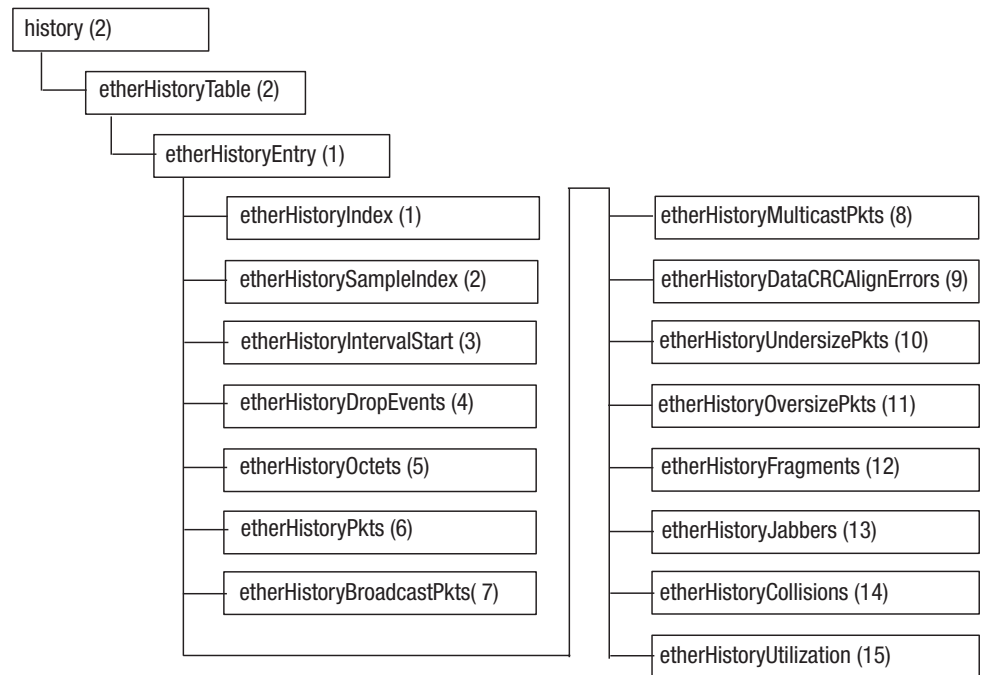


Figure 100. Structure of etherHistoryTable

Table 432. Descriptions of etherHistoryTable objects

Sub- OID	Object	Description
(1)	Index	Index to the history of which this entry is a part.
(2)	SampleIndex	Index that uniquely identifies the particular sample that this entry represents among all samples associated with the same row of the historyControlTable; the index starts at 1 and increases by 1 as each new sample is taken.
(3)	IntervalStart	Value of sysUpTime (in the MIB2 systems group) at the start of the interval over which this sample is being measured.
(4)	DropEvents	Number of events in which packets were dropped by the probe due to lack of resources during this sample interval.
(5)	Octets	Number of octets of data received, including those in bad packets, during this interval.
(6)	Pkts	Number of packets of data received during this interval.
(7)	BroadcastPkts	Number of good packets received that were addressed to the broadcast address.
(8)	MulticastPkts	Number of good packets received that were addressed to the multicast address.
(9)	CRCAlignErrors	Number of packets received during this interval that had the correct length, but had either a bad Frame Check Sequence (FCS) or a bad FCS with a non integral number of octets.

Table 432. Descriptions of etherHistoryTable objects (continued)

Sub- OID	Object	Description
(10)	UndersizePkts	Number of packets that were < 64 octets in length but were otherwise well-formed.
(11)	OversizePkts	Number of packets that were >1518 octets in length but were otherwise well-formed.
(12)	Fragments	Number of packets <64 octets in length and had either a bad FCS or an alignment error.
(13)	Jabbers	Number of packets that were >1518 octets in length and had either a bad FCS or an alignment error.
(14)	Collisions	Best estimate of the number of collisions on this Ethernet segment during this sample interval.
(15)	Utilization	Best estimate for the utilization of the sub network during this sampling interval.

## tokenRingMLHistoryTable

tokenRingMLHistoryTable is part of the RMON extension and collects information on the counts maintained in the tokenRingMLStatsTable. The objects in this table are shown in Figure 101 on page 545 and listed in Table 433 on page 545.

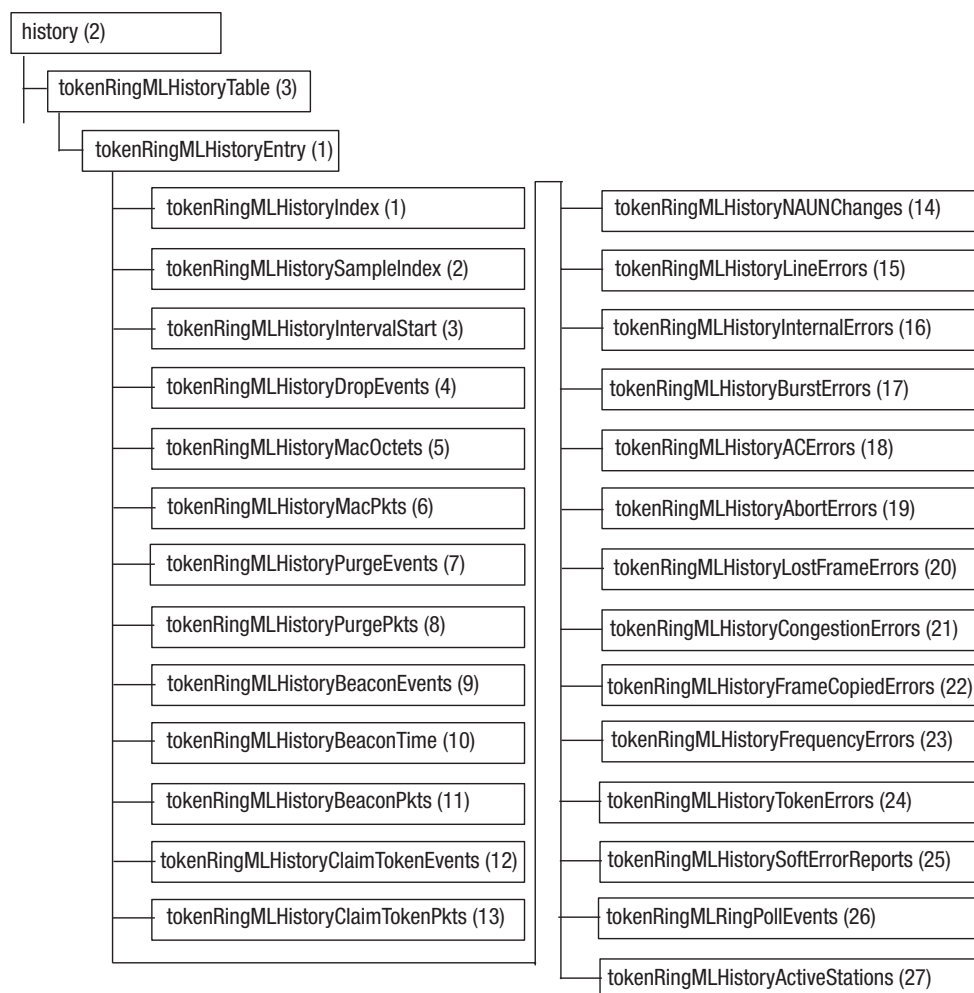


Figure 101. Structure of tokenRingMLHistoryTable

Table 433. Descriptions of tokenRingMLHistoryTable objects

Sub- OID	Object	Description
(1)	Index	History of which this entry is a part.
(2)	SampleIndex	Index that uniquely identifies the particular Mac-layer sample; this entry represents all Mac-layer samples associated with the same historyControlEntry.
(3)	IntervalStart	Value of sysUpTime at the start of the interval over which this sample was measured.
(4)	DropEvents	Number of events in which packets were dropped due to lack of resources during this sampling interval.
(5)	MacOctets	Number of octets of data in MAC packets, excluding those in bad frames during this sampling interval.
(6)	MacPkts	Number of MAC packets detected, excluding those in bad frames during this sampling interval.

Table 433. Descriptions of tokenRingMLHistoryTable objects (continued)

Sub- OID	Object	Description
(7)	RingPurgeEvents	Number of times ring enters a ring purge state during this sample interval.
(8)	RingPurgePkts	Number of ring purge MAC packets detected by the probe during this sampling interval.
(9)	BeaconEvents	Number of times the ring enters a beaconing state from a non-beaconing state during this sampling interval.
(10)	BeaconTime	Amount of time the ring has been in the beaconing state during this sampling interval.
(11)	BeaconPkts	Number of beacon MAC packets detected by the probe during this sampling interval.
(12)	ClaimTokenEvents	Number of times the ring enters the claim token state from either a normal ring state or a ring purge state during this sampling interval.
(13)	ClaimTokenPkts	Number of claim token packets detected by the probe during this sampling interval.
(14)	NAUNChanges	Number of NAUN changes detected by the probe during this sampling interval.
(15)	LineErrors	Number of line errors reported in error reporting packets detected by probe during this sampling interval.
(16)	InternalErrors	Number of adapter internal errors reported in error reporting packets during this sampling interval.
(17)	BurstErrors	Number of burst errors reported in error reporting packets during this sampling interval.
(18)	ACErrors	Number of address copied errors reported in error reporting packets during this sampling interval.
(19)	AbortErrors	Number of abort delimiter errors reported in error reporting packets during this sampling interval.
(20)	LostFrameError	Number of lost frame errors reported in error reporting packets during this sampling interval.
(21)	CongestionErrors	Number of receive congestion errors reported in error reporting packets during this sampling interval.
(22)	FrameCopiedErrors	Number of frame copied errors reported in error reporting packets during this sampling interval.
(23)	FrequencyErrors	Number of frequency errors reported in error reporting packets during this sampling interval.
(24)	TokenErrors	Number of token errors reported in error reporting packets during this sampling interval.
(25)	SoftErrorReports	Number of soft error report frames detected by the probe during this sampling interval.
(26)	tokenRingMLRingPollEvents	Number of ring poll events detected by the probe during this sampling interval.

Table 433. Descriptions of tokenRingMLHistoryTable objects (continued)

Sub- OID	Object	Description
(27)	ActiveStations	Maximum number of active stations detected by the probe during this sampling interval.

## tokenRingPHistoryTable

tokenRingPHistoryTable is part of the RMON extension and collects information on the counts maintained in the tokenRingPStatsTable. The objects in this table are shown in Figure 102 and listed in Table 434.

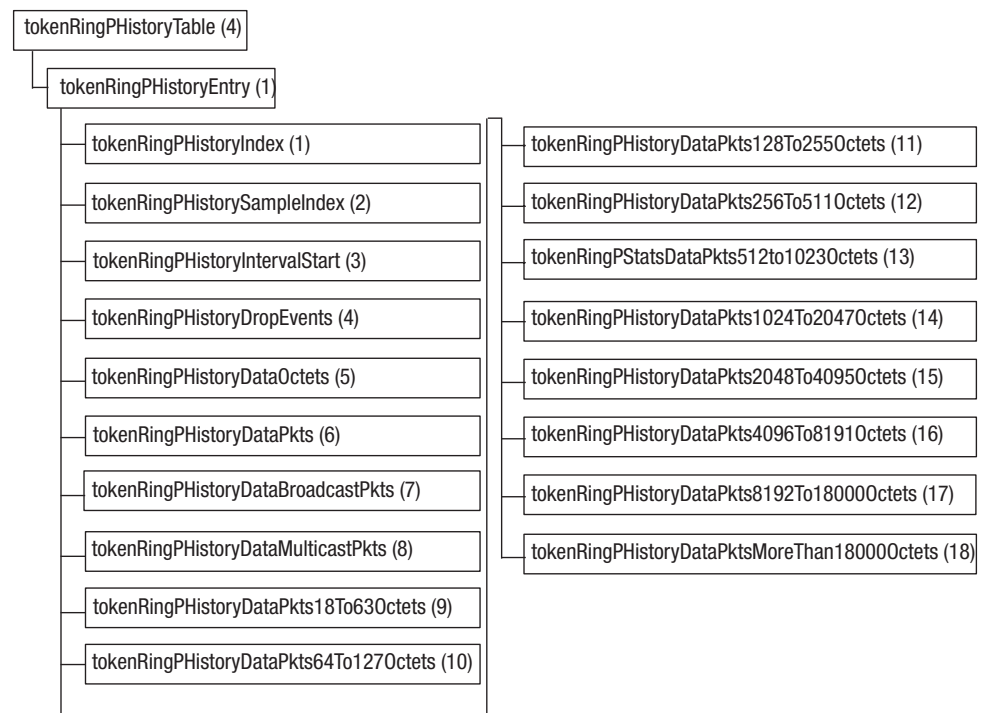


Figure 102. Structure of tokenRingPHistoryTable

Table 434. Descriptions of tokenRingPHistoryTable objects

Sub- OID	Object	Description
(1)	Index	History of which this entry is a part.
(2)	SampleIndex	Index that uniquely identifies the particular sample this entry represents among all samples associated with the same historyControlEntry.
(3)	IntervalStart	Value of sysUpTime at the start of the interval over which this sample was taken.
(4)	DropEvents	Number of events in which packets were dropped due to a lack of probe resources during this sampling interval.
(5)	DataOctets	Number of octets of data in good frames received on the network in non-MAC packets during this sampling interval.

Table 434. Descriptions of tokenRingPHistoryTable objects (continued)

Sub- OID	Object	Description
(6)	DataPkts	Number of non-MAC packets in good frames received during this sampling interval.
(7)	DataBroadcastPkts	Number of good non-MAC frames received directed to an LLC broadcast address during this sampling interval.
(8)	DataMulticastPkts	Number of good non-MAC frames received directed to a local or global multicast or functional address during this sampling interval.
(9)	DataPkts18to63octets	Number of good non-MAC frames received between 18 and 63 octets in length inclusive, excluding framing bits but including FCS octets, during this sampling interval.
(10)	DataPkts64to127octets	Number of good non-MAC frames between 64 and 127 octets long during this sampling interval.
(11)	DataPkts128to255octets	Number of good non-MAC frames between 128 and 255 octets long during this sampling interval.
(12)	DataPkts256to511octets	Number of good non-MAC frames between 256 and 511 octets long during this sampling interval.
(13)	DataPkts512to1023octets	Number of good non-MAC frames between 512 and 1023 octets long during this sampling interval.
(14)	DataPkts1024to2047octets	Number of good non-MAC frames between 1024 and 2047 octets long during this sampling interval.
(15)	DataPkts2048to4095octets	Number of good non-MAC frames between 2048 and 4095 octets long this sampling interval.
(16)	DataPkts4096to8191octets	Number of good non-MAC frames between 4096 and 8191 octets long during this sampling interval.
(17)	DataPkts8192to18000octets	Number of good non-MAC frames between 8192 and 18000 octets long during this sampling interval.
(18)	DataPktsMoreThan18000octets	Number of good non-MAC frames >18000 octets long received during this sampling interval.

## historyControl2Table

historyControl2Table is part of the RMON extension and gives additional information of dropped frames not counted in the statsDropEvents. The objects in this table are shown in Figure 103 and listed in Table 435.

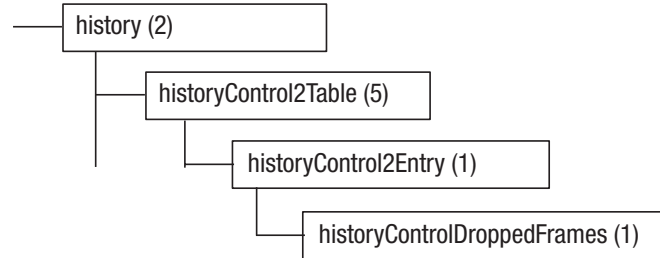


Figure 103. Structure of historyControl2Table

Table 435. Descriptions of historyControl2Table objects

Sub-OID	Object	Description
(1)	DroppedFrames	Number of frames received by the probe and therefore not accounted for in the statsDropEvents, but which the probe chose not to count for some reason.

---

## alarm group

The alarm group allows the management console user to set a sampling interval and alarm threshold for any counter or integer recorded by Netcool/SSM.

A rising threshold is crossed if the current sampled value is greater than or equal to the rising threshold and the value at the last sampling interval was less than the rising threshold. After a rising-alarm event is generated, another such event will not be generated until the sampled value has fallen below the rising threshold, reached the falling threshold, and then subsequently reached the rising threshold again, that is, a hysteresis mechanism limits the generation of events. The reverse situation applies for a falling threshold.

## alarmTable

alarmTable contains information about the sampling interval and threshold parameters for the particular alarm variable being monitored. The objects in this table are shown in Figure 104 on page 550 and listed in Table 436 on page 550.

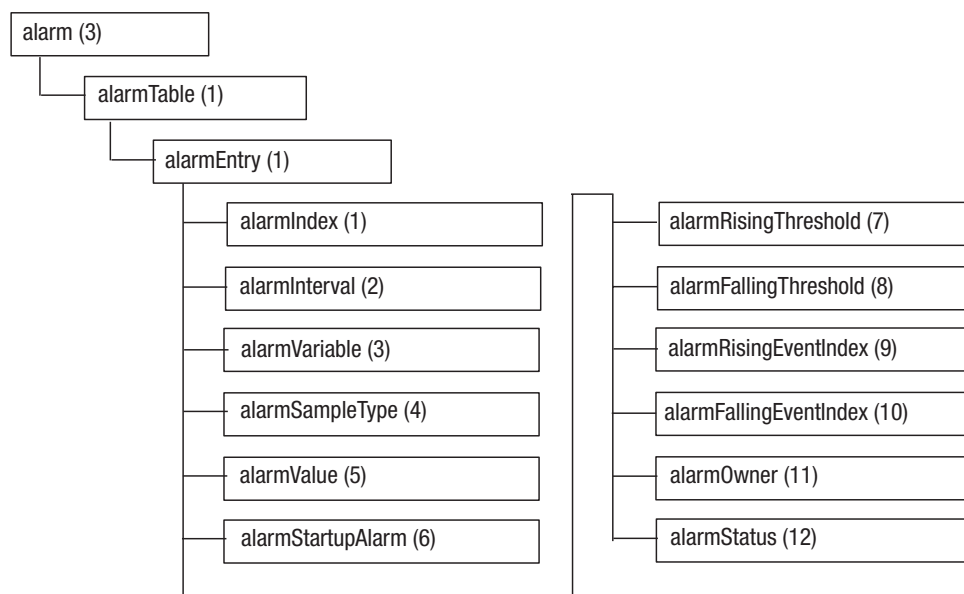


Figure 104. Structure of alarmTable

Table 436. Descriptions of alarmTable objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies a row in the alarm table; each such row specifies a sample at a particular interval for a particular object in the monitor's MIB.
(2)	Interval	Interval in seconds over which the data is sampled and compared with the rising and falling thresholds.
(3)	Variable	Object identifier of the particular variable in the RMON MIB to be sampled; only variables that resolve to the ASN.1 primitive type INTEGER may be sampled.
(4)	SampleType	Method of calculating the value to be compared to the thresholds:  absoluteValue(1) - Uses the absolute value of the variable for direct comparison with thresholds.  deltaValue(2) - Subtracts the value of variable in the last sample from the current value.
(5)	Value	Value of the statistic during the last sampling period.
(6)	StartupAlarm	Set to value of:  risingAlarm(1)  fallingAlarm(2)  risingOrFallingAlarm(3)
(7)	RisingThreshold	Rising threshold of the sampled statistic.
(8)	FallingThreshold	Falling threshold of the sampled statistic.
(9)	RisingEventIndex	Index of the eventEntry that is used when the rising threshold is crossed.



Table 436. Descriptions of alarmTable objects (continued)

Sub- OID	Object	Description
(10)	FallingEventIndex	Index of the eventEntry that is used when the falling threshold is crossed.
(11)	Owner	Owner of this entry.
(12)	Status	Status of this entry.

## host group

The host group contains counters for various types of traffic to and from hosts attached to the sub network. It discovers new hosts on the network by keeping a list of source and destination MAC addresses seen in good packets, maintaining, for each such host, a set of statistics.

For each interface specified by a row in the hostControlTable there is a row in the hostTable for each MAC address discovered on the interface. The two data tables contain the same information organized in two different ways

## hostControlTable

hostControlTable specifies for which interfaces the functions are performed. The objects in this table are shown in Figure 105 and listed in Table 437.

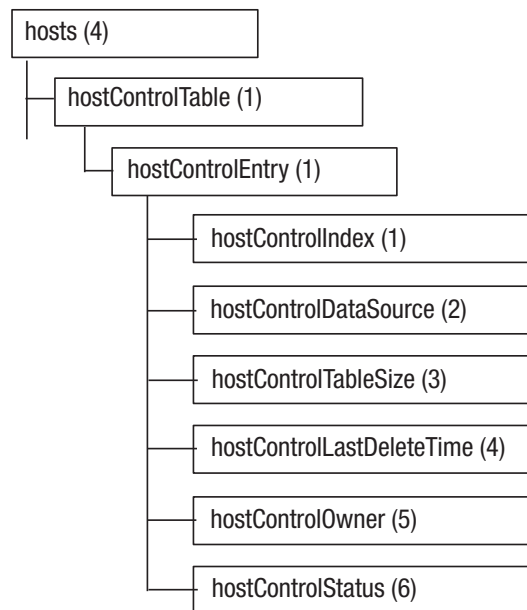


Figure 105. Structure of hostControlTable

Table 437. Descriptions of hostControlTable objects

Sub- OID	Object	Description
(1)	Index	Index that uniquely identifies a row in the hostControlTable.

Table 437. Descriptions of hostControlTable objects (continued)

Sub- OID	Object	Description
(2)	DataSource	Identifies the interface, and hence the subnetwork, that is the source of data for this row.
(3)	TableSize	Number of rows in the hostControlTable that are associated with this row.
(4)	LastDeleteTime	Value of sysUpTime corresponding to the last time an entry was deleted from the portion of the hostControlTable associated with this row; if no deletions have occurred, this value will be zero.
(5)	Owner	Owner of this entry.
(6)	Status	Status of this entry.

## hostTable

For each interface specified in the hostControlTable, the hostTable contains one row for each MAC (medium access control) address discovered on that interface. The objects in hostTable are shown in Figure 106 and listed in Table 438 on page 553.

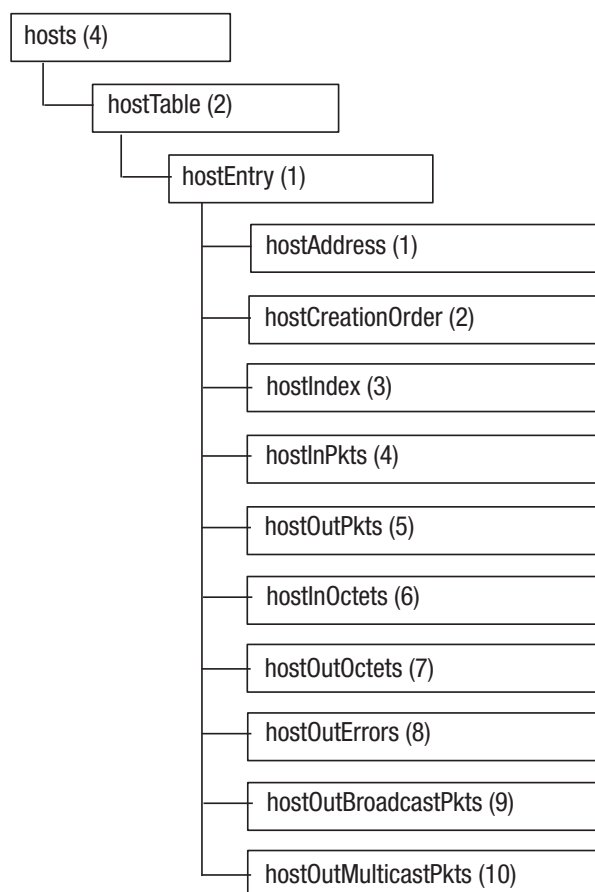


Figure 106. Structure of hostTable

Table 438. Description of hostTable objects

Sub- OID	Object	Description
(1)	Address	MAC address of this host.
(2)	CreationOrder	Index that defines the relative ordering of the creation time of the hosts captured for a particular hostControlEntry; value can be 1..N where N is the value of the associated hostControlTableSize; entries added earlier have lower index values.
(3)	Index	Set of collected host statistics of which this entry is a part.
(4)	InPkts	Number of good packets delivered to this address.
(5)	OutPkts	Number of packets, including bad packets, transmitted by this address.
(6)	InOctets	Number of octets transmitted to this address, including octets in bad packets.
(7)	OutOctets	Number of octets transmitted by this address, including octets in bad packets.
(8)	OutErrors	Number of bad packets transmitted by this address.
(9)	OutBroadcastPkts	Number of good broadcast packets transmitted by this address.
(10)	OutMulticastPkts	Number of good multicast packets transmitted by this address.

## hostTimeTable

hostTimeTable contains exactly the same information, row by row, as the previous table, but the hosts are sorted by time-of-creation order rather than by host MAC Address. This table has two important uses:

- Fast download of this potentially large table, since its size is known
- Efficient discovery by the management station of new entries in the table, since they will be added at the end of the current table.

The objects in hostTimeTable are shown in Figure 107 on page 554 and listed in Table 439 on page 554.

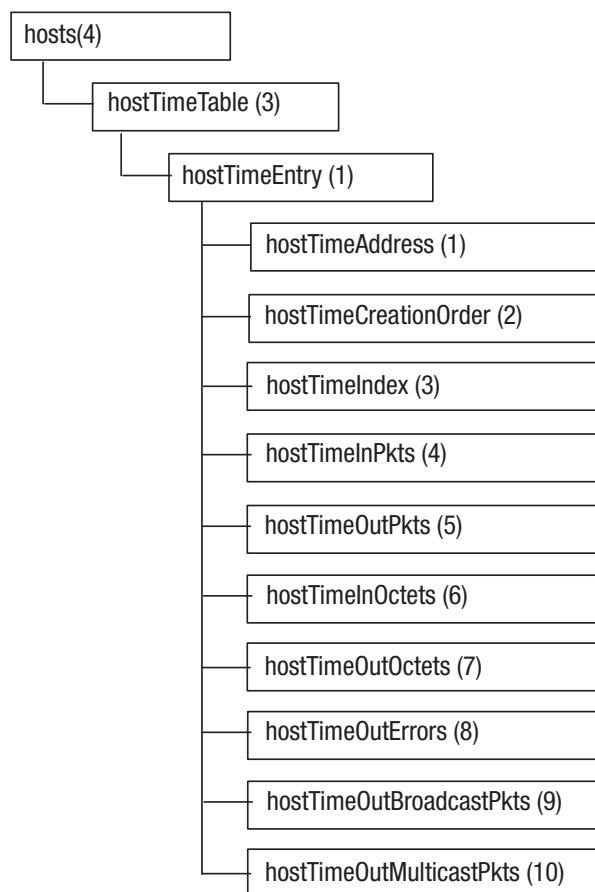


Figure 107. Structure of *hostTimeTable*

Table 439. Descriptions of *hostTimeTable* objects

Sub- OID	Object	Description
(1)	Address	MAC address of this host
(2)	CreationOrder	Index that defines the relative ordering of the creation time of the hosts captured for a particular <i>hostControlEntry</i> .
(3)	Index	Set of collected host statistics of which this entry is a part.
(4)	InPkts	Number of good packets delivered to this address.
(5)	OutPkts	Number of packets, including bad packets, transmitted by this address.
(6)	InOctets	Number of octets transmitted to this address, including octets in bad packets.
(7)	OutOctets	Number of octets transmitted by this address, including octets in bad packets.
(8)	OutErrors	Number of bad packets transmitted by this address.
(9)	OutBroadcastPkts	Number of good broadcast packets transmitted by this address.
(10)	OutMulticastPkts	Number of good multicast packets transmitted by this address.

## hostControl2Table

hostControl2Table is part of the RMON extension and gives additional information of dropped frames not counted in the statsDropEvents. The objects in hostControl2Table are shown in Figure 108 and listed in Table 440.

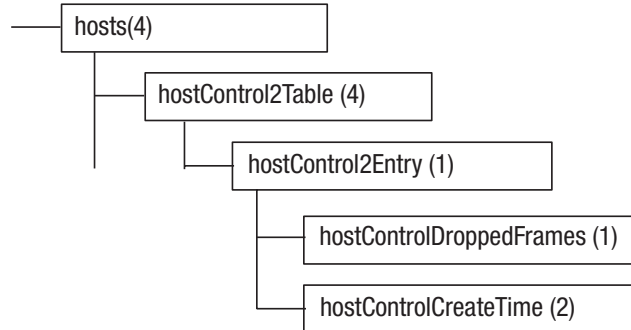


Figure 108. Structure of hostControl2Table

Table 440. Descriptions of hostControl2Table objects

Sub- OID	Object	Description
(1)	DroppedFrames	Number of frames that were received by the probe and therefore not accounted for in statsDropEvents, but which the probe chose not to count, for some reason.
(2)	CreateTime	Value of sysUpTime when this control entry was last activated.

---

## hostTopN group

The hostTopN group maintains statistics that describe the hosts that top an ordered list based on some parameter in the host table.

These statistics, derived from data in the host group, are organized into reports where a report is the set of statistics for one host group object on one interface, or sub-network, collected during one sampling period. Each report contains the results for one variable and that variable represents the amount of change in a host group object over the sampling interval; thus, these statistics are rate based.

## hostTopNControlTable

Rows in hostTopNControlTable define a top-N report prepared for an interface or sub-network. The objects in this table are shown in Figure 109 on page 556 and listed in Table 441 on page 556.

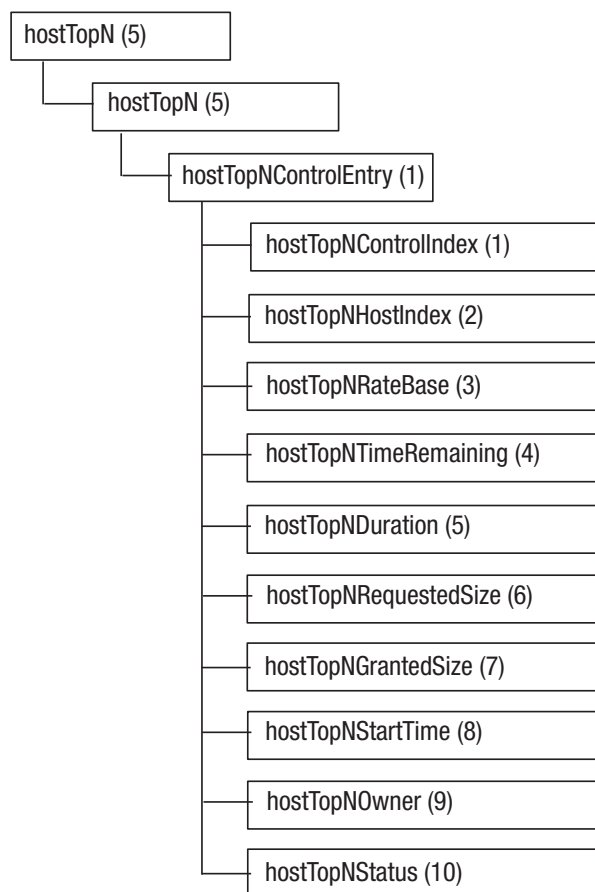


Figure 109. Structure of *hostTopNControlTable*

Table 441. Descriptions of *hostTopNControlTable* objects

Sub- OID	Object	Description
(1)	ControlIndex	Index that uniquely identifies a row in the <i>hostControlTable</i> and so defines one top-N report prepared for one interface.
(2)	HostIndex	Represents host table entry that matches a value of <i>hostControlIndex</i> and <i>hostIndex</i> and therefore specifies a particular sub-network.
(3)	RateBase	Specifies one of seven variables from <i>hostTable</i> and so defines which parameter to assess in the top-N report.
(4)	TimeRemaining	Number of seconds left in the sampling interval for the report currently being generated; when this entry decrements to zero, the report becomes available.
(5)	Duration	Sampling interval in seconds for this report.
(6)	RequestedSize	Maximum number of hosts requested for the top-N table for this report.
(7)	GrantedSize	Maximum number of hosts in the top-N table for this report.
(8)	StartTime	Value of <i>sysUpTime</i> when this top-N report was last started.
(9)	Owner	Owner of this entry.
(10)	Status	Status of this entry.

## hostTopNTable

hostTopNTable contains the report data, that is, the set of statistics for one host group object on one interface or sub network collected during one sampling interval. The objects in this table are shown in Figure 110 and listed in Table 442.

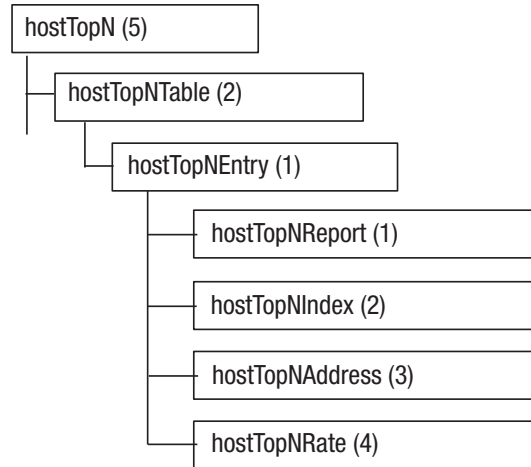


Figure 110. Structure of hostTopNTable

Table 442. Descriptions of hostTopNTable objects

Sub- OID	Object	Description
(1)	Report	Identifies the top-N report of which this entry is a part.
(2)	Index	Index that uniquely identifies a row among all data rows associated with this report.
(3)	Address	MAC address of this host.
(4)	Rate	Amount of change in the selected variable during this sampling interval; the selected variable is this host's instance of the object selected by hostTopNRateBase.

---

## matrix group

The matrix group shows error and utilization information about the traffic between pairs of hosts on a sub-network.

The data is stored in matrix form, so the operator can retrieve information for any pair of network addresses. The data tables store statistics for a particular conversation between two addresses. As with the host group, the tables contain the same information organized in different ways.

## matrixControlTable

matrixControlTable controls the entries in the two matrix data tables. The objects in this table are shown in Figure 111 and listed in Table 443.

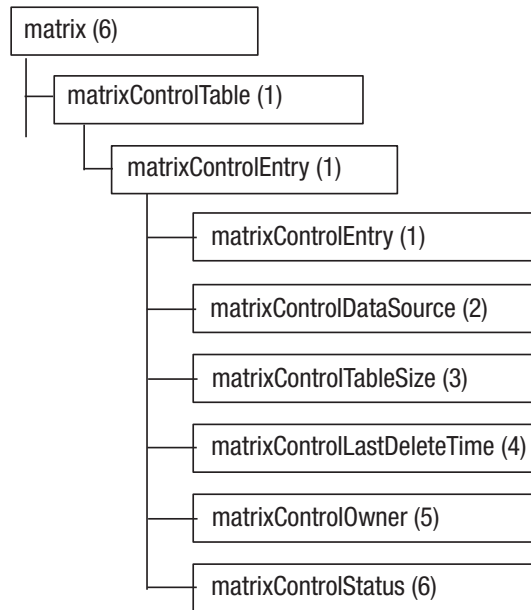


Figure 111. Structure of matrixControlTable

Table 443. Descriptions of matrixControlTable objects

Sub-OID	Object	Description
(1)	Index	Index that uniquely identifies a row in the matrixControlTable; each such entry defines a function that discovers conversations on a particular interface and places statistics about them in the matrixSDTable and the matrixDSTable on behalf of this matrixControlEntry.
(2)	DataSource	Identifies the sub network that is the source of the data in this row.
(3)	TableSize	Number of rows in the matrixSDTable that are associated with this row; this object is read-only and is set by the monitor.
(4)	LastDeleteTime	Value of sysUpTime corresponding to the last time that an entry was deleted from this portion of the matrixSDTable and the proportion of the matrixSDtable associated with this row.
(5)	Owner	Owner of this entry.
(6)	Status	Status of this entry.



## matrixSDTable

matrixSDTable stores statistics from a particular source host to a number of destinations using a list of traffic matrix entries indexed, within the source MAC address, by the destination MAC address. The objects in this table are shown in Figure 112 and listed in Table 444.

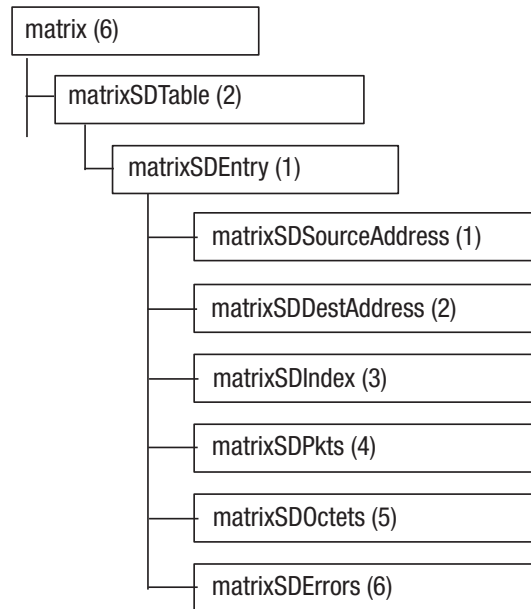


Figure 112. Structure of MatrixSDTable

Table 444. Structure of MatrixSDTable

Sub- OID	Object	Description
(1)	SourceAddress	Source MAC address.
(2)	DestAddress	Destination MAC address.
(3)	Index	Set of collected matrix statistics of which this row is a part.
(4)	Pkts	Number of packets transmitted from this source address to this destination address, including bad packets.
(5)	Octets	Number of octets contained in all packets transmitted from this source address to this destination address.
(6)	Errors	Number of bad packets transmitted from this source address to this destination address.

## matrixDSTable

matrixDSTable stores the same information as the matrixSDTable but the traffic matrix entries are indexed, within matrixDSIndex, by destination MAC address and then by the source MAC address. The objects in this table are shown in Figure 113 and listed in Table 445.

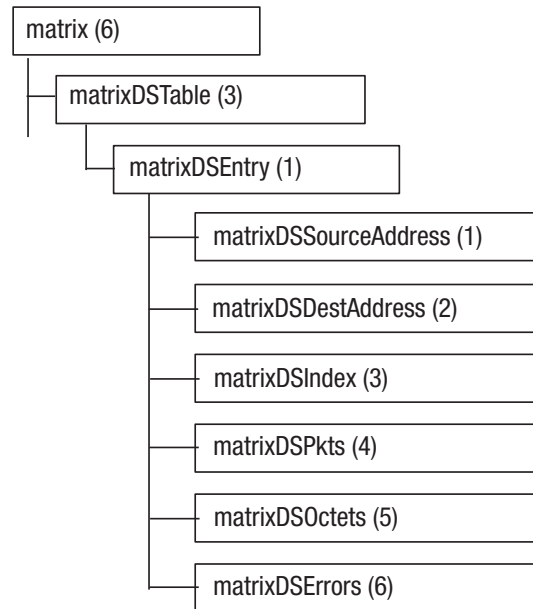


Figure 113. Structure of matrixDSTable

Table 445. Descriptions of matrixDSTable objects

Sub-OID	Object	Description
(1)	SourceAddress	Source MAC address.
(2)	DestAddress	Destination MAC address.
(3)	Index	Set of collected matrix statistics of which this row is a part.
(4)	Pkts	Number of packets transmitted from this destination address to this source address, including bad packets.
(5)	Octets	Number of octets contained in all packets transmitted from this destination address to this source address.
(6)	Errors	Number of bad packets transmitted from this destination address to this source address.

## matrixControl2Table

matrixControl2Table is part of the RMON extension and gives additional information of dropped frames not counted in the statsDropEvents. The objects in this table are shown in Figure 114 and listed in Table 446.

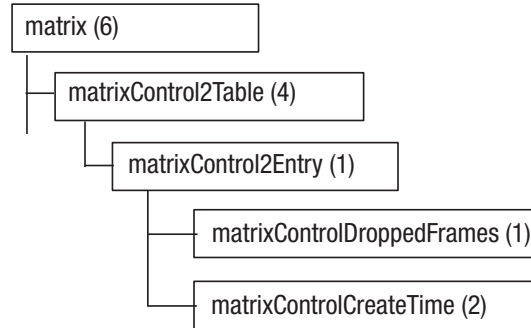


Figure 114. Structure of MatrixControl2Table

Table 446. Descriptions of MatrixControl2Table objects

Sub- OID	Object	Description
(1)	DroppedFrames	Total number of frames that were received by the probe and therefore not accounted for in the statsDropEvents, but which the probe chose not to count for some reason.
(2)	CreateTime	Value of sysUpTime when this control entry was last activated.

---

## filter group

The filter group allows the monitor to observe packets that match an arbitrary filter expression.

The packets that match the filter expression form a logical data and event stream or channel. The monitor may capture all packets that pass the filter or simply record statistics based on such packets. Each filter associated with a channel is ORed with the others. Within a filter, any bits checked in the data and status fields are ANDed with respect to other bits in the same filter. The NotMask also allows for checking for inequality and the channelAcceptType object allows for inversion of the whole equation. This group consists of two tables: a filterTable that specifies the filters to be applied and a channelTable that defines a channel defined by one or more filters.

## filterTable

filterTable provides a list of packet filter entries. The objects in this table are shown in Figure 115 on page 562 and listed in Table 447 on page 562.

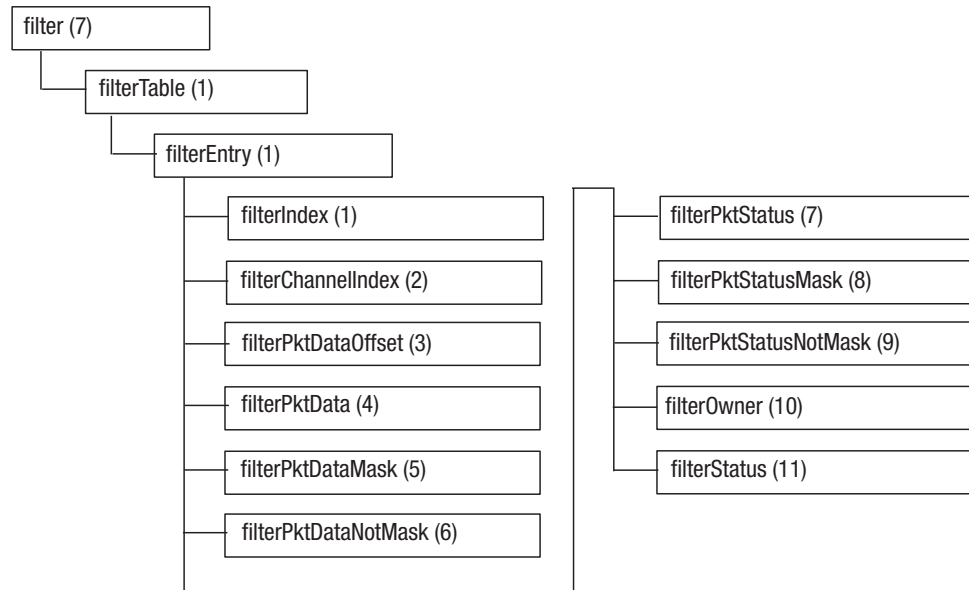


Figure 115. Structure of filterTable

Table 447. Descriptions of filterTable objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies one row in the filterTable; each row defines one data filter and one status filter that is applied to every packet received on an interface.
(2)	ChannelIndex	Identifies the channel of which this filter is a part.
(3)	PktDataOffset	Offset from the beginning of a packet where a match of the packet data will be attempted.
(4)	PktData	Data that is to be matched with the input packet. If the packet is too short, it will fail to match; a zero length filter will pass all packets.
(5)	PktDataMask	Mask that is applied to the match process.
(6)	PktDataNotMask	Inversion mask that is applied to the match process; for the purposes of the matching algorithm, if the associated filterPktData object is longer than this mask, this mask is conceptually extended with '0' bits until it reaches the length of the filterPktData object.
(7)	PktStatus	Status that is to be matched with the input packet.
(8)	PktStatusMask	Mask that is applied to the status match process.
(9)	PktStatusNotMask	Inversion mask that is applied to the status match process.
(10)	Owner	Owner of this entry.
(11)	Status	Status of this entry.

## channelTable

The objects in channelTable are shown in Figure 116 and listed in Table 448.

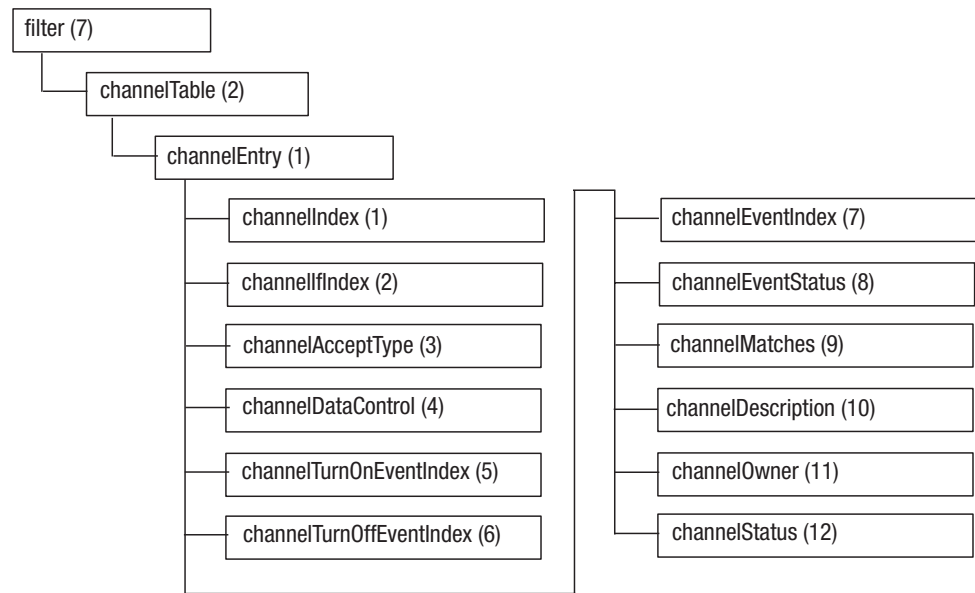


Figure 116. Structure of channelTable

Table 448. Description of channelTable objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies one row in the channelTable, thus defining a logical data and event stream.
(2)	IfIndex	Uniquely identifies the monitor interface to which the associated filters are applied to allow data into this channel.
(3)	AcceptType	Controls the action of the filters associated with this channel:  acceptMatched(1) - Packets will be accepted only if they pass both the packet data and packet status matches of at least one of the associated filters.  acceptFailed(2) - Packets are accepted only if they fail either the packet data match or the packet status match of every associated filter.
(4)	DataControl	Controls the flow of data through this channel:  on(1) - Data, status and events will flow through this channel.  off(2) - Data, status and events will not flow through this channel.
(5)	TurnOnEventIndex	Identifies the value of the particular event that is configured to toggle the associated channelDataControl from off to on when the event is generated.

Table 448. Description of channelTable objects (continued)

Sub- OID	Object	Description
(6)	TurnOffEventIndex	Identifies the particular event that is configured to turn the associated channelDataControl from on to off when the event is generated.
(7)	EventIndex	Identifies the event that is configured to be generated when the associated channelDataControl is on and a packet is matched; if no event is intended for this channel, it must be set to zero.
(8)	EventStatus	Event status of this channel; options are:  eventReady(1)  eventFired(2)  eventAlwaysReady(3)
(9)	Matches	Counter that records the number of packets matched; it is updated even when channelControlData is set to off.
(10)	Description	Comment or textual description of the channel.
(11)	Owner	Owner of this entry.
(12)	Status	Status of this entry.

## channel2Table

channel2Table is part of the RMON extension and gives additional information of dropped frames not counted in the statsDropEvents. The objects in this table are shown in Figure 117 and listed in Table 449.

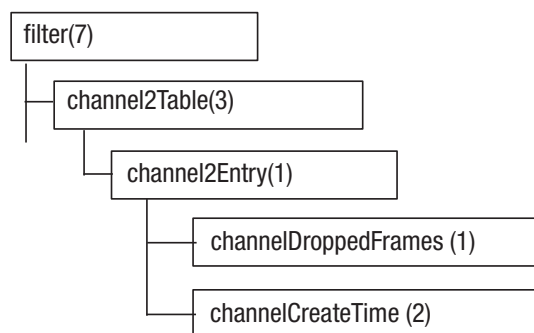


Figure 117. Structure of channel2Table

Table 449. Descriptions of channel2Table objects

Sub- OID	Object	Description
(1)	DroppedFrames	Number of frames that were received by the probe and therefore not accounted for in the statsDropEvents, but for which the probe chose not to count for some reason.
(2)	CreateTime	Value of sysUpTime when this control entry was last activated.

## filter2Table

filter2Table is part of the RMON extension and gives additional information regarding the filtering algorithm. The objects in this table are shown in Figure 118 and listed in Table 450.

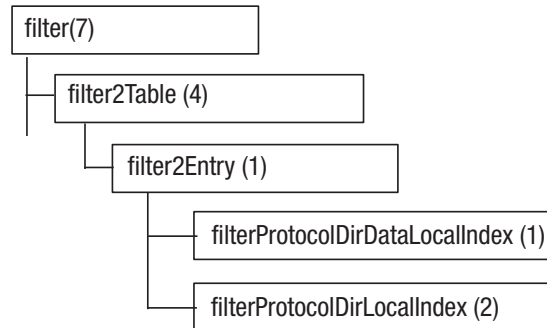


Figure 118. Structure of filter2Table

Table 450. Descriptions of filter2Table objects

Sub- OID	Object	Description
(1)	DataLocalIndex	If the packet matches this protocol directory entry, perform the regular filter algorithm as if the beginning of this named protocol is the beginning of the packet, potentially applying the filterPktDataOffset value to move further into the packet.
(2)	LocalIndex	Discard the packet if the packet does not match this protocol directory entry.

---

## capture group

The capture group governs how data is sent to a management console. It allows packets to be captured upon a filter match and thus allows establishment of a buffering scheme for the packets captured from a particular channel.

## bufferControlTable

bufferControlTable specifies the details of the buffering function. The objects in this table are shown in Figure 119 on page 566 and listed in Table 451 on page 566.

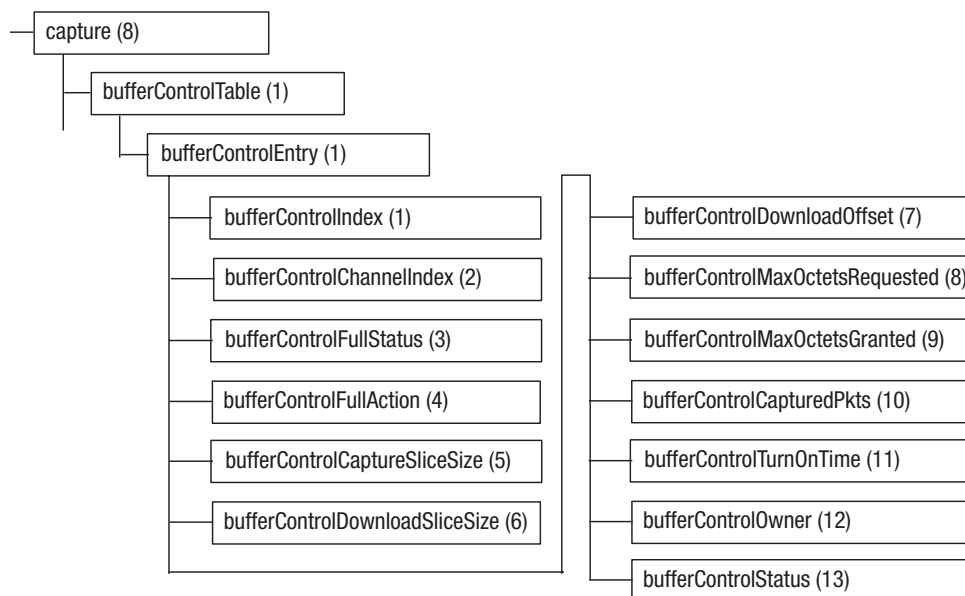


Figure 119. Structure of bufferControlTable

Table 451. Descriptions of bufferControlTable objects

Sub-OID	Object	Description
(1)	Index	Integer that uniquely identifies a row in the bufferControlTable.
(2)	ChannelIndex	Identifies the channel that is the source of packets for this row.
(3)	FullStatus	Shows whether the buffer has room to accept new packets; if this takes on the value spaceAvailable(1), the buffer has room to accept more packets; if it is full(2), its meaning depends on the value of bufferControlFullAction.
(4)	FullAction	Controls action of the buffer when it reaches full status; if value is lockWhenFull(1) the buffer will not accept more packets; if value is wrapWhenFull(2), the buffer acts as a circular buffer, deleting old packets as new ones arrive.
(5)	CaptureSliceSize	Max number of octets that will be saved in this buffer; if set to 0, buffer will save as many as possible. Default - 100.
(6)	DownloadSliceSize	Maximum number of octets of each packet in this buffer that will be returned in a single SNMP retrieval of this packet.
(7)	DownloadOffset	Offset of the first octet in each packet in this buffer that will be returned in a single SNMP retrieval of that packet.
(8)	MaxOctetsRequested	Requested buffer size in octets; if set to -1, the buffer will save as many octets as is possible.
(9)	MaxOctetsGranted	Granted buffer size in octets.
(10)	CapturedPkts	Number of packets currently in this capture buffer.
(11)	TurnOnTime	Value of sysUpTime when this buffer first turned on.
(12)	Owner	Owner of this entry.



Table 451. Descriptions of *bufferControlTable* objects (continued)

Sub- OID	Object	Description
(13)	Status	Status of this entry.

## captureBufferTable

captureBufferTable holds the actual buffered packets. The objects in this table are shown in Figure 120 and listed in Table 452.

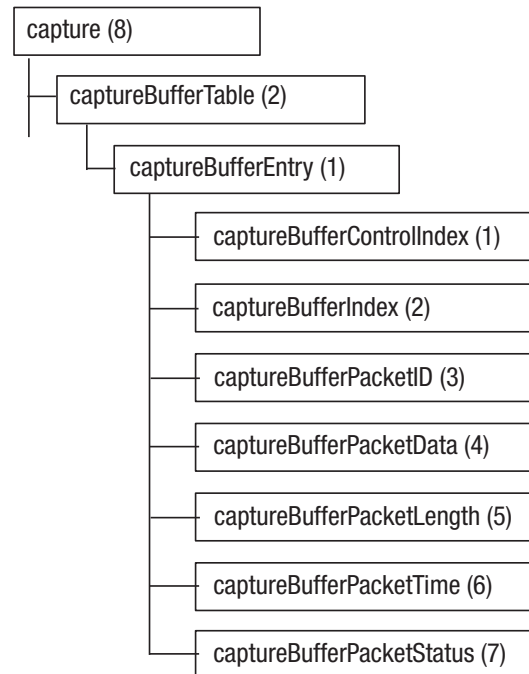


Figure 120. Structure of *captureBufferTable*

Table 452. Descriptions of *captureBufferTable* objects

Sub- OID	Object	Description
(1)	ControlIndex	Index to the buffer with which this packet is associated.
(2)	Index	Index that uniquely identifies this particular packet among all packets associated with the same buffer.
(3)	PacketID	Index that describes the order of packets that are received on a particular interface.
(4)	PacketData	Actual packet data stored for this row.
(5)	PacketLength	Actual length of the packet as received off the wire.
(6)	PacketTime	Number of milliseconds that had passed from the time the buffer was initially turned on to the time this packet was captured.
(7)	PacketStatus	Value that indicates the error status of this packet.

---

## event group

The event group controls the generation and notification of events generated by the RMON probe.

Each event entry is fired by an associated condition located elsewhere in the MIB. An event entry can also be associated with a function elsewhere in the MIB that will be executed when the event is generated. Each event entry may optionally specify that a log entry be created on its behalf whenever the event occurs.

### eventTable

eventTable is a list defining the events to be generated. The objects in this table are shown in Figure 121 and listed in Table 453.

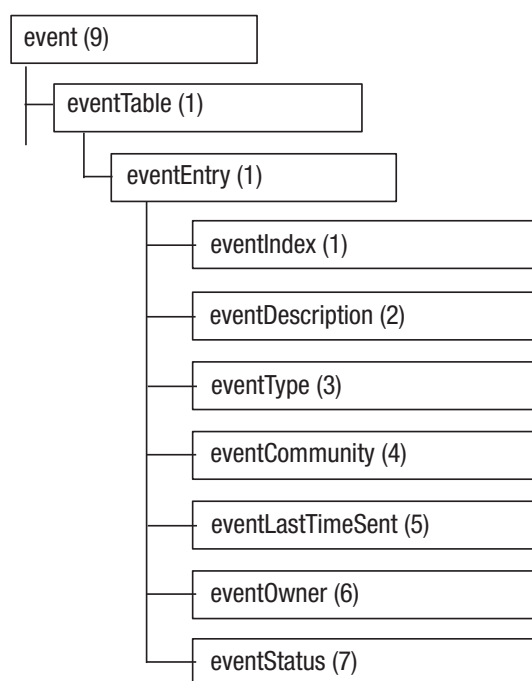


Figure 121. Structure of eventTable

Table 453. Descriptions of eventTable objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies a row on the eventTable.
(2)	Description	Textual description of this event.
(3)	Type	Value of:  none(1)  log(2)  SNMP-trap(3)  log-and-trap(4)

Table 453. Descriptions of eventTable objects (continued)

Sub- OID	Object	Description
(4)	Community	Specifies the community of management stations to receive the trap if an SNMP trap is to be sent.
(5)	LastTimeSent	Value of sysUpTime when this eventEntry last generated an event.
(6)	Owner	Owner of this entry.
(7)	Status	Status of this entry.

## logTable

logTable keeps a log of a particular event's generation. The objects in this table are shown in Figure 122 and listed in Table 454.

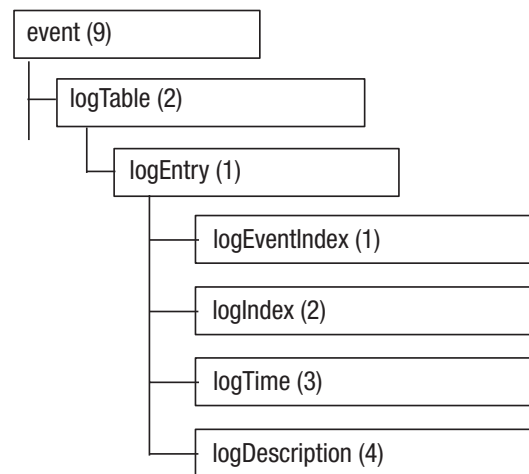


Figure 122. Structure of logTable

Table 454. Descriptions of logTable Objects

Sub- OID	Object	Description
(1)	EventIndex	Identifies the event that generated this log entry.
(2)	Index	A unique index that uniquely identifies this particular log entry among all entries associated with the same event type.
(3)	Time	Value of sysUpTime when this event was generated.
(4)	Description	Implementation-dependent description of the event that activated this log entry.

---

## tokenRing group

The tokenRing group is part of the RMON extension and maintains statistics and configuration information for token ring sub networks.

The group includes MAC-level statistics, promiscuous network statistics as well as Ring Station group, configuration and source routing.

### ringStationControlTable

ringStationControlTable is part of the RMON extension and is a list of ringStation control entries that are each defined by a list of parameters that set up the discovery of stations on a particular interface and the collection of statistics about these stations. The objects in this table are shown in Figure 123 and listed in Table 455.

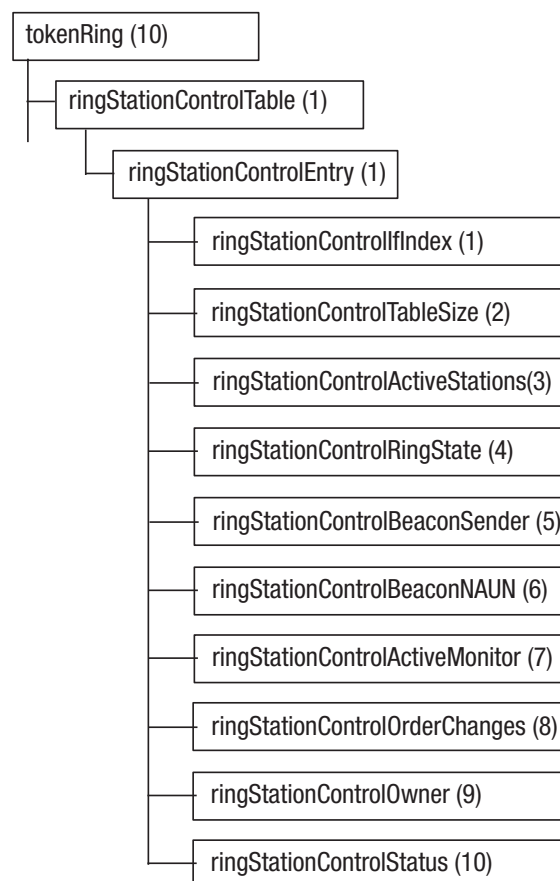


Figure 123. Structure of ringStationControlTable

Table 455. Descriptions of ringStationControlTable objects

Sub-OID	Object	Description
(1)	IfIndex	Uniquely identifies the interface on this remote network monitoring device from which ringStation data is collected.
(2)	TableSize	Number of ringStationEntries in the ringStationTable associated with this interface.

Table 455. Descriptions of ringStationControlTable objects (continued)

Sub- OID	Object	Description
(3)	ActiveStations	Number of active ringStationEntries monitored on this interface.
(4)	RingState	Current status of this ring; can take on the values normalOperation(1) ringPurgeState(2) claimTokenState(3) beaconFrameStreamingState(4) beaconBitStreamingState(5) beaconRingSignalLossState(6) beaconSetRecoveryModeState(7)
(5)	BeaconSender	MAC address of the sender of the last beacon frame.
(6)	BeaconNAUN	MAC address of the NAUN in the last beacon frame.
(7)	ActiveMonitor	MAC address of the active monitor on this sub network.
(8)	OrderChanges	Number of add and delete events in the ringStationOrderTable associated with this interface.
(9)	Owner	Owner of this entry.
(10)	Status	Status of this entry.

## ringStationTable

ringStationTable is part of the RMON extension and contains statistics and status information associated with each token ring station on the local ring. The objects in this table are shown in Figure 124 on page 572 and listed in Table 456 on page 572.

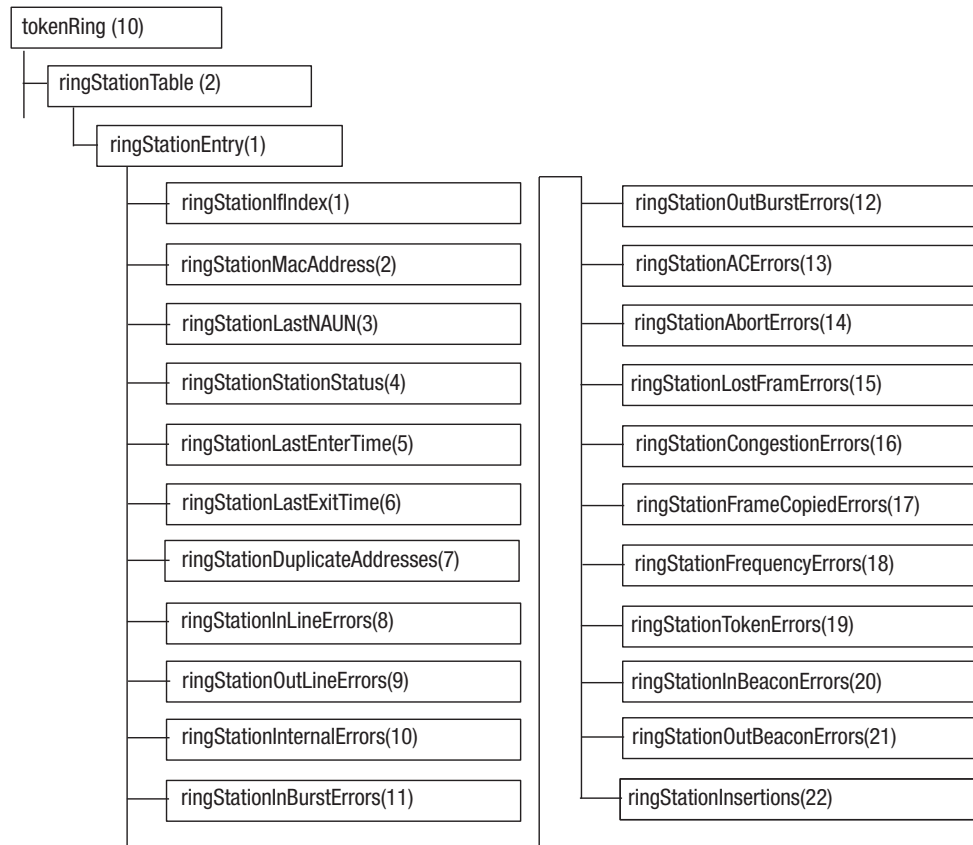


Figure 124. Structure of ringStationTable

Table 456. Descriptions of ringStationTable objects

Sub- OID	Object	Description
(1)	IfIndex	MAC address of this station.
(2)	MacAddress	MAC address of this station.
(3)	LastNAUN	MAC address of the last known NAUN of this station.
(4)	StationStatus	Indicates whether station is actively participating in logical ring.
(5)	LastEnterTime	Value of sysUpTime when this station last entered the logical ring.
(6)	LastExitTime	Value of sysUpTime when the monitor detected this station last exited logical ring.
(7)	DuplicateAddresses	Number of times this station received a duplicate address error.
(8)	InLineErrors	Number of line errors reported by this station in error reporting packets detected by the probe.
(9)	OutLineErrors	Number of line errors reported by this station in error reporting packets sent by the nearest active downstream neighbor of this station and detected by the probe.
(10)	InternalErrors	Number of adapter internal errors reported by this station in error reporting packets detected by the probe.

Table 456. Descriptions of ringStationTable objects (continued)

Sub- OID	Object	Description
(11)	InBurstErrors	Number of burst errors reported by this station in error reporting packets detected by the probe.
(12)	OutBurstErrors	Number of burst errors reported by this station in error reporting packets sent by the nearest active downstream neighbor of this station and detected by the probe.
(13)	ACErrors	Number of address copied errors in error reporting packets sent by the nearest active downstream neighbor of this station and detected by the probe.
(14)	AbortErrors	Number of abort delimiters reported by this station in error reporting packets detected.
(15)	LostFrameErrors	Number of abort delimiters reported by this station in error reporting packets.
(16)	CongestionErrors	Number of receive congestion errors reported by this station in error reporting packets detected by the probe.
(17)	FrameCopiedErrors	Number of frame copied errors reported by this station in error reporting packets detected by the probe.
(18)	FrequencyErrors	Number of frequency errors reported by this station in error reporting packets detected.
(19)	TokenErrors	Number of token errors reported by this station in error reporting packets detected by the probe.
(20)	InBeaconErrors	Number of beacon frames sent by this station and detected by the probe.
(21)	OutBeaconErrors	Number of beacon frames detected by the probe that name this station as the NAUN.
(22)	Insertions	Number of times the probe detected this station inserting onto the ring.

## ringStationOrderTable

ringStationOrderTable is part of the RMON extension and provides the order of the stations on monitored rings. The objects in this table are shown in Figure 125 on page 574 and listed in Table 457 on page 574.

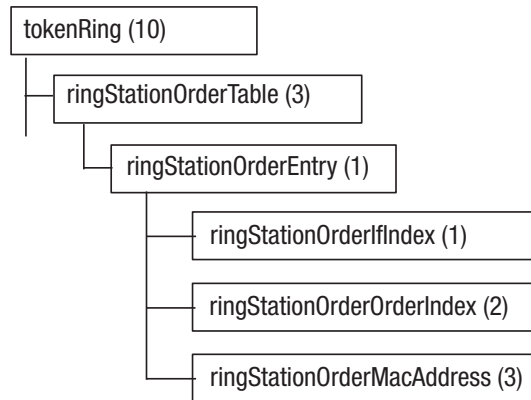


Figure 125. Structure of ringStationOrderTable

Table 457. Descriptions of ringStationOrderTable objects

Sub-OID	Object	Description
(1)	IfIndex	Uniquely identifies the interface, and hence the sub network, for this entry.
(2)	OrderIndex	Denotes the location of this station on this ring with respect to other stations on this ring (it is set to be one more than the # of hops downstream to this station from the monitor station).
(3)	MacAddress	MAC address of this station.

## ringStationConfigControlTable

ringStationConfigControlTable is part of the RMON extension and is a list of ring station control entries where each entry controls active management of individual stations by the probe. The objects in this table are shown in Figure 126 and listed in Table 458 on page 575.

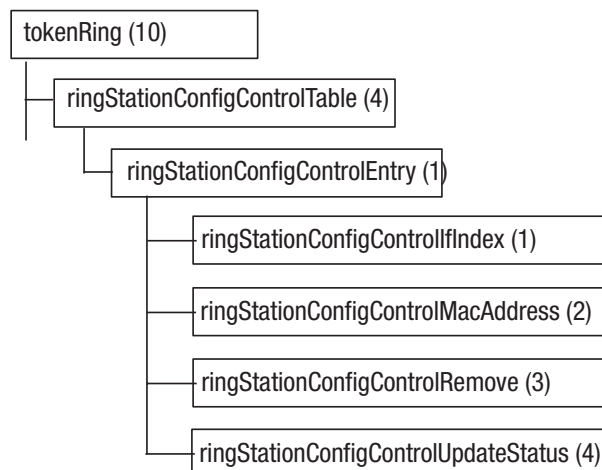


Figure 126. Structure of ringStationConfigControlTable



Table 458. Descriptions of ringStationConfigControlTable objects

Sub- OID	Object	Description
(1)	IfIndex	Identifies the interface, and hence the sub network.
(2)	MacAddress	MAC address of the station controlled by this entry.
(3)	Remove	Can be set to:  stable(1)  removing(2)
(4)	UpdateStatus	If set to updating(2) enables the configuration information associated with this entry to be updated.

## ringStationConfigTable

ringStationConfigTable is part of the RMON extension and manages token ring stations. The objects in this table are shown in Figure 127 and listed in Table 459.

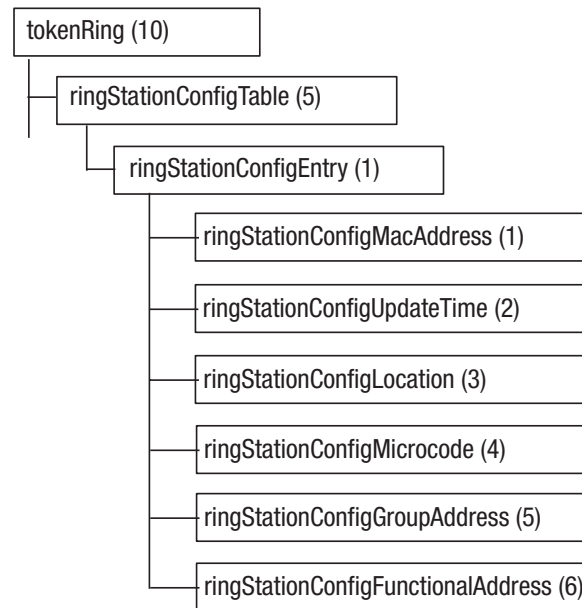


Figure 127. Structure of ringStationConfigTable

Table 459. Descriptions of ringStationConfigTable objects

Sub- OID	Object	Description
(1)	MacAddress	MAC address of this station.
(2)	UpdateTime	Value of sysUpTime when this configuration information was last updated.
(3)	Location	Assigned physical location of this station.
(4)	Microcode	Micro code version (or EC level) of this station.
(5)	GroupAddress	Low-order 4 octets of the group address recognized by this station.

Table 459. Descriptions of ringStationConfigTable objects (continued)

Sub- OID	Object	Description
(6)	FunctionalAddress	Functional addresses recognized by this station.

## sourceRoutingStatsTable

sourceRoutingStatsTable is part of the RMON extension and contains utilization statistics derived from source routing information optionally present in token ring packets. The objects in this table are shown in Figure 128 and listed in Table 460.



Figure 128. Structure of sourceRoutingStatsTable

Table 460. Descriptions of sourceRoutingStatsTable objects

Sub- OID	Object	Description
(1)	IfIndex	Uniquely identifies the interface on this remote network monitoring device on which source routing statistics will be detected.
(2)	RingNumber	Ring number of the ring monitored by this entry.
(3)	InFrames	Number of frames sent into this ring from another ring.

Table 460. Descriptions of sourceRoutingStatsTable objects (continued)

Sub- OID	Object	Description
(4)	OutFrames	Number of frames sent from this ring to another ring.
(5)	ThroughFrames	Number of frames sent from another ring, through this ring, to another ring.
(6)	AllRoutesBroadcastFrames	Number of good frames received that were all routes broadcast.
(7)	SingleRoutesBroadcastFrames	Number of good frames received that were single routes broadcast.
(8)	InOctets	Number of octets received into this ring from another ring.
(9)	OutOctets	Number of octets in good frames sent into another ring from this ring.
(10)	ThroughOctets	Number of octets in good frames sent from another ring, via this ring, to another ring.
(11)	AllRoutesBroadcastOctets	Number of octets received that were all routes broadcast.
(12)	SingleRoutesBroadcastOctets	Number of octets received that were single routes broadcast.
(13)	LocalLLCFrames	Number of frames received with no RIF (routing information field) and were not all routes broadcast frames.
(14)	1HopFrames	Number of frames received whose route had 1 hop, were not all routes broadcast frames, and whose source or destination were on this ring.
(15)	2HopsFrames	Number of frames received whose route had 2 hops, were not all routes broadcast frames, and whose source or destination were on this ring.
(16)	3HopsFrames	Number of frames received whose route had 3 hops, were not all routes broadcast frames, and whose source or destination were on this ring.
(17)	4HopsFrames	Number of frames received whose route had 4 hops, were not all routes broadcast frames, and whose source or destination were on this ring.
(18)	5HopsFrames	Number of frames received whose route had 5 hops, were not all routes broadcast frames, and whose source or destination were on this ring.
(19)	6HopsFrames	Number of frames received whose route had 6 hops, were not all routes broadcast frames, and whose source or destination were on this ring.
(20)	7HopsFrames	Number of frames received whose route had 7 hops, were not all routes broadcast frames, and whose source or destination were on this ring.
(21)	8HopsFrames	Number of frames received whose route had 8 hops, were not all routes broadcast frames, and whose source or destination were on this ring.

Table 460. Descriptions of sourceRoutingStatsTable objects (continued)

Sub- OID	Object	Description
(22)	MoreThan8HopsFrames	Number of frames received whose route had more than 8 hops, were not all routes broadcast frames, and whose source or destination was on this ring.
(23)	Owner	Owner of this entry.
(24)	Status	Status of this entry.

## ringStationControl2Table

ringStationControl2Table is part of the RMON extension and gives additional information of dropped frames not counted in the statsDropEvents. The objects in this table are shown in Figure 129 and listed in Table 461.

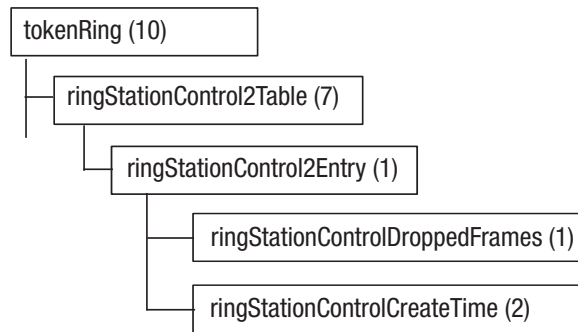


Figure 129. Structure of ringStationControl2Table

Table 461. Descriptions of ringStationControl2Table objects

Sub- OID	Object	Description
(1)	DroppedFrames	Number of frames that received by the probe and therefore not accounted for in the statsDropEvents, but which the probe chose not to count for some reason.
(2)	CreateTime	Value of sysUpTime when this control entry was last activated.

## sourceRoutingStats2Table

sourceRoutingStats2Table is part of the RMON extension and gives additional information of dropped frames not counted in the statsDropEvents. The objects in this table are shown in Figure 130 on page 579 and listed in Table 462 on page 579.

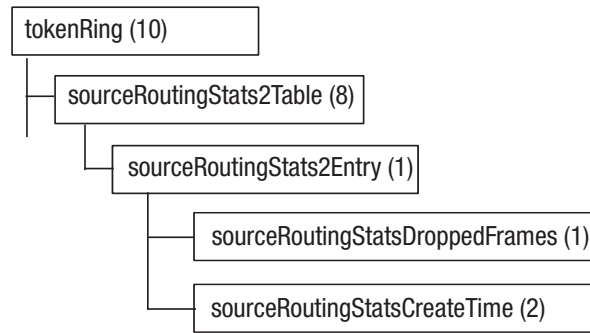


Figure 130. Structure of sourceRoutingStats2Table

Table 462. Descriptions of sourceRoutingStats2Table objects

Sub-OID	Object	Description
(1)	DroppedFrames	Number of frames received by the probe and therefore not accounted for in the statsDropEvents, but which the probe chose not to count for some reason.
(2)	CreateTime	Value of sysUpTime when this control entry was last activated.

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

### Buffer control table

The capture commands create rows in the buffer control table (bufferControlTable).

The general syntax of these commands is:

```
capture property=value
capture create [property=value ...]
capture reset
```

Table 463 lists the properties supported in these commands.

Table 463. Configuration command parameters - capture

Property	Type	Description	Sets MIB object
channel	int	The channel index that acts as the packet source.	ChannelIndex
fullaction	enum	The action taken on buffer full:  lock  wrap	FullAction

Table 463. Configuration command parameters - capture (continued)

Property	Type	Description	Sets MIB object
capsize	int	The capture slice size.	CaptureSliceSize
dlsize	int	The download slice size.	DownloadSliceSize
dloffset	int	The download offset.	DownloadOffset
bufsize	int	The capture buffer size.	MaxOctetsRequested

The `rmonCaptureMaxOctets` inivar sets the upper limit on the buffer size (in octets). If the requested buffer size specified in `bufferControlMaxOctetsRequested` exceeds this value, the size of the buffer granted is limited to the value of `rmonCaptureMaxOctets`. If the inivar is not set, the default maximum is 1000000 octets.

## Ethernet statistics table

The `etherstats` commands create rows in the ethernet statistics table (`etherStatsTable`).

The general syntax of these commands is:

```
etherstats property=value
etherstats create [property=value ...]
etherstats reset
```

Table 464 lists the properties supported in these commands.

Table 464. Configuration command parameters - etherstats

Property	Type	Description	Sets MIB object
datasource	OID	The interface monitored by the control row.	DataSource

## History control table

The history commands create rows in the history control table (`historyControlTable`).

The general syntax of these commands is:

```
history property=value
history create [property=value ...]
history reset
```

Table 465 lists the properties supported in these commands.

Table 465. Configuration command parameters - history

Property	Type	Description	Sets MIB object
datasource	OID	The interface monitored by the control row.	DataSource
buckets	int	The number of rows requested in the ethernet history table.	BucketsRequested
sample	int	The sample interval (in seconds).	Interval

## Filter table

The filter commands create rows in the filter table (filterTable).

The general syntax of these commands is:

```
filter property=value  
filter create [property=value ...]  
filter reset
```

Table 466 lists the properties supported in these commands.

Table 466. Configuration command parameters - filter

Property	Type	Description	Sets MIB object
channel	int	The channel index.	ChannelIndex
dataoffset	int	The data offset.	PktDataOffset
data	string (hex)	The packet match data.	PktData
datamask	string (hex)	The packet data mask.	PktDataMask
datanotmask	string (hex)	The packet data not-mask.	PktDataNotMask
status	int	The packet status.	PktStatus
statusmask	int	The packet status mask.	PktStatusMask
statusnotmask	int	The packet status not-mask.	PktStatusNotMask

## Host control table

The hosts commands create rows in the host control table (hostControlTable).

The general syntax of these commands is:

```
hosts property=value  
hosts create [property=value ...]  
hosts reset
```

Table 467 lists the properties supported in these commands.

Table 467. Configuration command parameters - hosts

Property	Type	Description	Sets MIB object
datasource	OID	The interface monitored by the control row.	DataSource

## TopN control table

The hosttopn commands create rows in the host TopN control table (hostTopNControlTable).

The general syntax of these commands is:

```
hosttopn property=value  
hosttopn create [property=value ...]  
hosttopn reset
```

Table 468 on page 582 lists the properties supported in these commands.

Table 468. Configuration command parameters - hosttopn

Property	Type	Description	Sets MIB object
hostindex	string	The host row index.	HostIndex
ratebase	enum	Selects the parameter evaluated in the topN data:  bcastPkts  inBytes  inPkts  mcastPkts  outBytes  outErrors  outPkts	RateBase
time	enum	The time remaining in the current sample interval.	TimeRemaining
size	int	The number of host report rows requested.	RequestedSize

## Matrix control table

The matrix commands create rows in the matrix control table (matrixControlTable).

The general syntax of these commands is:

```
matrix property=value
matrix create [property=value ...]
matrix reset
```

Table 469 lists the properties supported in these commands.

Table 469. Configuration command parameters - matrix

Property	Type	Description	Sets MIB object
datasource	OID	The interface monitored by the control row.	DataSource

## Channel table

The channel commands create rows in the channel table (channelTable).

The general syntax of these commands is:

```
channel property=value
channel create [property=value ...]
channel reset
```

Table 470 lists the properties supported in these commands.

Table 470. Configuration command parameters - channel

Property	Type	Description	Sets MIB object
ifindex	int	The index of the data source.	IfIndex



Table 470. Configuration command parameters - channel (continued)

Property	Type	Description	Sets MIB object
accept	enum	Controls the action of the filters associated with this channel:  failed  matched	AcceptType
datacontrol	enum	Data control:  on - Enables data collection  off - Suspends data collection	DataControl
onevent	int	The index of the control row activation event.	TurnOnEventIndex
offevent	int	The index of the control row deactivation event.	TurnOffEventIndex
event	int	The index of the event generated when a match occurs.	EventIndex
eventstatus	enum	Event flow control:  alwaysready  fired  ready	EventStatus
description	string	A description of the control row.	Description



## Appendix C. RMON2 MIB group

The RMON2 MIB extends the original RMON MIB to include groups to enable the RMON MIB to monitor protocol traffic above the MAC-level.

RMON2's capabilities mean that an RMON probe can monitor traffic on the basis of network-layer protocols and addresses, including the IP, thus the probe can see the traffic coming onto the LAN via routers, not just at the LAN to which it is attached. Secondly, RMON2's facility to decode and monitor application level traffic, such as email, file transfer and WWW protocols, means the probe can record to and from hosts for specific applications. Figure 131 shows the RMON extended structure.

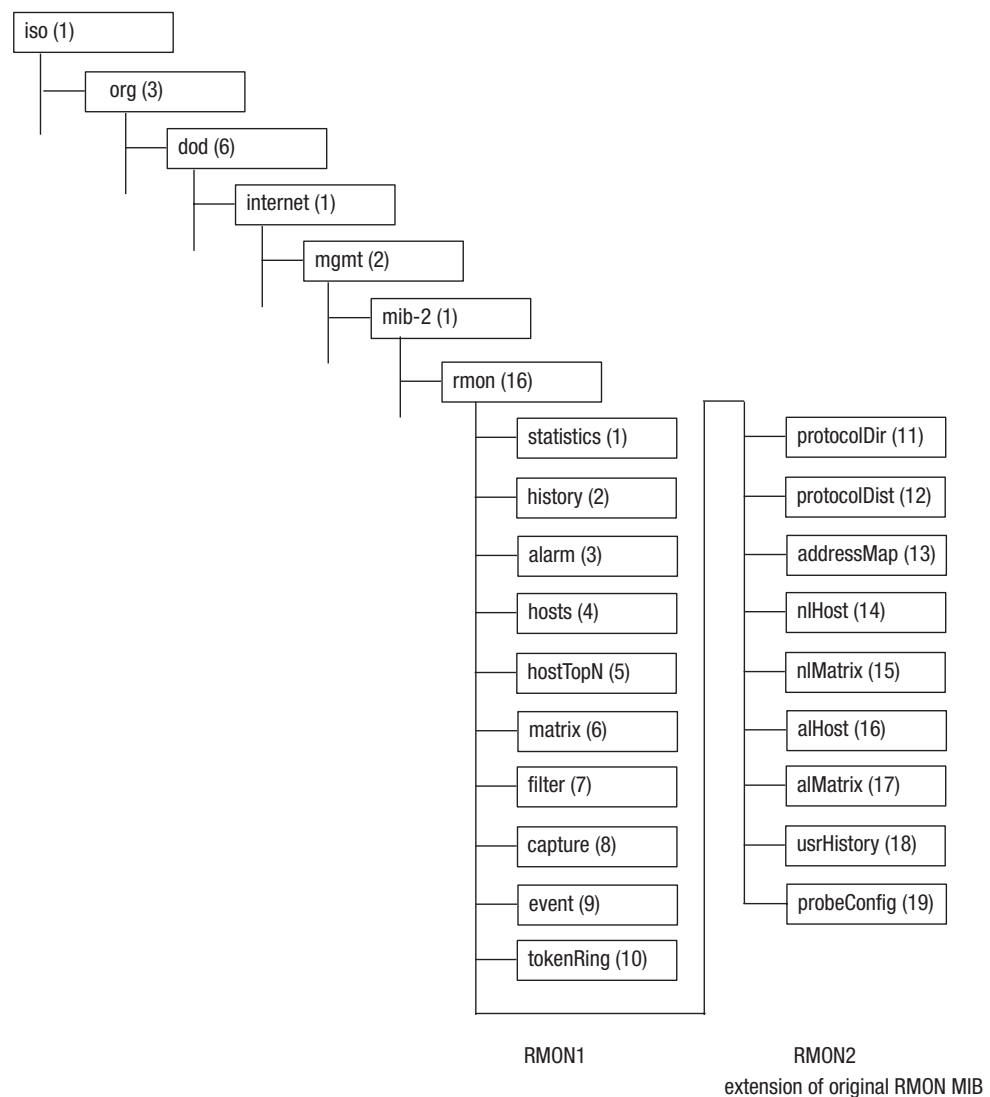


Figure 131. RMON2 MIB group structure

## protocolDir group

protocolDir is a master directory containing an inventory of information about all of the protocols that the probe can interpret. It lists the protocols that the probe is able to decode and count. These protocols represent different network-layer, transport-layer and higher-layer protocols. The probe should boot up with this table preconfigured with those protocols to be monitored.

Figure 132 shows the structure of the protocolDir group. Table 471 lists the objects in this group.

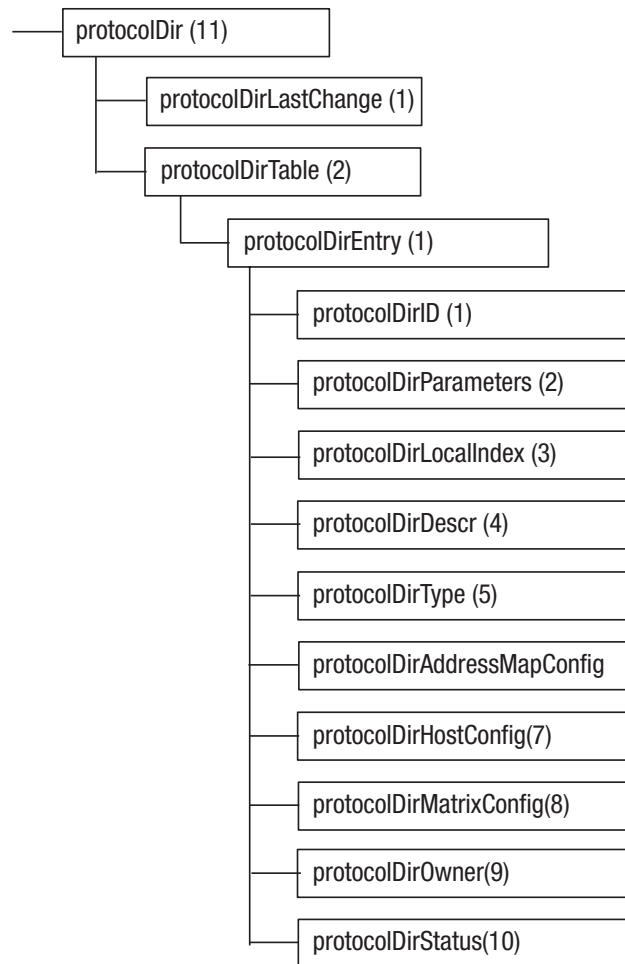


Figure 132. Structure of protocolDirTable

Table 471. Descriptions of protocolDirTable Objects

Sub- OID	Object	Description
(1)	ID	Contains a unique octet string for a specific protocol.
(2)	Parameters	Contains information about the probe's capability with respect to a particular protocol.  countsFragments (bit 0) - enables correct counting of encapsulated higher-layer protocols including those that fragment PDUs.

Table 471. Descriptions of protocolDirTable Objects (continued)

Sub- OID	Object	Description
(3)	LocalIndex	An arbitrary unique index number associated with this entry; the value for each supported protocol must remain constant at least from one re-initialization of the entity's NMS to the next.
(4)	Desc	Textual description of the protocol encapsulation.
(5)	Type	<p>Determines whether or not this protocol directory entry can be extended by the user by creating protocol directory entries, which are children of this protocol; it may take on the values:</p> <p>extensible(0) - can extend this table by creating entries that are children to this protocol</p> <p>addressRecognitionCapable(1) - if the probe can count packets and also recognize source and destination fields for finer-grained counting.</p>
(6)	AddressMapConfig	<p>Describes and configures the probe's support for address mapping for this protocol; it may take on the values:</p> <p>notSupported(1)</p> <p>supportedOff(2)</p> <p>supportedOn(3)</p>
(7)	HostConfig	<p>Describes and configures the probe's support for the network-layer and applications-layer host tables for this protocol; it may take on the following values with respect to the network-layer and the application-layer host table for this protocol:</p> <p>notSupported(1)</p> <p>supportedOff(2)</p> <p>supportedOn(3)</p>
(8)	MatrixConfig	<p>Describes and configures the probe's support for the network-layer and applications-layer matrix tables for this protocol; it may take on the following values with respect to the network-layer and the application-layer host matrix tables for this protocol:</p> <p>notSupported(1)</p> <p>supportedOff(2)</p> <p>supportedOn(3)</p>
(9)	Owner	Owner of this entry.
(10)	Status	Status of this entry.

## protocolDist group

The protocolDist group collects aggregate statistics on the distribution of traffic generated by each protocol, per LAN segment.

### protocolDistControlTable

protocolDistControlTable controls the collection of basic statistics for all supported protocols. The objects in this table are shown in Figure 133 and listed in Table 472.

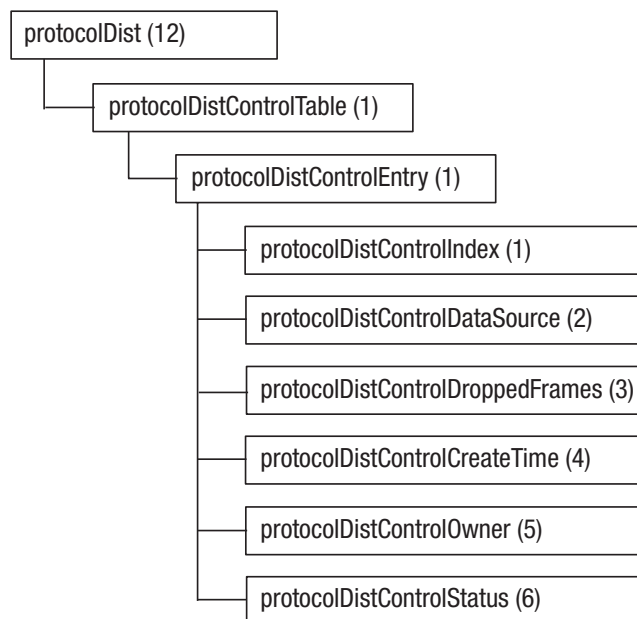


Figure 133. Structure of protocolDistTable

Table 472. Descriptions of protocolDistTable objects

Sub- OID	Object	Description
(1)	ControlIndex	Integer that uniquely identifies a row in the protocolDistControlTable.
(2)	ControlDataSource	Identifies the interface that is the source of data for this row.
(3)	ControlDroppedFrames	Number of received frames for this interface that the probe chose not to count; frames are not counted if probe is out of resources.
(4)	ControlCreateTime	Value of sysUpTime when this control entry was activated.
(5)	ControlOwner	Owner of this entry.
(6)	ControlStatus	Status of this entry.

## protocolDistStatsTable

protocolDistStatsTable contains an entry for every protocol in protocolDirTable that has been seen in at least one packet. The objects in this table are shown in Figure 134 and listed in Table 473.

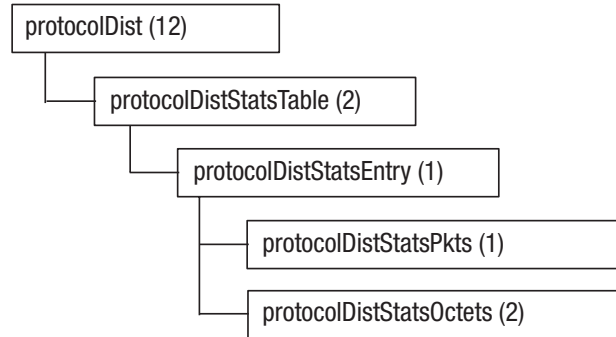


Figure 134. Structure of protocolDistStatsTable

Table 473. Descriptions of protocolDistStatsTable objects

Sub- OID	Object	Description
(1)	Pkts	Number of packets received without errors for this protocol.
(2)	Octets	Number of octets transmitted in good packets to this address since it was added to the n1HostTable.

---

## addressMap group

The addressMap group matches each netwGlork address to a specific MAC address and port on an attached device and the physical address on the sub network.

The objects in this group are shown in Figure 135 and listed in Table 474.

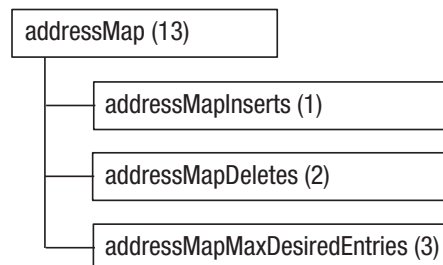


Figure 135. Structure of addressMap group

Table 474. Descriptions of addressMap objects

Sub- OID	Object	Description
(1)	Inserts	Number of times an address mapping entry has been inserted into the data table.

Table 474. Descriptions of addressMap objects (continued)

Sub- OID	Object	Description
(2)	Deletes	Number of times an address mapping entry has been deleted from the data table.
(3)	MaxDesiredEntries	Desired maximum number of entries in the address map table. <b>Note:</b> An entry of -1 denotes any number of entries.

## addressMapControlTable

addressMapControlTable controls the collection of network-layer address to physical address to interface mappings. Each entry in this table enables the discovery of addresses on a new interface and the placement of address mappings into the central addressMapTable. The objects in this table are shown in Figure 136 and listed in Table 475.

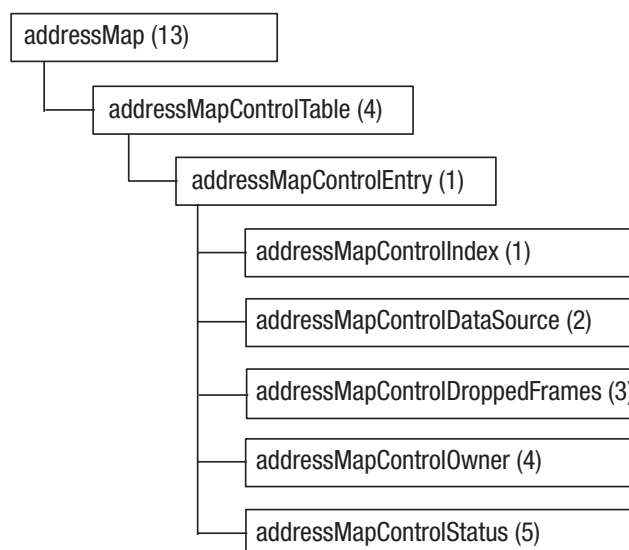


Figure 136. Structure of addressMapControlTable

Table 475. Descriptions of addressMapControlTable objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies a row in the addressMapControlTable.
(2)	DataSource	Identifies the interface that is the source of data for this row and that this row is configured to analyze.
(3)	DroppedFrames	Number of received frames for this interface that the probe chose not to count; it does not include packets that were not counted because they had MAC-layer address errors.
(4)	Owner	Owner of this entry.
(5)	Status	Status of this entry.



## addressMapTable

addressMapTable is a list of network-layer address to physical address interface mappings. The objects in this table are shown in Figure 137 and listed in Table 476.

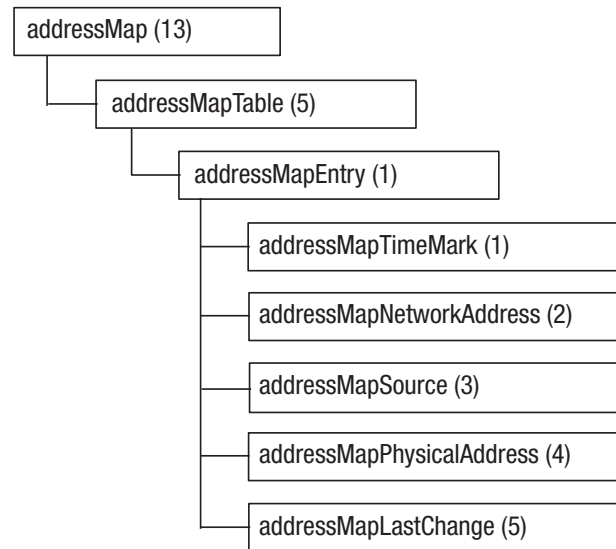


Figure 137. Structure of addressMapTable

Table 476. Descriptions of addressMapTable objects

Sub- OID	Object	Description
(1)	TimeMark	Time filter for this entry.
(2)	NetworkAddress	Network address for this entry.
(3)	Source	Last interface or repeater port on which the associated network address was seen.
(4)	PhysicalAddress	Last MAC address on which the associated network address was seen.
(5)	LastChange	Value of sysUpTime when this entry was most recently updated.

---

## nlHost group

The nlHost group monitors packets on traffic into and out of hosts on the basis of network-layer address. This allows the manager to look beyond the router to the connected hosts. It controls both the network and application-layer host tables.

The agent populates this table for all network-layer protocols in the protocol directory table whose value of protocolDirHostConfig is equal to supportedOn(3).

## hlHostControlTable

hlHostControlTable contains a list of higher-layer (above the MAC-layer) host table control entries. The objects in this table are shown in Figure 138 and listed in Table 477.

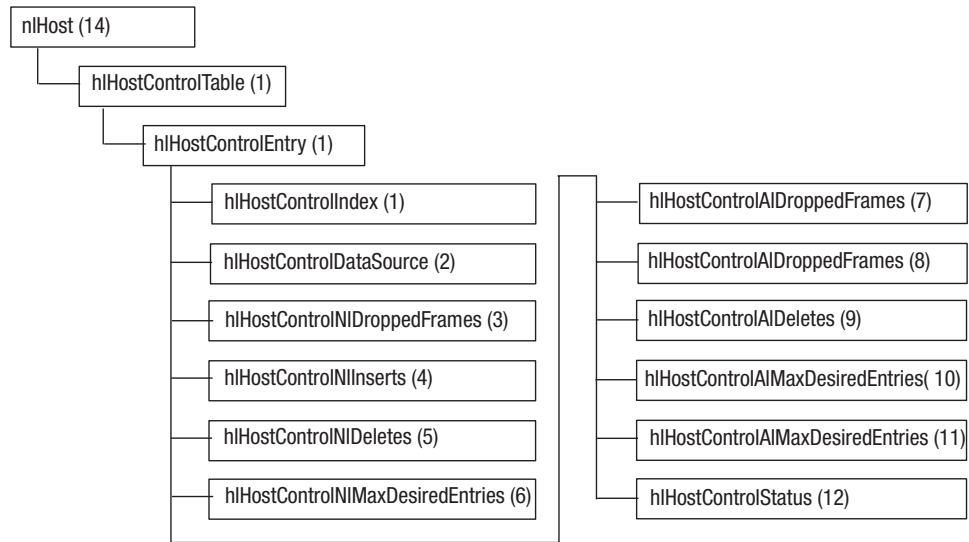


Figure 138. Structure of nlHostControlTable

Table 477. Descriptions of hlHostControlTable objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies a row in the nlHostControlTable; each such entry defines a function that discovers hosts on a particular interface and places statistics about them in the nlHostTable and, optionally, in the alHostTable on behalf of this nlHostControlEntry.
(2)	DataSource	Identifies the interface that is the source of the data for the data table entries defined by this row.
(3)	NIDroppedFrames	Number of received frames for this interface that the probe chose not to count for the associated nlHostEntries.
(4)	NlInserts	Number of times an nlHostEntry has been inserted into nlHostTable.
(5)	NlDeletes	Number of times an nlHostEntry has been deleted from nlHostTable.
(6)	NlMaxDesiredEntries	Desired maximum number of entries in the nlHost table.
(7)	AlDroppedFrames	Number of received frames for this interface that the probe chose not to count for the associated alHostEntries.
(8)	AlInserts	Number of times an alHostEntry has been inserted into alHostTable.
(9)	AlDeletes	Number of times an alHostEntry has been deleted from alHostTable.

Table 477. Descriptions of *nlHostControlTable* objects (continued)

Sub- OID	Object	Description
(10)	AlMaxDesiredEntries	Desired maximum number of entries in the <i>nlHostTable</i>
(11)	Owner	Owner of this entry.
(12)	Status	Status of this entry.

## nlHostTable

*nlHostTable* creates entries for all network-layer protocols in the protocol directory table whose value of *protocolDirNlMatrixConfig* is *supportedOn(3)*. The objects in this table are shown in Figure 139 and listed in Table 478.



Figure 139. Structure of *nlHostTable*

Table 478. Descriptions of *nlHostTable* objects

Sub- OID	Object	Description
(1)	TimeMark	Time filter for this entry.
(2)	Address	Network address for this entry.
(3)	InPkts	Number of error-free packets transmitted to this address since it was added to the table.
(4)	OutPkts	Number of error-free packets transmitted by this address since it was added to the table.

Table 478. Descriptions of nIHostTable objects (continued)

Sub- OID	Object	Description
(5)	InOctets	Number of octets transmitted to this address since it was added to the table.
(6)	OutOctets	Number of octets transmitted from this address since it was added to the table.
(7)	OutMacNonUnicastPkts	Number of packets transmitted by this address that were directed to the MAC broadcast address, or to any MAC multicast address, since this entry was added to the table.
(8)	CreateTime	Value of sysUpTime when this entry was created.

## nIMatrix group

The nIMatrix group deals with the collection of statistics on pairs of hosts based on the network-layer address. The data tables in this group collect statistics similar to those collected by the RMON1 matrix and hostTopN groups.

### hIMatrixControlTable

hIMatrixControlTable contains a list of higher level (non-MAC) matrix control entries. The objects in this table are shown in Figure 140 and listed in Table 479 on page 595.

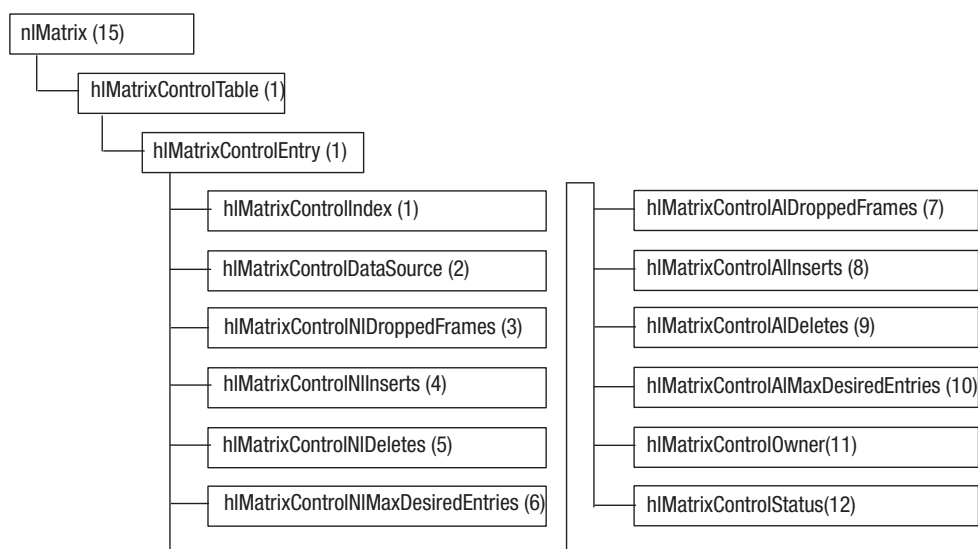


Figure 140. Structure of hIMatrixControlTable

Table 479. Descriptions of nIMatrixControlTable objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies a row in the nIMatrixControlTable and defines a function that discovers conversations on a particular interface; statistics about the conversations are placed in the nIMatrixSDTable and in the nIMatrixDSTable.
(2)	DataSource	Identifies the interface that is the source of the data for this rows data table entries.
(3)	NIDroppedFrames	Number of received frames for this interface that the probe chose not to count for this particular entry.
(4)	NIInserts	Number of times an nIMatrix entry has been inserted into the nIMatrix data tables.
(5)	NIDeletes	Number of times an nIMatrix entry has been deleted from the nIMatrix data tables.
(6)	NIMaxDesiredEntries	The desired maximum number of entries in the nIMatrix tables.
(7)	AlDroppedFrames	Number of received frames for this interface that the probe chose not to count for the associated aIMatrix entries.
(8)	AlInserts	Number of times an aIMatrix entry has been inserted into the aIMatrix data tables.
(9)	AlDeletes	Number of times an aIMatrix entry has been deleted from the aIMatrix data tables.
(10)	AlMaxDesiredEntries	Desired maximum number of entries in the aIMatrix tables.
(11)	Owner	Owner of this entry.
(12)	Status	Status of this entry.

## nIMatrixSDTable

nIMatrixSDTable stores statistics related to traffic from a source network-layer address to a number of destinations. The objects in this table are shown in Figure 141 on page 596 and listed in Table 480 on page 596.

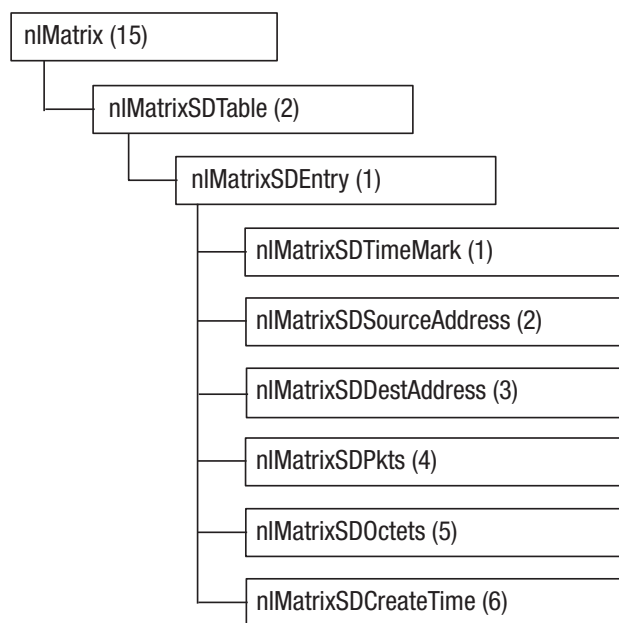


Figure 141. Structure of *nIMatrixSDTable*

Table 480. Descriptions of *nIMatrixSDTable* objects

Sub- OID	Object	Description
(1)	TimeMark	Time filter for this entry.
(2)	SourceAddress	Source network address for this entry.
(3)	DestAddress	Destination network address for this entry.
(4)	Pkts	Number of error-free packets transmitted from this source address to this destination address.
(5)	Octets	Number of octets, excluding packets with errors, transmitted from this source address to this destination address.
(6)	CreateTime	Value of sysUpTime when this control entry was activated.

## nIMatrixDSTable

nIMatrixDSTable contains the same information as nIMatrixSDTable but is indexed by destination address and then by source address. The objects in this table are shown in Figure 142 on page 597 and listed in Table 481 on page 597.

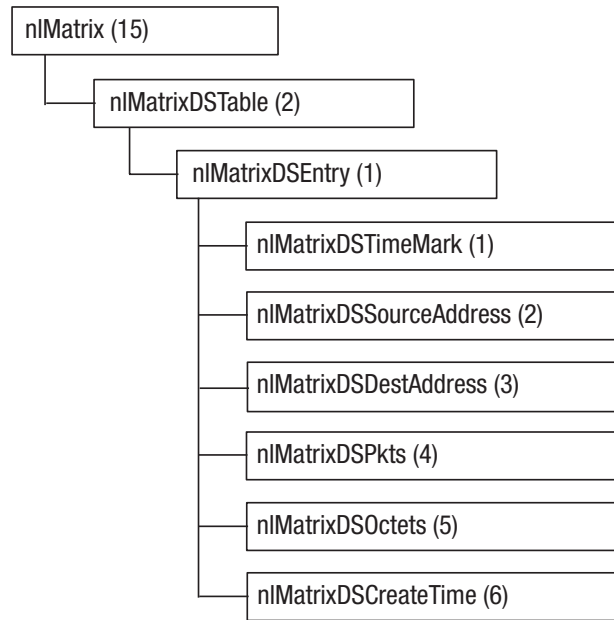


Figure 142. Structure of *nIMatrixDSTable*

Table 481. Descriptions of *nIMatrixDSTable* objects

Sub- OID	Object	Description
(1)	TimeMark	Time filter for this entry.
(2)	SourceAddress	Source network address for this entry.
(3)	DestAddress	Destination network address for this entry.
(4)	Pkts	Number of error-free packets transmitted from this destination address to this source address.
(5)	Octets	Number of octets, excluding packets with errors, transmitted from this destination address to this source address.
(6)	CreateTime	Value of sysUpTime when this control entry was activated.

## nIMatrixTopNControlTable

*nIMatrixTopNControlTable* specifies the set of parameters that control the creation of a report of the network-layer top-N matrix elements entries according to a selected metric. The objects in this table are shown in Figure 143 on page 598 and listed in Table 482 on page 598.

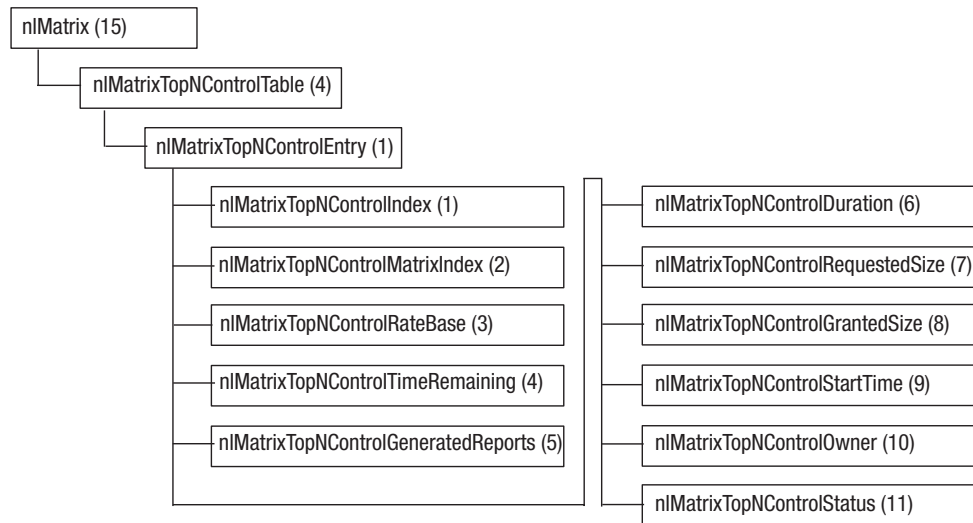


Figure 143. Structure of *nIMatrixTopNControlTable*

Table 482. Descriptions of *nIMatrixTopNControlTable* objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies a row in the <i>nIMatrixTopNControlTable</i> .
(2)	MatrixIndex	Specifies a particular sub network.
(3)	RateBase	Specifies the variable that is to be used to sort the table:  1 - <i>nIMatrixTopNPkts</i>  2 - <i>nIMatrixTopNOctets</i>
(4)	TimeRemaining	Number of seconds left in the sampling interval for the current report.
(5)	GeneratedReports	Number of reports that have been generated by this entry.
(6)	Duration	Sampling interval for this report in seconds.
(7)	RequestedSize	Maximum number of matrix entries requested for the top-N table.
(8)	GrantedSize	Maximum number of entries in the top-N table.
(9)	StartTime	Value of <i>sysUpTime</i> when this top-N report was last started.
(10)	Owner	Owner of this entry.
(11)	Status	Status of this entry.



## nlMatrixTopNTable

nlMatrixTopNTable contains a set of statistics for those network-layer matrix entries that have counted the highest number of octets or packets. The objects in this table are shown in Figure 144 and listed in Table 483.

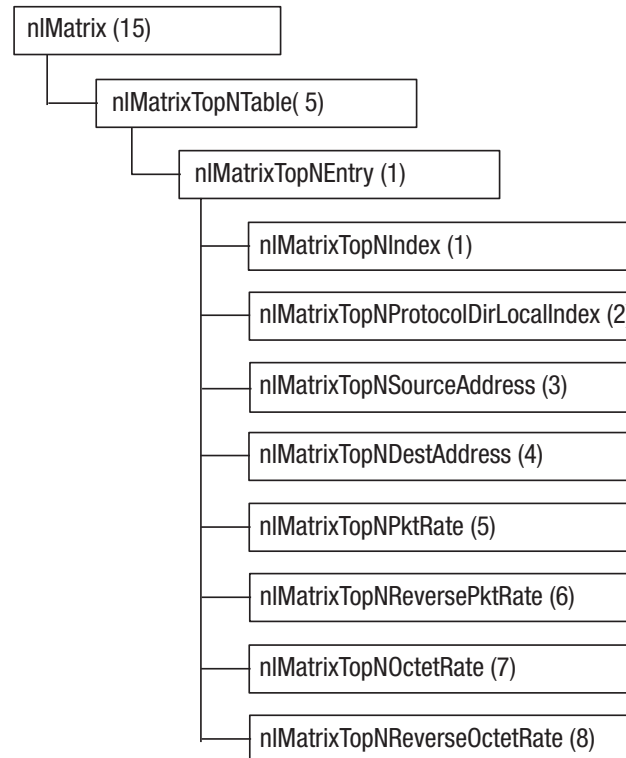


Figure 144. Structure of nlMatrixTopNTable

Table 483. Descriptions of nlMatrixTopNTable Objects

Sub- OID	Object	Description
(1)	Index	Index that uniquely identifies one row among all data rows associated with a particular report; each row represents a unique source-destination pair.
(2)	ProtocolDirLocalIndex	protocolDirLocalIndex value that uniquely identifies a network-layer protocol for this entry.
(3)	SourceAddress	Network address of the source host in this conversation.
(4)	DestAddress	Network address of the destination host in this conversation.
(5)	PktRate	Number of packets seen from the source host to the destination host during this sampling interval.
(6)	ReversePktRate	Number of packets seen from the destination host to the source host during this interval.
(7)	OctetRate	Number of octets seen from the source host to the destination host during this interval.
(8)	ReverseOctetRate	Number of octets seen from the destination host to the source host during this interval.

---

## alHost group

The alHost group holds a collection of statistics for a protocol from a particular network address that has been discovered on an interface of this device.

The probe creates entries for each protocol in the protocol directory table whose value of protocolDirHostConfig is equal to supportedOn(3). The objects in this group are shown in Figure 145 and listed in Table 484.

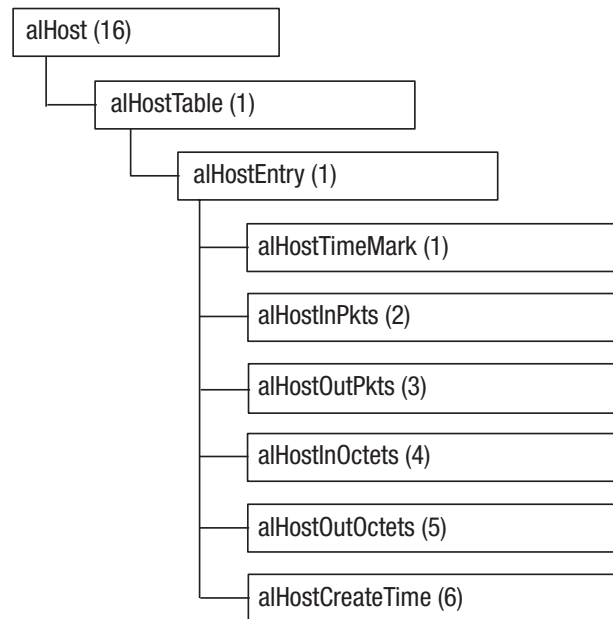


Figure 145. Structure of alHostTable

Table 484. Descriptions of alHostTable objects

Sub-OID	Object	Description
(1)	TimeMark	Time filter for this entry.
(2)	InPkts	Number of error-free packets of this protocol type transmitted to this address since it was added to the table.
(3)	OutPkts	Number of error-free packets of this protocol type transmitted by this address since it was added to the table.
(4)	InOctets	Number of octets of this protocol type transmitted to this address since it was added to the table, excluding packets with errors.
(5)	OutOctets	Number of octets of this protocol type transmitted by this address since it was added to the table, excluding packets with errors.
(6)	CreateTime	Value of sysUpTime when this entry was activated.

## alMatrix group

The alMatrix group deals with the collection of statistics on pairs of hosts based on the application-layer protocol.

The statistics relate to traffic between pairs of hosts for each protocol, for all conversations between pairs of hosts. The data tables in this group collect statistics similar to those collected by the RMON1 matrix and hostTopN groups.

### alMatrixSDTable

alMatrixSDTable stores statistics on traffic from a particular source to a number of destinations. It creates entries for all application-layer protocols in the protocol directory table whose value of protocolDirAlMatrixConfig is equal to supportedOn(3). The objects in this table are shown in Figure 146 and listed in Table 485.

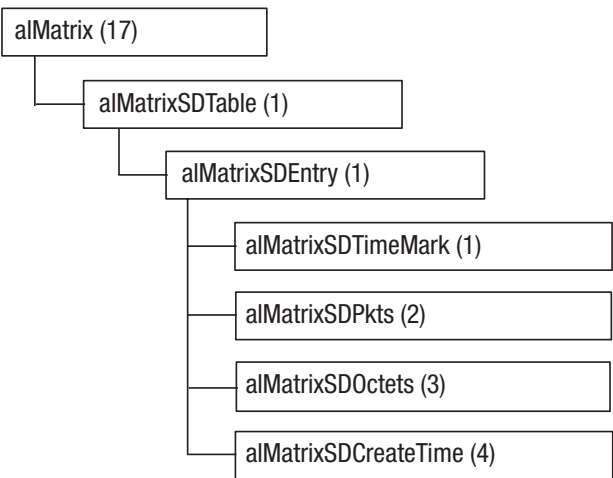


Figure 146. Structure of alMatrixSDTable

Table 485. Descriptions of alMatrixSDTable objects

Sub-OID	Object	Description
(1)	TimeMark	Time filter for this entry.
(2)	Pkts	Number of error-free packets transmitted from this source address to this destination address.
(3)	Octets	Number of octets, excluding packets with errors, transmitted from this source address to this destination address.
(4)	CreateTime	Value of sysUpTime when this control entry was activated.

## alMatrixDSTable

alMatrixDSTable contains the same information as alMatrixSDTable but is indexed by destination address and then by source address. The objects in this table are shown in Figure 147 and listed in Table 486.

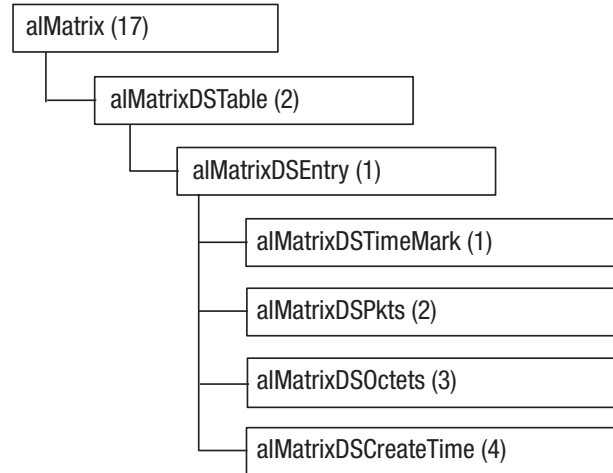


Figure 147. Structure of alMatrixDSTable

Table 486. Descriptions of alMatrixDSTable objects

Sub-OID	Object	Description
(1)	TimeMark	Time filter for this entry.
(2)	Pkts	Number of error-free packets transmitted from this destination address to this source address.
(3)	Octets	Number of octets, excluding packets with errors, transmitted from this destination address to this source address.
(4)	CreateTime	Value of sysUpTime when this control entry was activated.

## alMatrixTopNControlTable

alMatrixTopNControlTable specifies the set of parameters that control the creation of a report of the application-layer top-N matrix elements entries according to a selected metric. The objects in this table are shown in Figure 148 on page 603 and listed in Table 487 on page 603.

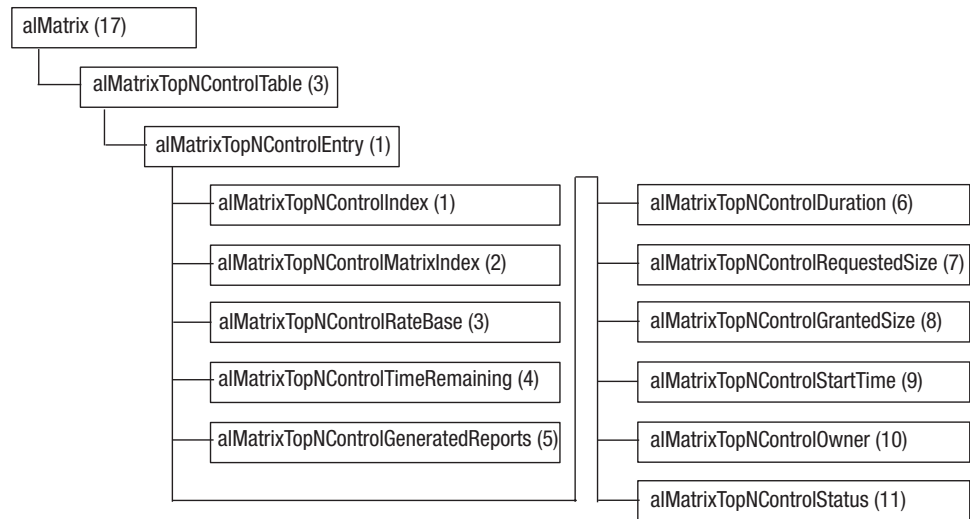


Figure 148. Structure of *alMatrixTopNControlTable*

Table 487. Descriptions of *nlMatrixTopNControlTable* objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies a row in the <i>alMatrixTopNControlTable</i> .
(2)	MatrixIndex	Specifies a particular sub network.
(3)	RateBase	Specifies one of two objects in <i>alMatrixTopNTable</i> that is to be used to sort the table, as well as the selector of the view of the matrix table that will be used; it is of the following type:  <i>alMatrixTopNTerminalsPkts</i> (1) <i>alMatrixTopNTerminalsOctets</i> (2) <i>alMatrixTopNAllPkts</i> (2) <i>alMatrixTopNAllOctets</i> (4)  <b>Note:</b> The values that specify “terminal” protocols will collect data only from protocols that have no child protocols.
(4)	TimeRemaining	Number of seconds left in the sampling interval for the report currently being collected.
(5)	GeneratedReports	Number of reports that have been generated by this entry.
(6)	Duration	Sampling interval, in seconds, for this report.
(7)	RequestedSize	Maximum number of matrix entries requested for the TopN table.
(8)	GrantedSize	Maximum number of entries in the TopN table.
(9)	StartTime	Value of <i>sysUpTime</i> when this TopN report was last started.
(10)	Owner	Owner of this entry.
(11)	Status	Status of this entry.

## alMatrixTopNTable

alMatrixTopNTable contains a set of statistics for those application-layer matrix entries that have counted the highest number of octets or packets. The objects in this table are shown in Figure 149 and listed in Table 488.

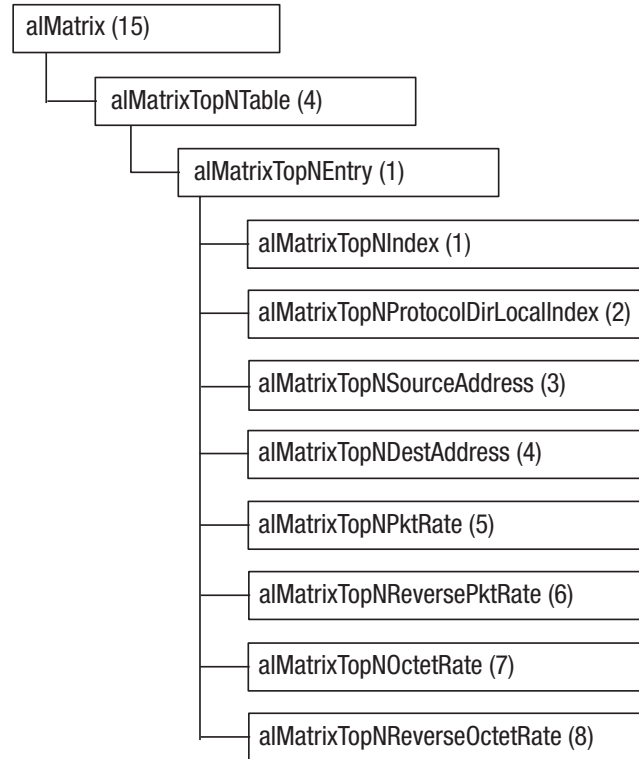


Figure 149. Structure of alMatrixTopNTable

Table 488. Descriptions of alMatrixTopNTable objects

Sub- OID	Object	Description
(1)	Index	Index that uniquely identifies one row, among all data rows, associated with a particular report; each row represents a unique source-destination pair of hosts.
(2)	ProtocolDirLocalIndex	protocolDirLocalIndex value that uniquely identifies a network-layer protocol for this entry.
(3)	SourceAddress	Network-layer address of the source host in this pairing.
(4)	DestAddress	Network-layer address of the destination host in this pairing.
(5)	AppProtocolDirLocalIndex	Application-level protocol being counted.
(6)	PktRate	Number of packets seen from the source host to the destination host during this sampling interval.
(7)	ReversePktRate	Number of packets seen from the destination host to the source host during this interval.
(8)	OctetRate	Number of octets seen from the source host to the destination host during this interval.

Table 488. Descriptions of *alMatrixTopNTable* objects (continued)

Sub- OID	Object	Description
(9)	ReverseOctetRate	Number of octets seen from the destination host to the source host during this sampling interval.

## usrHistory group

The *usrHistory* group combines mechanisms seen in the alarm and history groups. It periodically samples user-specified variables and logs that data, based on user-defined parameters.

This group consists of a three-level hierarchy of tables. At the top level is *usrHistoryControlTable* that specifies the details of the sampling function. Subordinate to this are one or more instances of *usrHistoryObjectTable* that specifies the variables to be sampled. Subordinate to each instance in the *usrHistoryObjectTable* are one or more instances of *usrHistoryTable* that record the specified data. Each row in the *usrHistoryTable* represents the value of a single MIB object instance during a specific sampling interval.

### usrHistoryControlTable

*usrHistoryControlTable* is a one-dimensional “read-create” table in which each row configures a collection of user history buckets. It creates a list of parameters that set up a group of user defined MIB objects to be sampled periodically. The objects in this table are shown in Figure 150 and listed in Table 489 on page 606.

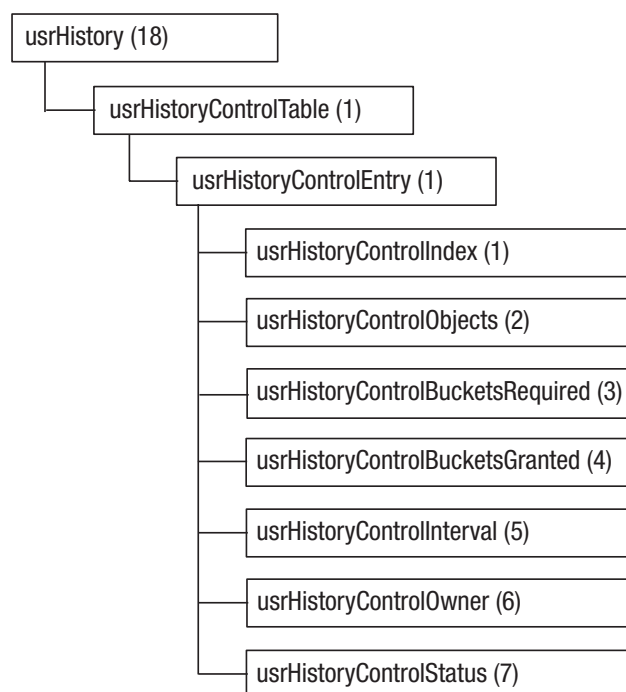


Figure 150. Structure of *usrHistoryControlTable*

Table 489. Descriptions of *usrHistoryControlTable* objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies a row in the <i>usrHistoryControlTable</i> ; each entry configures a set of samples at a particular interval for a specified set of MIB object instances.
(2)	Objects	Number of MIB objects for which data is to be collected in the portion of the <i>usrHistoryTable</i> associated with this entry.
(3)	BucketsRequested	Requested number of discrete sampling intervals over which data is to be saved in the part of the <i>usrHistoryTable</i> associated with this entry.
(4)	BucketsGranted	Actual number of discrete sampling intervals granted.
(5)	Interval	Interval in seconds over which data is sampled for each bucket in the part of the <i>usrHistoryTable</i> associated with this entry.
(6)	Owner	Owner of this entry.
(7)	Status	Status of this entry.

## usrHistoryObjectTable

*usrHistoryObjectTable* provides a list of MIB instances to be sampled periodically. The objects in this table are shown in Figure 151 and listed in Table 490.

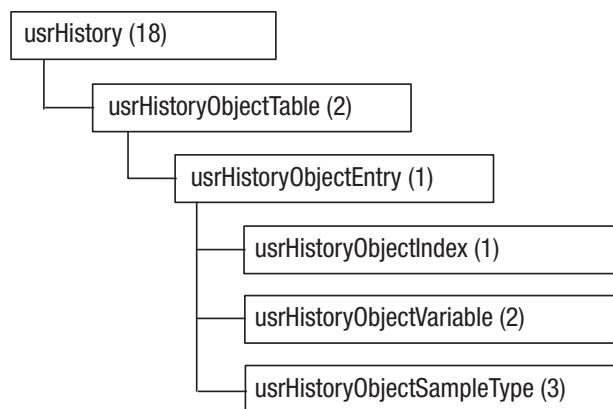


Figure 151. Structure of *usrHistoryObjectTable*

Table 490. Descriptions of *usrHistoryObjectTable* objects

Sub- OID	Object	Description
(1)	Index	Integer that uniquely identifies a row in the <i>usrHistoryObjectTable</i> ; each entry defines a MIB object instance.
(2)	Variable	Object identifier that specifies the particular MIB object instance to be sampled.



Table 490. Descriptions of *usrHistoryObjectTable* objects (continued)

Sub- OID	Object	Description
(3)	SampleType	Method of sampling the selected variable; can be:  absoluteValue(1)  deltaValue(2)

## usrHistoryTable

*usrHistoryTable* provides a list of user defined history entries. The objects in this table are shown in Figure 152 and listed in Table 491.

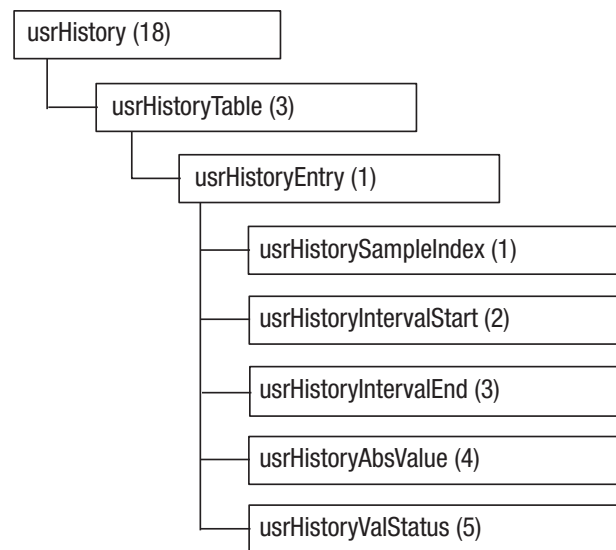


Figure 152. Structure of *usrHistoryTable*

Table 491. Descriptions of *usrHistoryTable* objects

Sub- OID	Object	Description
(1)	SampleIndex	Index that uniquely identifies the particular sample this entry represents among all samples associated with the same row of the <i>usrHistoryControlTable</i> .
(2)	IntervalStart	Value of <i>sysUpTime</i> at the start of the interval over which this sample was measured.
(3)	IntervalEnd	Value of <i>sysUpTime</i> at the end of the interval over which this sample was measured.
(4)	AbsValue	Absolute value of the user specified statistic during the last sampling interval.
(5)	ValStatus	Takes on the values:  valueNotAvailable(1)  valuePositive(2)  valueNegative(3)

---

## probeConfig group

The probeConfig group defines standard configuration parameters for the agent's capability (supported groups), software revision, reset control (warm/cold boot) as well as the trap destination table (a list of trap recipient IP hosts).

The objects in this group are shown in Figure 153 and listed in Table 492.

**Note:** The objects, serialConfigTable, netConfigTable, netDefaultGateway, and serialConnectionTable are not supported.

Combined the following download objects are supported: probeDownloadFile; probeDownloadTFTPServer; probeDownloadAction; and probeDownloadStatus. The agent connects to the TFTP server specified in probeDownloadTFTPServer, and downloads the file specified in probeDownloadFile into the agent root directory.

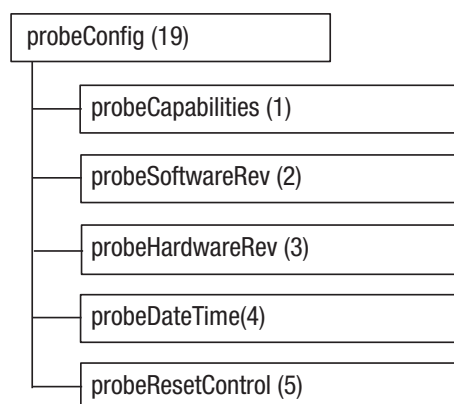


Figure 153. Structure of probeConfig group

Table 492. Descriptions of probeConfig objects

Sub-OID	Object	Description
(1)	probeCapabilities	Indicates what rmon groups are supported.
(2)	probeSoftwareRev	Software revision of this device: this string will have zero length if the revision is unknown.
(3)	probeHardwareRev	Hardware revision of this device.
(4)	probeDateTime	Probe's current date and time.
(5)	probeResetControl	Takes on the values:  running(1)  warmBoot(2)  coldBoot(3)

## trapDestTable

trapDestTable defines the destination addresses for traps generated from this device. The objects in this table are shown in Figure 154 and listed in Table 493.

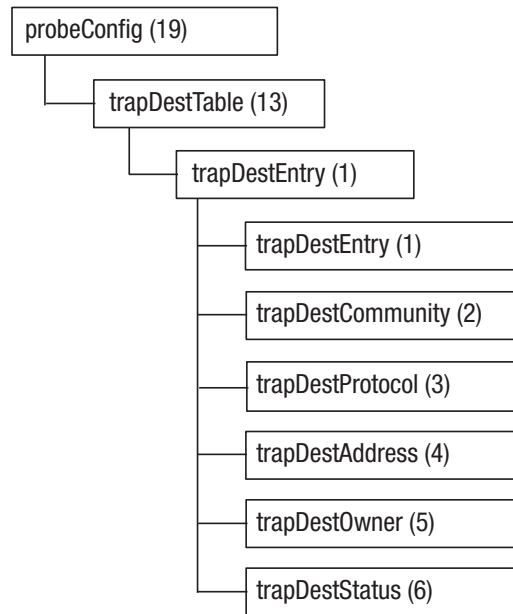


Figure 154. Structure of trapDestTable

Table 493. Descriptions of trapDestTable objects

Sub- OID	Object	Description
(1)	Index	Unique index for this row.
(2)	Community	Community to which these destination IP addresses belong; each time an associated event entry sends a trap due to an event, that trap will be sent to each address in the trapDestTable with a trapDestCommunity value equal to eventCommunity.
(3)	Protocol	Protocol with which to send this trap (IP).
(4)	Address	IP address to which to send traps on behalf of this entry.
(5)	Owner	Owner of this entry.
(6)	Status	Status of this entry.

---

## Configuration commands

The subagent provides a set of configuration commands for controlling its operation.

You can use these commands from the command console or in configuration files. For general instructions about how to use configuration commands, see the *Netcool/SSM Administration Guide*.

**Note:** Configuration commands are case-sensitive.

## Address map control table

The addressmap commands create rows in the address map control table (addressMapControlTable).

The general syntax of these commands is:

```
addressmap property=value
addressmap create [property=value ...]
addressmap reset
```

Table 494 lists the properties supported in these commands.

Table 494. Configuration command parameters - addressmap

Property	Type	Description	Sets MIB object
datasource	OID	The interface monitored by the control row.	DataSource

## Application-layer topN control table

The almatrixtopn commands create rows in the application-layer top-N control table (alMatrixTopNControlTable).

The general syntax of these commands is:

```
almatrixtopn property=value
almatrixtopn create [property=value ...]
almatrixtopn reset
```

Table 495 lists the properties supported in these commands.

Table 495. Configuration command parameters - almatrixtopn

Property	Type	Description	Sets MIB object
matrixindex	int	The index of the sub-network.	MatrixIndex
ratebase	int	Selects the variable used to sort the table:  1 - nIMatrixTopNPkts  2 - nIMatrixTopNOctets	RateBase
time	int	The time left (in seconds) in the current sampling interval.	TimeRemaining
size	int	The number of matrix entries requested for the table.	RequestedSize

## Higher-layer host control table

The hlhost commands create rows in the higher-layer host control table (hlHostControlTable).

The general syntax of these commands is:

```
hlhost property=value
hlhost create [property=value ...]
hlhost reset
```

Table 496 on page 611 lists the properties supported in these commands.

Table 496. Configuration command parameters - hlhost

Property	Type	Description	Sets MIB object
datasource	OID	The interface monitored by the control row.	DataSource
nlmax	enum	The number of entries requested in the NL table.	NlMaxDesiredEntries
almax	enum	The number of entries requested in the AL table.	AlMaxDesiredEntries

## Higher-layer matrix control table

The hlmatrix commands create rows in the higher-layer matrix control table (hlMatrixControlTable).

The general syntax of these commands is:

```
hlmatrix property=value
hlmatrix create [property=value ...]
hlmatrix reset
```

Table 497 lists the properties supported in these commands.

Table 497. Configuration command parameters - hlmatrix

Property	Type	Description	Sets MIB object
datasource	OID	The interface monitored by the control row.	DataSource
nlmax	enum	The number of entries requested in the NL table.	NlMaxDesiredEntries
almax	enum	The number of entries requested in the AL table.	AlMaxDesiredEntries

## Network-layer Top-N matrix control table

The nlmatrixtopn commands create rows in the network-layer top-N matrix control table (nlMatrixTopNControlTable).

The general syntax of these commands is:

```
nlmatrixtopn property=value
nlmatrixtopn create [property=value ...]
nlmatrixtopn reset
```

Table 498 lists the properties supported in these commands.

Table 498. Configuration command parameters - nlmatrixtopn

Property	Type	Description	Sets MIB object
matrixindex	int	The index of the sub-network.	MatrixIndex
ratebase	int	Selects the variable used to sort the table:  1 - nlMatrixTopNPkts  2 - nlMatrixTopNOctets	RateBase
time	int	The time left (in seconds) in the current sampling interval.	TimeRemaining

Table 498. Configuration command parameters - *nlmatrixtopn* (continued)

Property	Type	Description	Sets MIB object
size	int	The number of matrix entries requested for the table.	RequestedSize

## Protocol distribution control table

The `protocoldist` commands create rows in the protocol distribution control table (`protocolDistControlTable`).

The general syntax of these commands is:

```
protocoldist property=value
protocoldist create [property=value ...]
protocoldist reset
```

Table 499 lists the properties supported in these commands.

Table 499. Configuration command parameters - *protocoldist*

Property	Type	Description	Sets MIB object
datasource	OID	The interface monitored by the control row.	DataSource

## Address map scalars

The `addressmapglobal` command sets the value of scalar objects in the `addressMap` group.

The general syntax of this command is:

```
addressmapglobal property=value
```

Table 500 lists the properties supported in this command.

Table 500. Configuration command parameters - *addressmapglobal*

Property	Type	Description	Sets MIB object
maxdesiredentries	int	Desired maximum number of entries in the address map table. <b>Note:</b> An entry of -1 denotes any number of entries.	MaxDesiredEntries

## User history control table

The `usrhistory` commands create rows in the user history control table (`usrHistoryControlTable`).

The general syntax of these commands is:

```
usrhistory property=value
usrhistory create [property=value ...]
usrhistory reset
```

Table 501 lists the properties supported in these commands.

Table 501. Configuration command parameters - *usrhistory*

Property	Type	Description	Sets MIB object
buckets	string	The number of rows requested in the user history table.	BucketsRequested

Table 501. Configuration command parameters - *usrhistory* (continued)

Property	Type	Description	Sets MIB object
history	enum	Sets the sample interval (in seconds).	Interval

## User history object table

The *usrhistoryvar* commands create rows in the user history object table (*usrHistoryObjectTable*) and associates them with the next *usrHistoryControlTable* row created using the *usrhistory create* command.

The general syntax of these commands is:

```
usrhistoryvar property=value
usrhistoryvar store [property=value ...]
usrhistoryvar reset
```

Table 502 lists the properties supported in these commands.

Table 502. Configuration command parameters - *usrhistoryvar*

Property	Type	Description	Sets MIB object
var	OID	The variable to be sampled.	Variable
type	enum	The sampling method:  absolute  delta	SampleType





---

## Appendix D. Regular expressions

Netcool/SSM enables you to use regular expressions in many subagent configuration commands.

---

### Regular expression syntax

Table 503 describes the syntax of the regular expression tokens supported by Netcool/SSM.

*Table 503. Regular expression syntax*

Token	Matches
.	Any character.
^	The start of a line (a zero-length string).
\$	The end of a line; a new line or the end of the search buffer.
\<	The start of a word (where a word is a string of alphanumeric characters).
\>	The end of a word (the zero length string between an alphanumeric character and a non-alphanumeric character).
\b	Any word boundary (this is equivalent to (\< \>) ).
\d	A digit character.
\D	Any non-digit character.
\w	A word character (alphanumeric or underscore).
\W	Any character that is not a word character (alphanumeric or underscore).
\s	A whitespace character.
\S	Any non-whitespace character.
\c	Special characters and escaping. The following characters are interpreted according to the C language conventions: \0, \a, \f, \n, \r, \t, \v. To specify a character in hexadecimal, use the \xNN syntax. For example, \x41 is the ASCII character A.
\	All characters apart from those described above may be escaped using the backslash prefix. For example, to specify a plain left-bracket use \[.

Table 503. Regular expression syntax (continued)

Token	Matches
[ ]	<p>Any one of the specified characters in a set. An explicit set of characters may be specified as in [aeiou] as well as character ranges, such as [0-9A-Fa-f], which match any hexadecimal digit. The dash (-) loses its special meaning when escaped, such as in [A\ -Z] or when it is the first or last character in a set, such as in [-xyz0-9].</p> <p>All of the above backslash-escaping rules may be used within [ ]. For example, the expression [\x41-\x45] is equivalent to [A-D] in ASCII. To use a closing bracket in a set, either escape it using [\]] or use it as the first character in the set, such as []xyz].</p> <p>POSIX-style character classes are also allowed inside a character set. The syntax for character classes is [:class:]. The supported character classes are:</p> <ul style="list-style-type: none"> <li>[:alnum:] - alphanumeric characters.</li> <li>[:alpha:] - alphabetic characters.</li> <li>[:blank:] - space and TAB characters.</li> <li>[:cntrl:] - control characters.</li> <li>[:digit:] - numeric characters.</li> <li>[:graph:] - characters that are both printable and visible.</li> <li>[:lower:] - lowercase alphabetic characters.</li> <li>[:print:] - printable characters (characters that are not control characters).</li> <li>[:punct:] - punctuation characters (characters that are not letters, digits, control characters, or spaces).</li> <li>[:space:] - space characters (such as space, TAB and form feed).</li> <li>[:upper:] - uppercase alphabetic characters.</li> <li>[:xdigit:] - characters that are hexadecimal digits.</li> </ul> <p>Brackets are permitted within the set's brackets. For example, [a-z0-9!] is equivalent to [[:lower:][:digit:]!] in the C locale.</p>
[^]	Inverts the behavior of a character set [ ] as described above. For example, [^[:alpha:]] matches any character that is not alphabetical. The ^ caret symbol only has this special meaning when it is the first character in a bracket set.
{n}	Exactly n occurrences of the previous expression, where 0 <= n <= 255. For example, a{3} matches aaa.
{n,m}	Between n and m occurrences of the previous expression, where 0 <= n <= m <= 255. For example, a 32-bit hexadecimal number can be described as 0x[[:xdigit:]]{1,8}.
{n,}	At least n or more (up to infinity) occurrences of the previous expression.
*	Zero or more of the previous expression.
+	One or more of the previous expression.
?	Zero or one of the previous expression.
(exp)	<p>Grouping; any series of expressions may be grouped in parentheses so as to apply a postfix or bar ( ) operator to a group of successive expressions. For example:</p> <ul style="list-style-type: none"> <li>ab+ matches all of abbb</li> <li>(ab)+ matches all of ababab</li> </ul>
	<p>Alternate expressions (logical OR). The vertical bar ( ) has the lowest precedence of all tokens in the regular expression language. This means that ab cd matches all of cd but does not match abd (in this case use a(b c)d ).</p>

**Tip:** When defining regular expressions to match multi-byte characters, enclose each multi-byte character in parentheses ().

## Regular expression examples

These examples demonstrate how to use regular expressions to perform pattern matching.

Table 504 provides a set of regular expression examples, together with sample strings as well as the results of applying the regular expression to those strings.

There are two important cases in matching regular expressions with strings. A regular expression may match an entire string (a case known as a *string match*) or only a part of that string (a case known as a *sub-string match*). For example, the regular expression `\<int\>` will generate a *sub-string match* for the string `int x` but will not generate a *string match*. This distinction is important because some subagents do not support sub-string matching. Where applicable, the results listed in the examples differentiate between string and sub-string matches.

Table 504. Regular expression examples

This expression...	Applied to this string...	Results in...
.	a	String match
	!	String match
	abcdef	Sub-string match on a
	empty string	No match
M..COUNT	MINCOUNT	String match
	MXXCOUNTY	Sub-string match on MXXCOUNT
	NONCOUNT	No match
.*	empty string	String match
	Animal	String match
.+	Any non-empty string	String match
	empty string	No match
^	empty string	String match
	hello	Sub-string match of length 0 at position 0 (position 0 = first character in string)
\$	empty string	String match
	hello	Sub-string match of length 0 at position 5 (position 0 = first character in string)
^\$	empty string	String match
	hello	No match
\bee	tee	No match
	Paid fee	No match
	feel	No match
	eel	Sub-string match on ee

Table 504. Regular expression examples (continued)

This expression...	Applied to this string...	Results in...
.*thing.*	The thing is in here	String match
	there is a thing	String match
	it isn't here	No match
	thinxxx	No match
a*	empty string	String match
	aaaaaaaaa	String match
	a	String match
	aardvark	Sub-string match on aa
	this string	Sub-string match
((ab)*c)*	empty string	String match
	cccccccc	String match
	ccccabcccabc	String match
a+	empty string	No match
	aaaaaaaaa	String match
	a	String match
	aardvark	Sub-string match on aa
	this string	No match
((ab)+c)*	empty string	String match
	ababababcabc	String match
(ab){2}	abab	String match
	cdabababab	Sub-string match on abab
	ab	No match
[0-9]{4,}	123	No match
	a1234	Sub-string match on 1234
a{0}	empty string	String match
	a	No match
	hello	Sub-string match of length 0 at position 0 (position 0 = first character in string)
[0-9]{1,8}	this is not a number	No match
	a=4238, b=4392876	Sub-string match on 4238
([aeiou][^aeiou])+	Hello	Sub-string match on el
	!!! Supacalafraglistic	Sub-string match on upacalaf
[+-]?1	1	String match
	+1	String match
	-1	String match
	.1	Sub-string match on 1
	value+1	Sub-string match on +1

Table 504. Regular expression examples (continued)

This expression...	Applied to this string...	Results in...
a b	a	String match
	b	String match
	c	No match
	Daniel	Sub-string match on a
abcd efgh	abcd	String match
	efgh	String match
	abcdfgh	Sub-string match on abcd
[0-9A-F]+	BAADF00D	String match
	C	String match
	baadF00D	Sub-string match on F00D
	c	No match
	G	No match
	g	No match
x = \d+	x = 1234	String match
	x = 0	String match
	x = 1234a	Sub-string match on x = 1234
	x = y	No match
	x^=^ where ^ represents a space character	No match
\D\d	a1	String match
	a11	Sub-string match on a1
	-9	String match
	a	No match
	8	No match
	aa	No match
	4t	No match
\s+	Hello_w0rld	No match
	Hello^^^world where ^ represents a space character	Sub-string match on ^^^ where ^ represents a space character
	Widget^ where ^ represents a space character	Sub-string match on ^ where ^ represents a space character
	^^^ where ^ represents a space character	String match

Table 504. Regular expression examples (continued)

This expression...	Applied to this string...	Results in...
\S+	Hello_w0rld	Sub-string match of length 11 on Hello_w0rld
	Hello^^world where ^ represents a space character	Sub-string match on Hello
	Widget^ where ^ represents a space character	Sub-string match on Widget
	^^^ where ^ represents a space character	No match
\w+	D4n_v4n Vugt	Sub-string match on D4n_v4n
	^^hello where ^ represents a space character	Sub-string match on hello
	blah	String match
	x#1	No match
	foo bar	No match
\W	Hello there	Sub-string match of length 1 on separating space character
	~	String match
	aa	No match
	a	No match
	_	No match
	^^^444 == 5 where ^ represents a space character	Sub-string match of length 1 on first ^ where ^ represents a space character
\w+\s*=\s*\d+	x = 123	String match
	count0=555	String match
	my_var=66	String match
	0101010=0	String match
	xyz = e	No match
	delta=	No match
	==8	No match
[[:alnum:]]+	1234	String match
	...D4N13L	Sub-string match on D4N13L
[[:alpha:]]+	Bubble	String match
	...DANI3L	Sub-string match on DANI
	69	No match

Table 504. Regular expression examples (continued)

This expression...	Applied to this string...	Results in...
[[[:blank:]]+]	alpha^^^and beta where ^ represents a space character	Sub-string match on ^^^ where ^ represents a space character
	Animal	No match
	empty string	No match
[[[:space:]]+]	alpha^^^and beta where ^ represents a space character	Sub-string match on ^^^ where ^ represents a space character
	Animal	No match
	empty string	No match
[[[:cntrl:]]+]	...Hello W0rld!	No match
	empty string	No match
[[[:graph:]]+]	hello world	Sub-string match on hello
	^^^ where ^ represents a space character	No match
	^^! ? where ^ represents a space character	Sub-string match on ! ?
[[[:lower:]]+]	Animal	Sub-string match on nimal
	ABC	No match
	0123	No match
	foobar	String match
	^^0blaH! where ^ represents a space character	Sub-string match on bla
[_[:lower:]]+	foo_bar	String match
	this_thinG!!!	Sub-string match on _thin
[[[:upper:]]+]	YES	String match
	#define MAX 100	Sub-string match on MAX
	f00 b4r	No match
[[[:print:]]+]	hello world	String match
	^^^ where ^ represents a space character	String match
[[[:punct:]]+]	didn't	Sub-string match on '
	Animal	No match
[[[:xdigit:]]+]	43298742432392187ffe	String match
	x = bAAdF00d	Sub-string match on bAAdF00d
	4327afeffegokpoj	Sub-string match on 4327afeffe
c:\\temp	c:\\temp	String match





---

## Notices and trademarks

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan, Ltd.  
19-21, Nihonbashi-Hakozakicho, Chuo-ku  
Tokyo 103-8510, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:**

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation  
958/NH04  
IBM Centre, St Leonards  
601 Pacific Hwy  
St Leonards, NSW, 2069  
Australia

IBM Corporation  
896471/H128B  
76 Upper Ground  
London SE1 9PZ  
United Kingdom

IBM Corporation  
2Z4A/101  
11400 Burnet Road  
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the

names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

---

## Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.



Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.







Printed in USA

SC23-8763-02

