

CUDB LDAP Data Access

FACILITY DESCRIPTION

Copyright

© Ericsson AB 2016, 2017. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

Trademark List

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



Contents

1	Introduction	1
1.1	Scope	1
1.2	Revision Information	2
1.3	Target Groups	2
1.4	Prerequisites	2
1.5	Typographic Conventions	3
2	Overview	5
2.1	Prerequisites	6
2.2	Architecture	6
2.3	Description	9
2.4	Dependencies and Interactions	38
3	Operation and Maintenance	41
3.1	Configuration	41
3.2	Provisioning	43
3.3	Fault Management	44
3.4	Performance Management	44
3.5	Security	44
3.6	Logging	45
4	Appendix A: Traffic Prioritization Configuration Guidelines	47
4.1	Traffic Prioritization Configuration	47
4.2	Traffic Prioritization Examples	47
4.3	Performance Management	63
	Glossary	65
	Reference List	67





1 Introduction

This document describes how the Ericsson Centralized User Database (CUDB) provides access over the Lightweight Directory Access Protocol (LDAP) to the data stored in the CUDB system.

1.1 Scope

This document provides information on the CUDB LDAP data access feature, its management, and its operation by covering the following topics:

- Architecture.
- Service, information, and security models.
- Operation and maintenance.

For more information, refer to the following documents:

- [RFC 4510 LDAP: Technical Specification Road Map](#), Reference [22].
- [RFC 4511 LDAP: The Protocol](#), Reference [23].
- [RFC 4512 LDAP: Directory Information Models](#), Reference [24].
- [RFC 4513 LDAP: Authentication Methods and Security Mechanisms](#), Reference [25].
- [RFC 4514 LDAP: String Representation of Distinguished Names](#), Reference [26].
- [RFC 4515 LDAP: String Representation of Search Filters](#), Reference [27].



1.2 Revision Information

- Rev. A** This document is based on 5/155 34-HDA 104 03/9 with the following changes:
- Terminology updates throughout the document because of virtualized deployment support, including figure updates.
 - Updates throughout the document on configurations containing nodes without local Processing Layer Database.
 - Section 2 on page 5, Section 2.4 on page 38, and Section 3.1.3 on page 41: Updated information on function availability.
 - Section 2.3.2.5 on page 15: Updated with information on locking and deleting parent entries.
- Rev. B** Other than editorial changes, this document has been revised as follows:
- Section 2.3.6.1 on page 25: Updated Figure 16.
 - Section 3.1.3 on page 41: Added a value combination for the `readModeInPL` and `readModeInDS` LDAP user attributes and a note about the configuration of `readModeInDS=LP`.
- Rev. C** Other than editorial changes, this document has been revised as follows:
- Section 2.3.6 on page 24: Added a reference to *Section 6.4.1 ReadMode Control, CUDB LDAP Interwork Description*, Reference [6]
 - Section 3.1.3 on page 41: Added a reference to *Section 6.4.1 ReadMode Control, CUDB LDAP Interwork Description*, Reference [6].

1.3 Target Groups

This document is intended for system administrators and users working with the CUDB LDAP Data Access feature.

1.4 Prerequisites

This document assumes the knowledge of the general LDAP standard, and the related LDAP protocol, service and information models.



1.5 Typographic Conventions

Typographic conventions can be found in the following document:

- *Typographic Conventions*





2 Overview

CUDB is a highly available, geographically redundant, real-time and distributed database exposed as a LDAP directory to the network application. As such, CUDB provides an LDAP data access service.

Each CUDB node provides an LDAP data access point, giving access to the whole database, regardless of the data distribution across the interconnected CUDB nodes.

The CUDB data access feature is offered through LDAP. LDAP data access service is specified by the set of Internet Engineering Task Force (IETF) RFC standards listed in [RFC 4513 LDAP: Authentication Methods and Security Mechanisms](#), Reference [25]. In practice, it is defined in terms of the LDAP service and the architecture of the underlying system which provides it. Particularly, architecture is that of a partitioned, replicated and distributed directory, as described in Section 2.2 on page 6.

The LDAP data access feature is defined in terms of the following supported information, service and security models:

Information Model

The information model encloses both the LDAP schema and the naming model. The LDAP schema lists the set of attributes and object classes that are allowed in CUDB entries. The naming model describes how data instances are accessed in the Directory Information Tree (DIT). Both areas are extended by client applications by adding their own definitions to the CUDB system.

For more information, refer to the following documents:

- *CUDB LDAP Schema Management*, Reference [2] regarding the client extensions and mechanisms.
- *CUDB LDAP Data Views*, Reference [3] regarding the LDAP Data Views function and custom data views.

Note: The LDAP Data Views function can only be used if the Application Facilitator Value Package is available.

- *CUDB Application Integration Guide*, Reference [4] and *CUDB Node Schema Update*, Reference [5] regarding the procedures to follow.

Service Model

The CUDB service model is described by the LDAP standard, as specified in [RFC 4511 LDAP: The Protocol](#), Reference [23], along with specifics provided in this document and in *CUDB LDAP Interwork Description*, Reference [6].



The CUDB data access service is optimized for real time data access, both reading and updating.

Data access is provided through standard LDAP operations (Bind, Add, Delete, Modify, Abandon, Search and Unbind).

The LDAP service follows a client-server approach in which CUDB is the LDAP server and the network application is the LDAP client. The LDAP protocol is also used internally among CUDB nodes (internal proxy function), where CUDB behaves as an LDAP client as well.

Security Model

The CUDB security model is implemented at transport, authentication and authorization levels.

At transport level, CUDB supports the secure LDAP over SSL (LDAPS) suite as defined in [RFC 4513 LDAP: Authentication Methods and Security Mechanisms](#), Reference [25].

At authentication level, both Simple Authentication and Security Layer (SASL) and simple authentication following the LDAP RFC standard are supported.

Finally, at authorization level, CUDB supports Access Control Lists (ACLs) on per user or per user group basis. The data in the CUDB LDAP directory that may be accessed by each client can be specified and restricted using those ACLs.

For more details about the security model, see Section 3.5 on page 44.

2.1 Prerequisites

This section is not applicable to this feature.

2.2 Architecture

This section describes the architecture of LDAP data access.

2.2.1 Single Point of Access

The CUDB system is a distributed system, made of connected CUDB nodes, providing LDAP data access service. Every node provides an LDAP data access entry point, meaning an LDAP service connection by which LDAP data access is provided by such a node.

The internal CUDB network and node architecture are transparent to external LDAP users (network applications), as it is data distribution among different nodes.



This way, CUDB provides a Single Point of Access (SPoA) to distributed data, meaning that any LDAP entry point provides access to any data instance, regardless of data distribution and (to a certain extent) availability events.

Figure 1 shows an overview of SPoA.

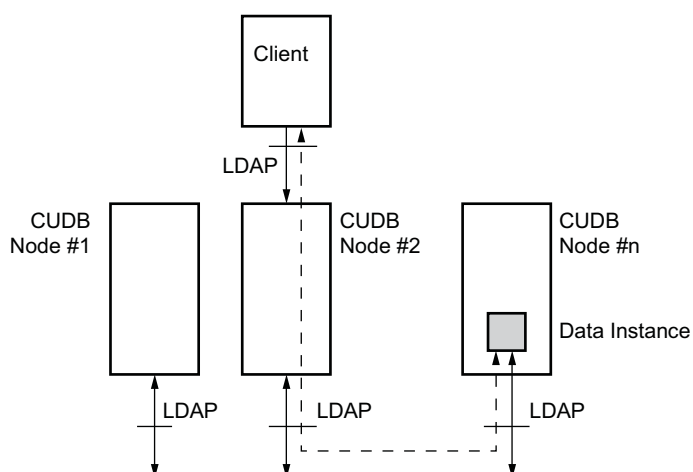


Figure 1 Single Point of Access

2.2.2

Data Allocation Schemes

A CUDB system distributes data in two different layers, the Processing Layer (PL) and the Data Store (DS) Layer as is shown in Figure 2.

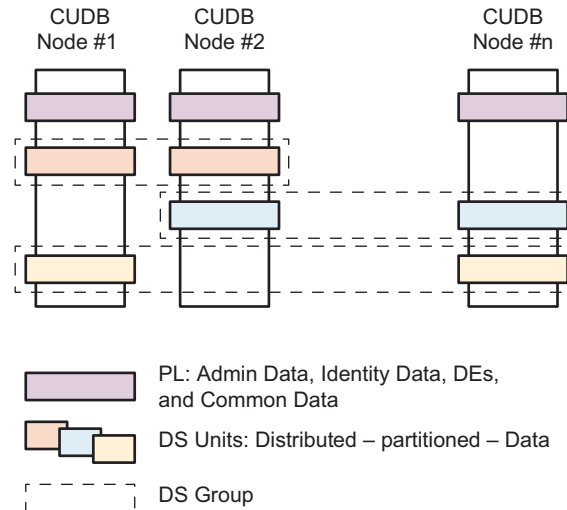


Figure 2 Data Distribution Layers

DS Layer: Distributed Data

Distributed data is stored in DS Units, implemented as a database cluster. Each DS Unit is placed in a CUDB node and possibly replicated in other nodes. All the replicas of a DS Unit form a DS Group (DSG). Distributed data typically consists of individual subscriber data.

Whenever replicated, one of the replicated DS Units is considered to be the authoritative source of data and is called the DS master replica and the rest are slave replicas.

PL: Non-Distributed Data

The non-distributed data is stored in Processing Layer Database (PLDB), which is implemented as a database cluster. Each node can contain one replica PLDB instance.

Non-distributed data includes the following:

- Administration data, including LDAP user information.
- Identities data, implemented by aliases.
- Distribution Entries (DEs).
- Common data, for example data needed to be accessed from distributed data.

For every CUDB site, at least one PLDB replica is required. Therefore, several replicas of the PLDB exist in the CUDB system (Refer to *CUDB Node*



Configuration Data Model Description, Reference [13] for more information). One of the replicated PLDBs is considered to be the authoritative source of data and is called the PLDB master replica, while the rest are called slave replicas.

2.3 Description

This section describes the information, service and security models of the LDAP data access, as well as the import and export function.

2.3.1 Information Model

This section describes the information model of the LDAP data access feature.

2.3.1.1 LDAP Schema

CUDB provides the required schemas used in the predefined entries that build up the LDAP basic DIT. Furthermore, the attributes and object classes included in the CUDB schemas can also be used by applications for their own entries.

Applications can also provide their own schema files that define specific object classes and attributes to be used by the applications data.

LDAP schemas are typically provided at installation time although CUDB also allows the possibility of including new LDAP schemas once the system is up and running. CUDB also supports updating existing LDAP schemas once an application is integrated in CUDB. For more information on schema update, refer to *CUDB LDAP Schema Management*, Reference [2] and *CUDB Node Schema Update*, Reference [5].

An LDAP schema in CUDB is made up of the following contributions:

- Core Schema: basic attributes and object classes defined in [RFC 4511 LDAP: The Protocol](#), Reference [23] and [RFC 4512 LDAP: Directory Information Models](#), Reference [24].
- CUDB Schema: internal definitions of internal usage, ranging from identity related definitions, LDAP client related definitions (for example, behavior on proxy operations, authentication related, and others), structural default object classes and so on.
- Application Schema: each LDAP client can add its own schema definitions, so that CUDB can support appropriate definitions. Additional definitions range for new identities, attributes or object classes.

2.3.1.2 CUDB Default Distributed Information Tree

CUDB is exposed as an LDAP directory to applications. CUDB provides a default LDAP Distributed Information Tree (DIT) to be extended by applications to store and query their data.



CUDB divides the DIT into the following regions:

- Root Entry: the first entry to consider in the CUDB. It is defined by the Relative Distinguished Name (RDN) from the root DSE (for details, refer to [RFC 4511 LDAP: The Protocol](#), Reference [23]) and adapted to customer needs at installation time by Ericsson personnel.
- Administration branch: holds CUDB users and groups of users allowed to access the system, together with their properties.
- Identities branch: holds the different identities defined in the system. Identities are mapped to aliases, allowing access to any distributed or non-distributed entry (Distinguished Name, DN) in the DIT except for the entries in the administration branch.
- DEs: refer to those entries in the DIT that hold data distribution information. Children of DEs are stored in the DS layer, distributed across different nodes. CUDB allows custom DEs to be defined.
- Distributed data: entries below DEs. Within the distributed data, CUDB supports the possibility to include different sub-entries at different levels. It also supports including extensible entries (`ei` attribute, see details in *CUDB LDAP Interwork Description*, Reference [6]) to define alias toward non-distributed data or distributed data within the same DS group. Alias toward distributed data in a different DS group is not supported.

The LDAP data access service accesses this branch directly, whenever the LDAP user request includes any distributed data as part of the received DN or when the distributed data is obtained after dereferencing alias when an identity is received.

- Common data branches: within the Common Data branches, CUDB supports the possibility to include different sub-entries at different levels. It also supports including extensible entries (including the `ei` attribute). Alias toward DEs and distributed data are not allowed.

For more information on how to define alias toward non-distributed data, see *CUDB LDAP Interwork Description*, Reference [6].

Figure 3 shows the regions as well as the correspondence of those regions with sample nodes.

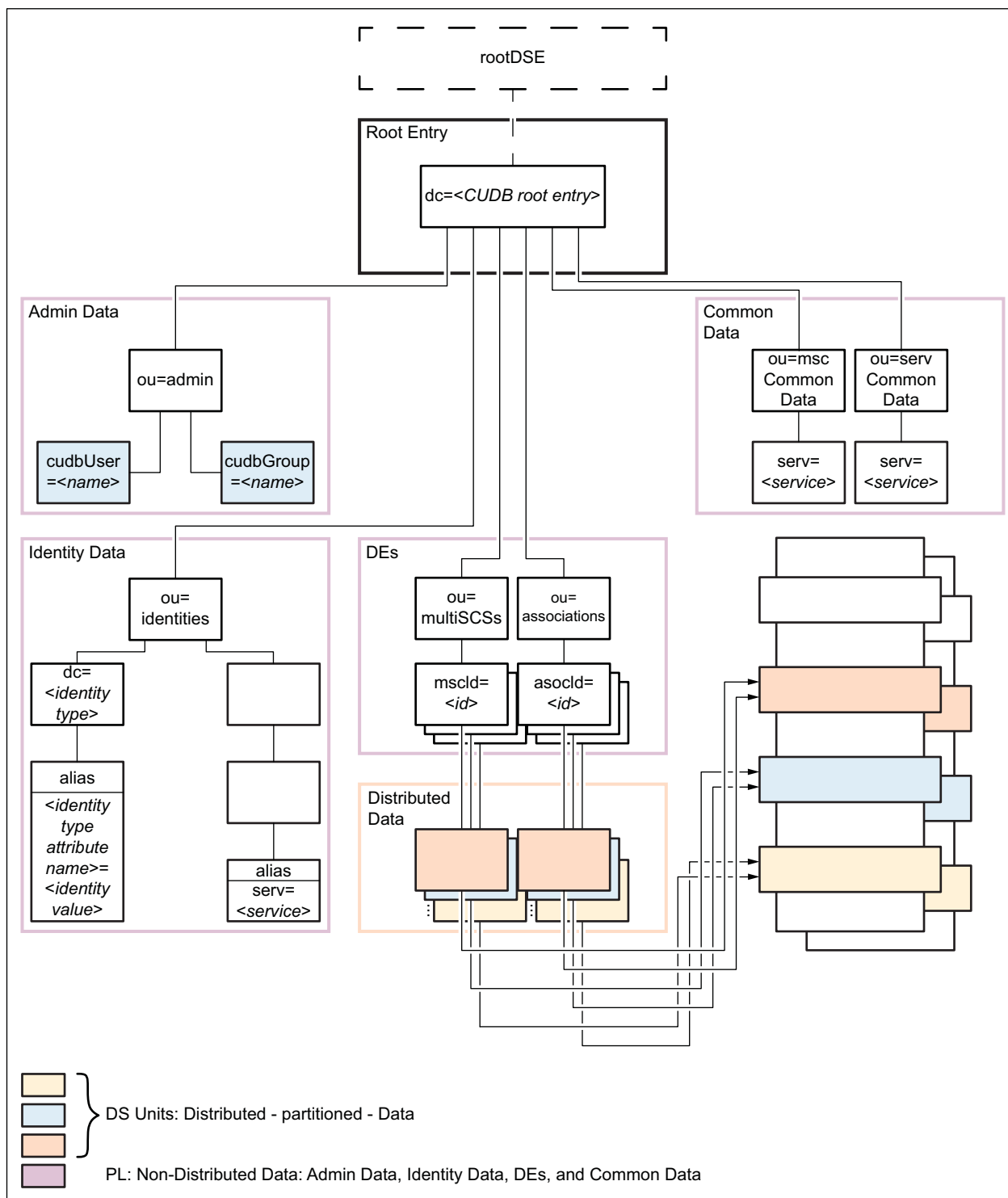


Figure 3 DIT Regions and Correspondence with Sample Nodes

In Figure 3, the different groups of distributed data are allocated regardless of DS units.

The data in Administration branch, Identities branch, DE, and Common Data branch is stored in the PLDB; and the distributed data is stored in the DS.

2.3.2 Default Directory Information Tree

Although the CUDB is prepared to support any LDAP trees that operators could request, by default it is installed with a built-in DIT. Figure 4 shows the default DIT.

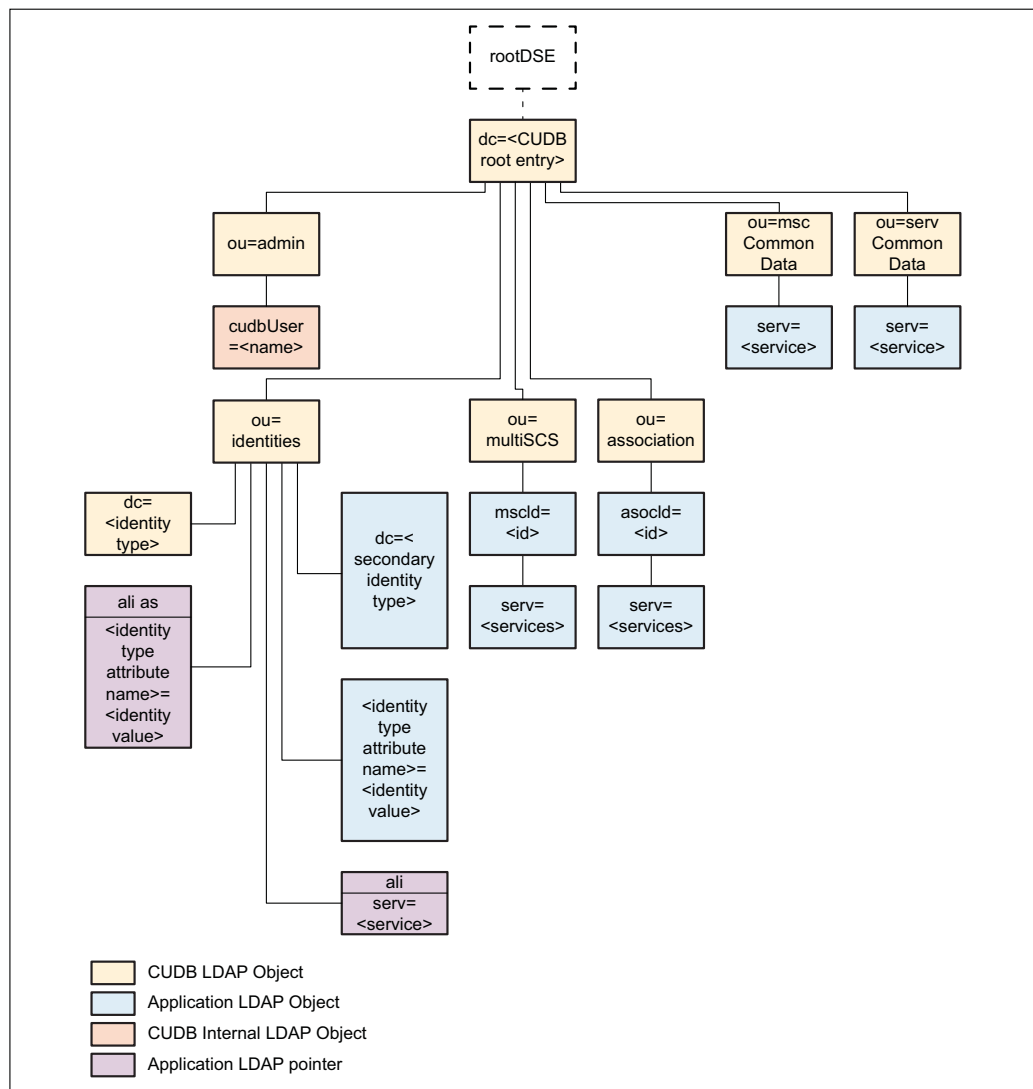


Figure 4 Default CUDB DIT

Refer to *CUDB LDAP Interwork Description*, , Reference [6] for additional details on the predefined CUDB DIT, regarding the available CUDB attributes and object classes in addition to those defined in the standard LDAP RFCs.



The following subsections include information on each of the main data entries within the default CUDB DIT in relation with the LDAP data access service.

2.3.2.1 CUDB Root Entry

The DN and RDN of the CUDB Root Entry (`dc=<CUDB root entry>`) are fully configurable during CUDB installation. For more information, refer to *CUDB LDAP Interwork Description*, Reference [6].

2.3.2.2 Administration LDAP User Data

Under the administration branch (`ou=admin, dc=<CUDB root entry>`), CUDB stores LDAP user information like authentication credentials and LDAP user configuration parameters. For more information on the LDAP user parameters see Section 3.1.3 on page 41.

The LDAP data access service can access this branch to authenticate the LDAP user (LDAP Bind request) and to handle the LDAP user request (that is LDAP Add) according to the LDAP user configuration parameters.

2.3.2.3 Identity Data

Under the Identity branch (`ou=identities, dc=<CUDB root entry>`), CUDB stores the identities used to identify and access other data entries that can be either distributed or non-distributed data. An example of identities to be stored under this branch are the network identifiers (IMSI, MSISDN) of the subscriber data in the different telecom networks, but any other type of identifier can be defined to access any other particular data type information.

The identities entries include an alias to the entry (DN) where the data is located.

Two types of identity entries are available, depending on how these entries are used:

- **Primary identities:** these entries are always aliases to another entry (typically to a DE). As primary identities serve this very specific purpose, they are treated as special entries in CUDB. Moreover, CUDB supports a non standard behavior of the LDAP Modify operations allowing alias dereferencing when the entry to modify contains a primary identity.
- **Secondary identities:** these entries are similar to primary identities in the sense that they can refer to other entries, but in this case, within the same secondary identities, different aliases are available for different services, therefore the alias is not included in the identity itself, but on the subentries below the identity.

Figure 5 shows an example of primary identity (`impi=2`) and another example of a secondary identity (`msisdn=1`).

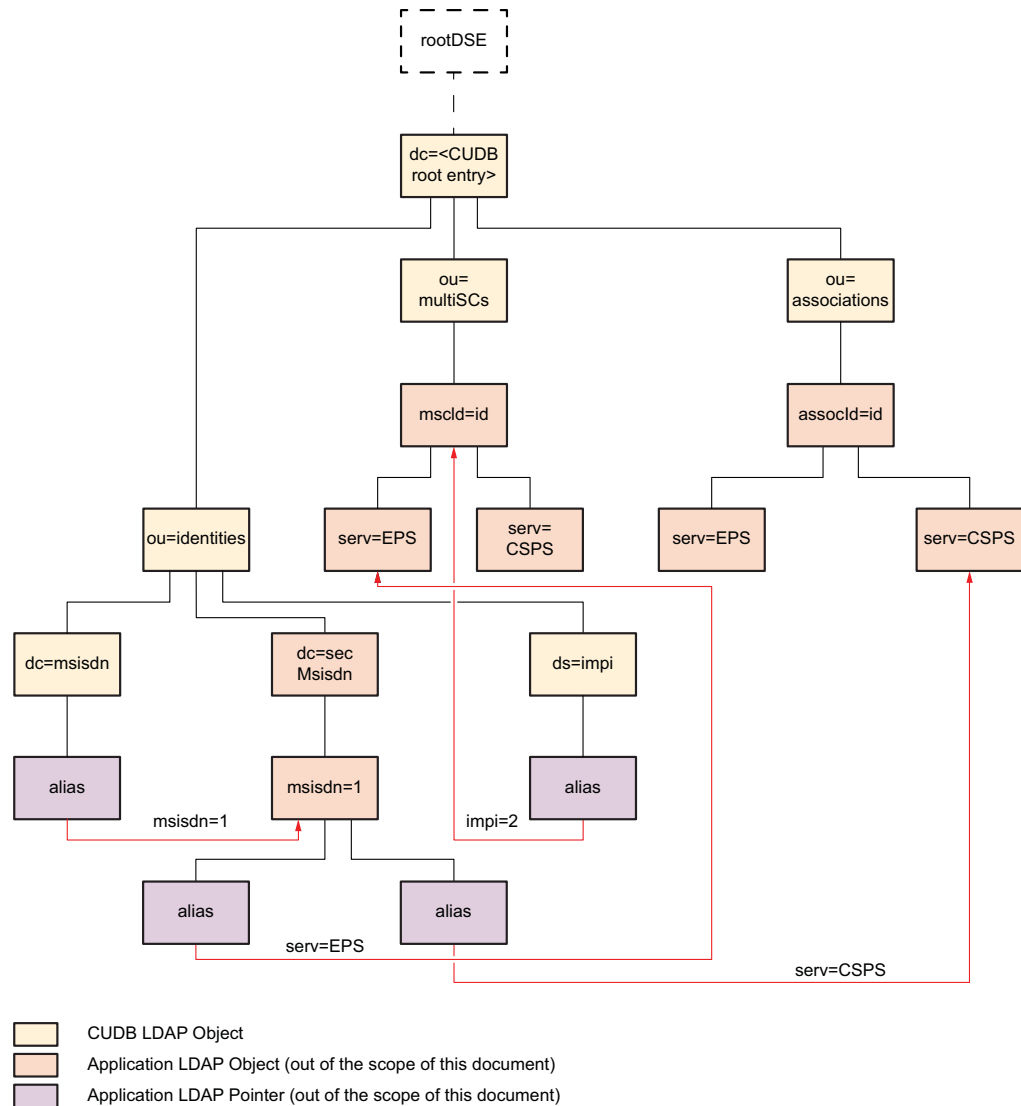


Figure 5 Primary and Secondary Identity Entry Examples

The primary identity entry (`impi=2`) includes a unique alias for that identity toward a DE (`mscid=id`), meaning that the data related to that identity can be found under the DE (`mscid=id`).

The secondary identity entry (`msisdn=1`) includes two aliases for that identity on its subentries. One alias included in `serv=EPS` pointing to a distributed data entry (`serv=EPS`, `mscid=id`) and another alias included in `serv=CSPS` toward another distributed data entry (`serv=CSPS`, `associd=id`) meaning that the data related to that identity can be found under two different distributed data entries.

The LDAP data access service accesses this branch whenever the LDAP user request includes the identity entry as part of the received DN.



2.3.2.4 Common Data

The CUDB default DIT includes two in-built common data branches:

- Multi Service Consumer (MSC) Common Data (`<ou=mscCommonData, dc=<CUDB root entry>`)
- Service Common Data (`<ou= servCommonData, dc=<CUDB root entry>`)

Usually, this type of data is common to several or most of the distributed data, meaning that it needs to be accessed from many or all DS groups in the system. Another reason for storing the data as non-distributed data is in case of very frequent access to this data with a very short delay.

As the common data is stored in all the CUDB nodes, only data with low memory demand is recommended to be a model for these entries.

Under the common data branches, the application can define their data model and attributes.

The MSC common data branch seems to be more appropriate to include common data profiles that apply to a group of distributed data but not common service wise. While the service common data branch is better to define data that applies to a certain service in general terms, for example, data related to default service rules.

2.3.2.5 DEs

The CUDB default DIT includes two built-in DE branches: Multi Service Consumer (`ou=multiSCs, dc=<CUDB root entry>`) and Association Data (`ou=associations, dc=<CUDB root entry>`), where the DEs can be defined (`mscID=<id>, ou=multiSCs, dc=<CUDB root entry>` and `assocID=<id>, ou=associations, dc=<CUDB root entry>`).

The parent entries of CUDB DEs are locked and cannot be deleted. CUDB will return Error Code 53 with the text message: Deleting parent of distribution entry is not allowed.

If those parent DE entries need to be deleted, contact the next level of Ericsson support.

2.3.2.6 Distributed Data

The distributed data can be accessed by a defined identity or directly by its DN.

An example of a distributed data entry is `serv=<service>, mscID=<id>, ou=multiSCs, dc=<CUDB root entry>`. Under the branches the application can define their data model and attributes.



For example, individual mobile or fixed subscriber data or individual session data can be stored as distributed data.

2.3.3 Service Model

LDAP data access follows standard LDAP behavior, except for variations introduced on LDAP modify and distributed LDAP search operations adapted to support real time operations in CUDB distributed directory for LDAP modify, and efficient access to distributed data for distributed LDAP search.

2.3.3.1 LDAP Modify Behavior

LDAP modify operation in CUDB is adapted to support transparent aliasing by following aliases defined in the identities subtree `ou=identities, dc=<CUDB Root Entry>`. As a result, LDAP modify operations transparently follow aliases defined in the identity subtree, that is, the operations are forwarded to destination nodes holding the affected distributed data.

2.3.3.2 Distributed LDAP Search

A distributed search is one where the data to retrieve is not sitting in one DSG but in several DSGs.

See Section 2.3.4.2 on page 20 for the details on which DSG replicas are accessed in a distributed search.

CUDB provides mechanisms to optimize distributed LDAP search queries by the definition of indexes. Indexes can be defined over one or more LDAP attributes belonging to the LDAP schemas used in CUDB. Procedures for describing search optimization indexes and related filtering behavior can be found in *CUDB Application Integration Guide*, Reference [4].

Search optimization with indexes is only applicable to one level and sub-tree search operations, except for identity entries where optimization is not supported.

When an LDAP search operation contains a filter in which one or several indexed attributes are involved, the search is a candidate for optimization if the filter meets specific conditions. Therefore, the selection of the attributes to index is an important task and must be made carefully by analyzing which queries require optimum performance that use filters too.

2.3.4 Processing LDAP Requests

LDAP requests enter CUDB through one of the nodes with PLDB but they might require processing at other CUDB nodes depending on which PLDB replica and which DSG replica –or replicas– must be accessed to process the request (see Section 2.3.6 on page 24). Regular LDAP requests might require processing in up to three CUDB nodes, as described in Section 2.3.4.1 on page



18. Distributed search requests might require processing at more than three CUDB nodes, as described in Section 2.3.4.2 on page 20. CUDB nodes without PLDB do not allow direct traffic and refuse these requests with error code 52, Node without PLDB can only receive proxy traffic.

Once an LDAP FE has chosen which replica (that is, which database cluster) to access, it does one of the following:

- Access the chosen replica straight away, if the chosen replica is hosted in the same node where the LDAP access happened (there are exceptions to this rule; see Section 2.3.4.2 on page 20) or
- Send a *proxy LDAP request* to the node where the chosen replica is, if the chosen replica is hosted in a different CUDB node. In these situations, the LDAP FE acts both as an LDAP *server* to receive LDAP requests from LDAP clients but also as an LDAP *client* when it issues LDAP requests of its own to be processed at other nodes in the CUDB system.

CUDB nodes can tell if an incoming LDAP request comes from an external LDAP client, or from another CUDB node.

Depending on the source of the LDAP request, the CUDB nodes can be classified as follows:

- Entry CUDB node: The node where the LDAP request from the external LDAP client is received.
- Terminating CUDB node: The node from where the pieces of data requested by an LDAP request coming from an external LDAP client are retrieved.
- Stepping-stone CUDB node: A node that also takes part in processing an LDAP request, but is not the entry or terminating CUDB node.

It is possible that a database cluster finds some trouble while processing a query. In those situations, the response to the failed LDAP request is 80, Error in PLDB in node <X>, DB Error Code:*error code* and *error message from the database* or 80, Error in DSG <X> in node <Y>, DB Error Code:*error code* and *error message from the database*, for errors in PLDB and in one of the DS clusters, respectively.

If the PLDB is down, or there is no PLDB in the node but the PLDB in the site is down, then LDAP FE answers with error code 52, CUDB node <X> is temporarily out of service.

It is possible that, in a distributed system like CUDB, some component goes down while the rest of the system still believes that component to be up and running. The behavior of CUDB in such situations is as follows:

- If the LDAP FE believes a local database cluster to be up and tries to access it, but that database cluster is down, the response to the failed LDAP request is 80, PLDB temporarily down in node <X> or

80, DSG <X> temporarily down in node <Y>, for trouble when accessing PLDB or one of the DS clusters, respectively.

- If the LDAP FE believes a remote node to be up and sends an outgoing proxy LDAP request to that node, but that node is down, the response to the failed LDAP request is 80, CUDB node <X> is temporarily out of service.

2.3.4.1

Normal Scenarios

Accessing Entries in PLDB

Figure 6 shows how PLDB entries are accessed in different scenarios.

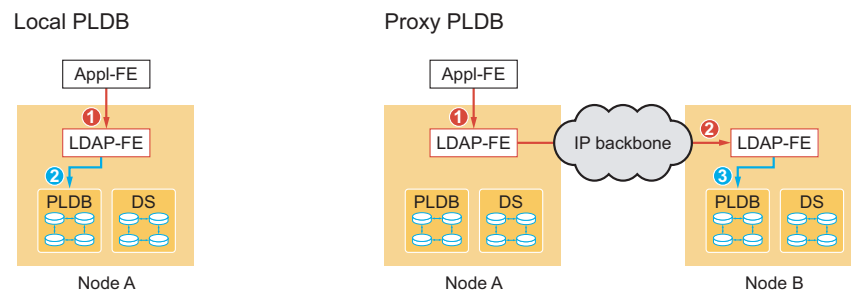


Figure 6 Accessing Entries in PLDB

Access to PLDB data is very straightforward because there is only one database to query. It is either be the PLDB replica in the entry CUDB node or the PLDB replica in a different CUDB node, the terminating CUDB node.

As mentioned earlier, an LDAP FE node will know when an LDAP request comes straight from an external LDAP client or from an LDAP FE in an entry CUDB that is, when it is an incoming proxy request.

Currently, all incoming proxy requests trying to access PLDB entries expect to access the master replica of PLDB.

When adding a DE to the LDAP tree, it is possible to state the geographical zone (see *CUDB Multiple Geographical Areas*, Reference [7]) or the specific DSG where the children LDAP entries are to be stored. If a non-existing geographical zone is stated, the response to the failed LDAP request is 19, Wrong Zone Assignment. If a non-existing DSG is stated, the response to the failed LDAP request is 19, Invalid DS Unit Group.

Accessing Entries in DSGs

Access to DS data is somewhat more complex than access to PLDB data because accessing data in a DSG requires a previous lookup in PLDB to



figure out which DSG or DSGs store the requested data (see Section 2.3.4.2 on page 20).

Figure 7 and Figure 8 show accessing entries in DSGs in different scenarios.

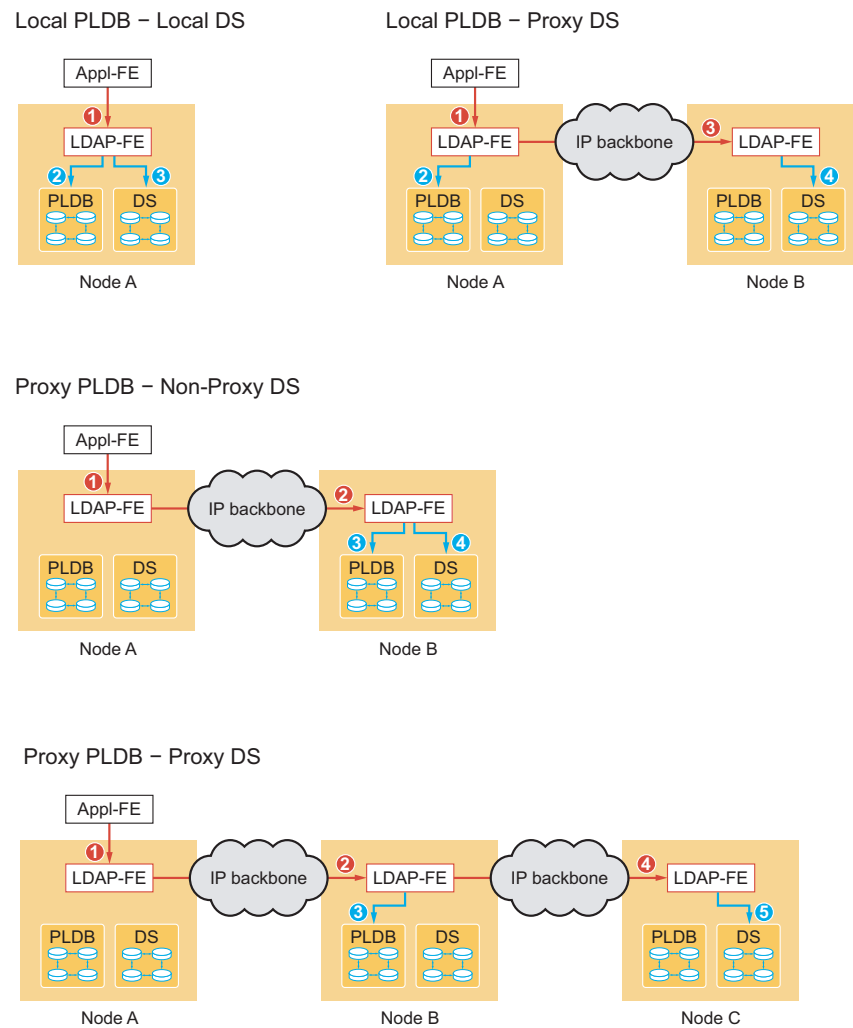
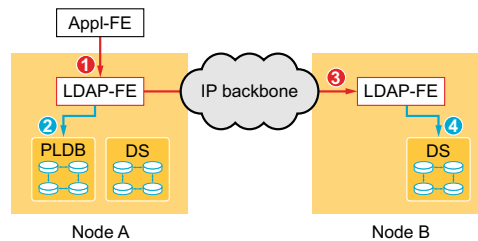


Figure 7 Accessing Entries in DSGs

Local PLDB – Proxy DS No PLDB



Proxy PLDB – Proxy DS No PLDB

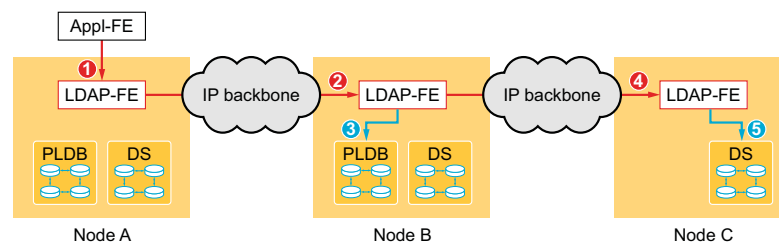


Figure 8 Accessing Entries in DSGs Where Master Replica Is Hosted by a Node without PLDB

Data in DSGs might include LDAP aliases pointing to entries stored in PLDB. If so, the PLDB data pointed to from DS data is retrieved from the PLDB replica hosted in the same node (or in case of a node with no local PLDB, from the PLDB in the same site) where the DSG replica is, irrespective of what the value of the `readModeInPL` parameter is.

Aliases pointing to data stored in a different DSG are not allowed. Aliases pointing to data in the same DSG are allowed and require no special consideration.

2.3.4.2

Special Scenarios

Deleting DEs

Figure 9 shows deleting DEs in different scenarios.

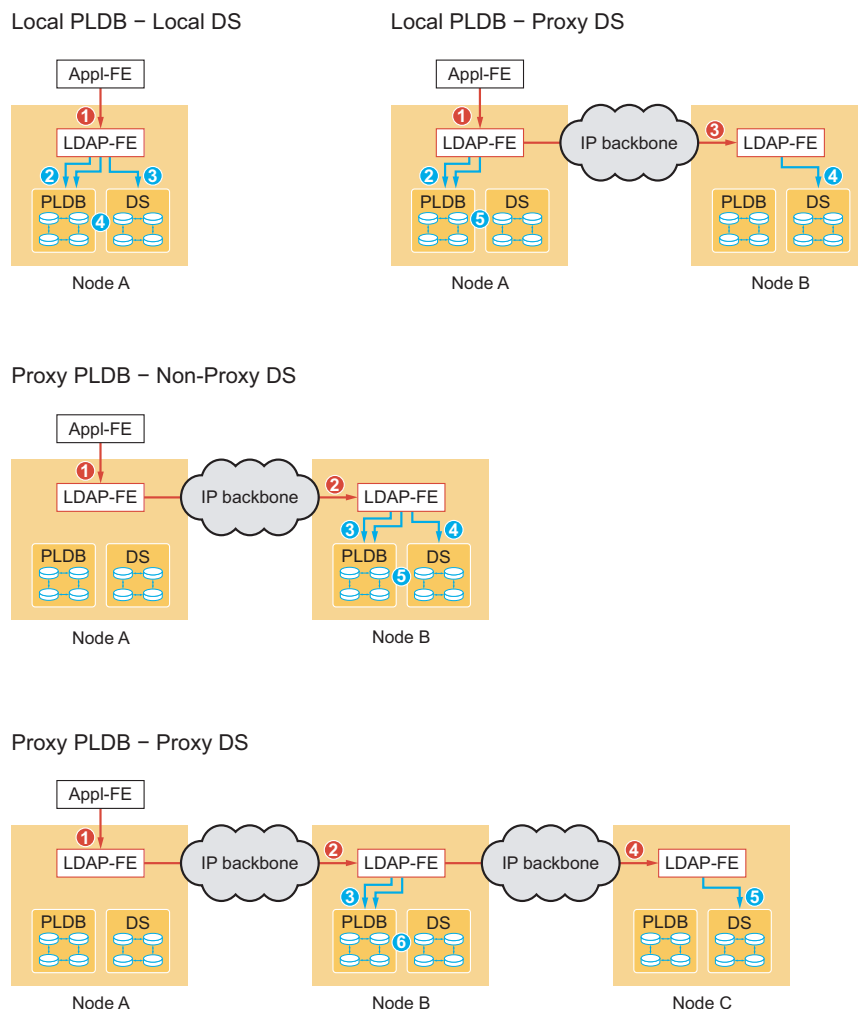


Figure 9 *Deleting DEs*

When deleting a DE, CUDB must first verify that the DE to delete has no child entries. Since DEs are located in the PLDB but right at the frontier between PLDB and DSGs, accessing the master PLDB replica is not enough (as delete is also considered a write operation). Before that entry is deleted, CUDB must perform a DSG lookup at the master replica of the corresponding DSG, irrespective of the `readModeInDs` setting configured for the LDAP user. Therefore, the sequence is as follows:

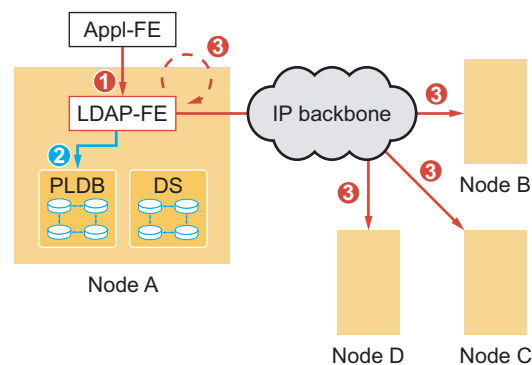
- 1 Lookup at the master PLDB replica to figure out which DSG stores (or would store) anything below that DE.
- 2 Lookup at the master replica of the corresponding DSG to verify that there are no children to the DE to be deleted.

3 Deletion at the master PLDB replica.

Distributed Search

Figure 10 shows distributed search in different scenarios.

Local PLDB – Proxy DS



Proxy PLDB – Proxy DS

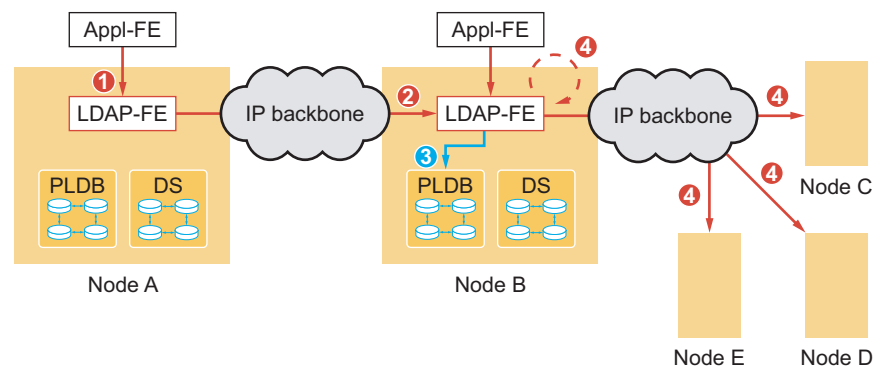


Figure 10 Distributed Search

A distributed search is one where the data to retrieve is not sitting in one DSG but in several DSGs. The LDAP FE doing the PLDB lookup notices it, and choose which DSG replicas to use in which CUDB nodes. It retrieves the required data from PLDB, then send LDAP queries to all the nodes hosting one or more of the selected DSG replicas. Notice that any of the selected DSG replicas is hosted in the same node where the PLDB lookup is executed, they still will be handled through a “self-proxy” LDAP request, instead of directly accessing the database cluster (hence the dotted arrow in Figure 10).

The criteria to select the DSG replicas to use is as follows:

- Select a random slave replica, if one is available, otherwise follow step b.



- b Select the master replica, if one is available, otherwise follow step c.
- c Declare the query unanswerable and return `53, DSG <X> not reachable`, where `<X>` is the first DSG for which no replica could be found.

These outgoing proxy queries are sent all at once, in parallel, to all the nodes hosting one or more of the selected DSG replicas.

Upon receiving this LDAP proxy request, the terminating LDAP FE accesses the corresponding DS clusters one by one, in sequence, and retrieves the requested data. The LDAP FE sends data back as it is retrieved from the databases, that is, it does not wait to have all data from all the involved DS clusters and then send it back to the proxying LDAP FE (either the entry LDAP FE or the stepping-stone LDAP FE).

If the terminating LDAP FE finds any problem, it immediately stops sending data, returns the appropriate error response to the proxying LDAP FE, and stops any further processing. This means that if some DSG replicas/DS clusters are still supposed to be accessed in that node that have not been accessed yet, they are not accessed at all.

The stepping-stone LDAP FE (if any) passes results through to the entry LDAP FE as it receives them from the terminating LDAP FEs. If the stepping-stone LDAP FE receives an error message from a terminating LDAP FE, it passes it through to the entry LDAP FE and considers the operation as finished, dropping further responses from other terminating LDAP FEs. Those other terminating LDAP FEs keep on processing their queries even if they are now useless. If any of the terminating LDAP FEs are in unavailable state, the distributed query will fail with `error 80` and diagnostic message `Distributed search failed`.

The entry LDAP FE passes results through to the LDAP client as it receives them from the stepping-stone LDAP FE or the terminating LDAP FEs. If the entry LDAP FE receives an error message, it passes it through to the LDAP client and considers the operation as finished. If the error message comes from a terminating LDAP FE, the entry LDAP FE drops further responses from other terminating LDAP FEs. Those other terminating LDAP FEs keep on processing their queries even if they are now useless.

2.3.4.3 Handling Responses to Outgoing Proxy Requests

Results from outgoing proxy requests are forwarded to the LDAP client that made the original request as they are received.

Typically, LDAP FEs just forward response messages from outgoing proxy requests to the LDAP client that made the original request. Some response messages are translated before they are sent to the original LDAP client. The response messages that are translated are as follows:

- `51, LDAP server overloaded in node <X>` is translated into `80, LDAP server overloaded in node <X>`.

- 51, PLDB overloaded in node <X> is translated into 80, PLDB overloaded in node <X>.
- 52, No available master replica for PLDB is translated into 80, No available master replica for PLDB.
- 52, CUDB node <X> is temporarily out of service is translated into 80, CUDB node <X> is temporarily out of service.

Translations happen at the stepping-stone LDAP FE (if any) or at the entry LDAP FE.

2.3.5 Master Election

Figure 11 shows the system behavior regarding master replica availability and election.

Note: The "Planned master election ongoing" state is related to the Manual Mastership Change and Automatic Mastership Change functionalities.

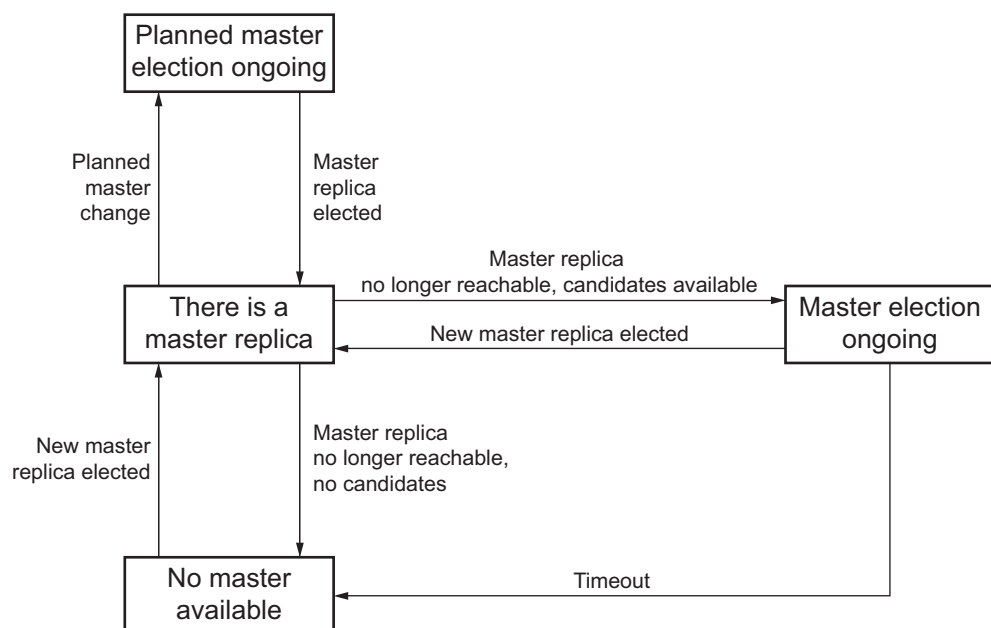


Figure 11 Master Election State

2.3.6 Replica Selection

The LDAP user configuration parameters and the specific replica availability situation of the CUDB system determine which replicas of PLDB and of the relevant DSG(s) are accessed. The relevant LDAP user configuration



parameters are `readModeInPL`, `readModeInDS`, and, to a lesser extent, `isProvisioningUser` and `isReProvisioningUser`. See Section 3.1.3 on page 41 for further details on LDAP user configuration parameters. LDAP user read mode configuration settings can be overridden for any single search request by using the `ReadMode` LDAP control. For more information, refer to *Section 6.4.1 ReadMode Control, CUDB LDAP Interwork Description*, Reference [6].

LDAP FEs receive information about the master and slave replica availability and partition status information from the Data Availability Coordination function (refer to *CUDB High Availability*, Reference [8]) and try to provide as much service as possible to LDAP clients.

As a general rule, LDAP FEs pay no attention to the partition status information (for example, majority, minority, symmetrical split) to select replicas although there are some exceptions to this rule (see Section 2.3.6.2 on page 31).

2.3.6.1

Normal Scenarios

This section lists the normal scenarios of replica selection.

PLDB Situations

Figure 12 shows all the possible PLDB replica availability scenarios, as seen by an LDAP FE.

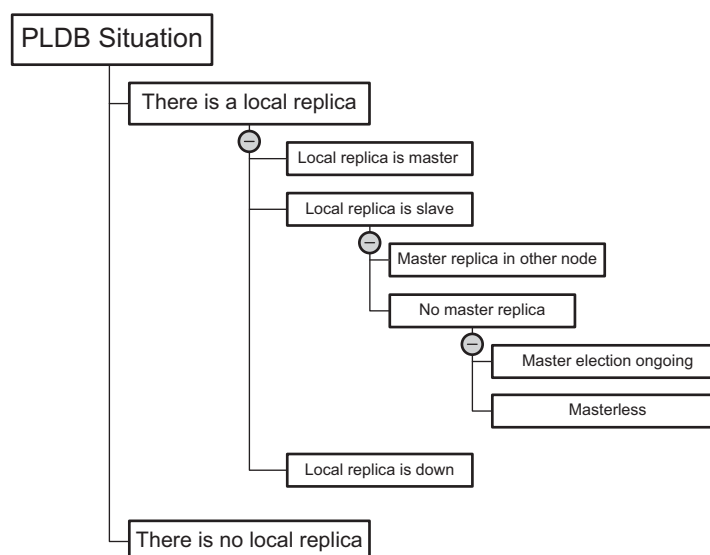


Figure 12 Possible PLDB Replica Availability Scenarios

All LDAP requests coming from an external LDAP client require an access to PLDB. Nodes without PLDB do not accept direct traffic and reject any request

with Error code 52. Incoming proxy operations may or may not require an access to PLDB.

Read Requests on PLDB Entries

Figure 13 shows the replica selection criteria, the LDAP result code, and diagnostic messages when no replica can be selected.

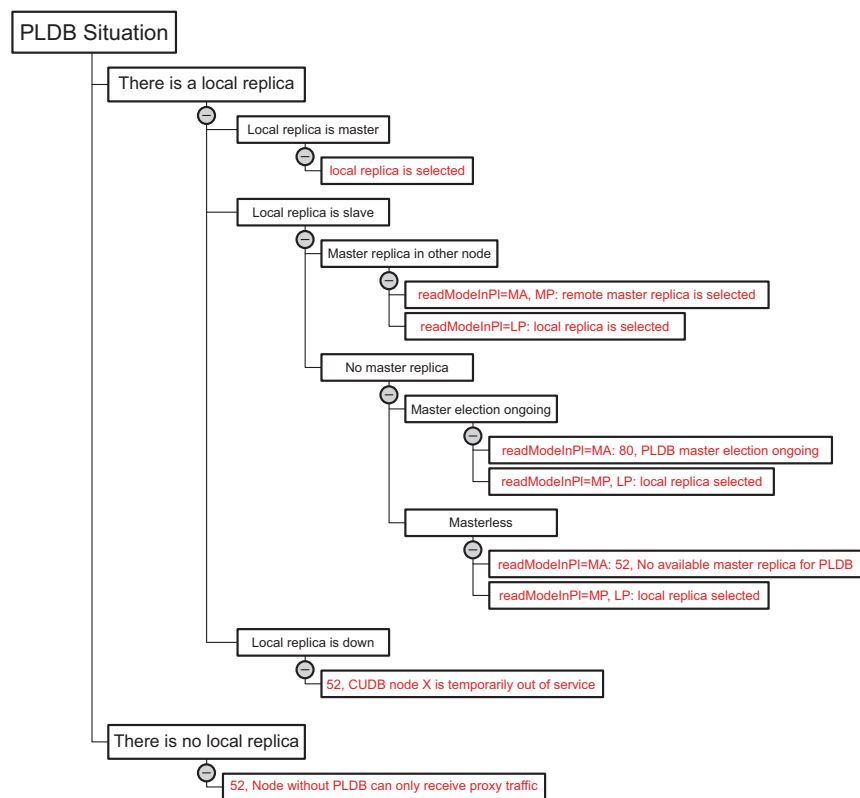


Figure 13 PLDB Replica Selection for Reading: Criteria, Error Codes and Diagnostic Messages

Write Requests on PLDB Entries

Figure 14 shows the replica selection criteria, the LDAP result code, and diagnostic messages when no replica can be selected.

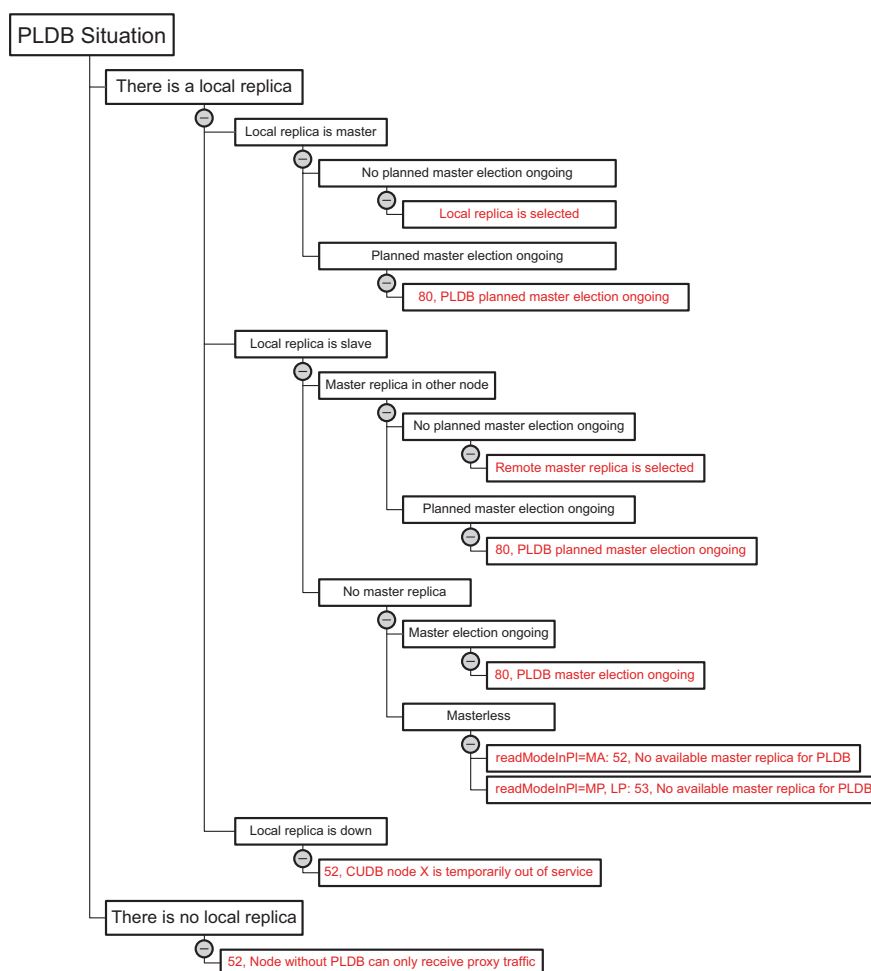


Figure 14 PLDB Replica Selection for Writing: Criteria, Error Codes and Diagnostic Messages

DSG Situations

Figure 15 shows all the possible DSG replica availability scenarios, as seen by an LDAP FE.

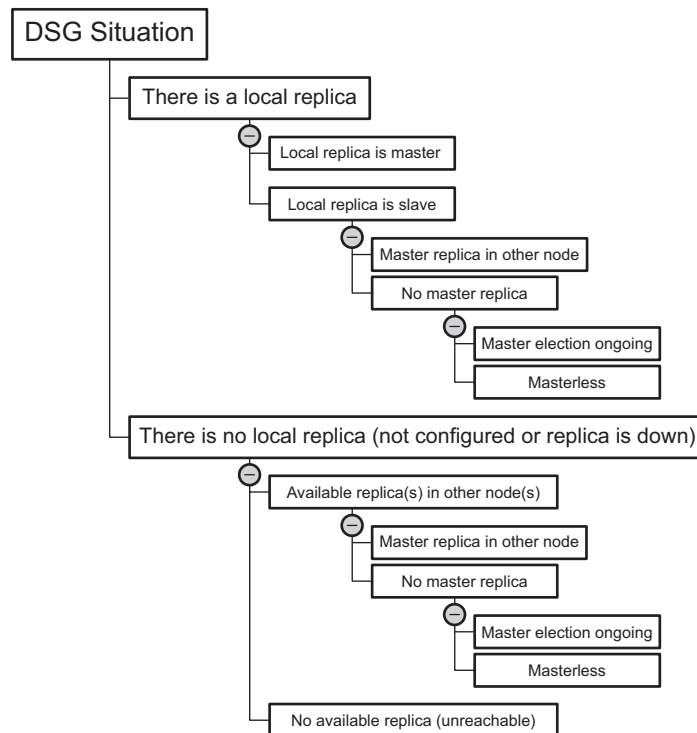


Figure 15 Possible DSG Replica Availability Scenarios

Read Requests on DS Entries

Figure 16 shows the replica selection criteria, the LDAP result code, and diagnostic messages when no replica can be selected.

A neighboring replica is one which is hosted in a CUDB node in the same site as the CUDB node where the LDAP FE runs.

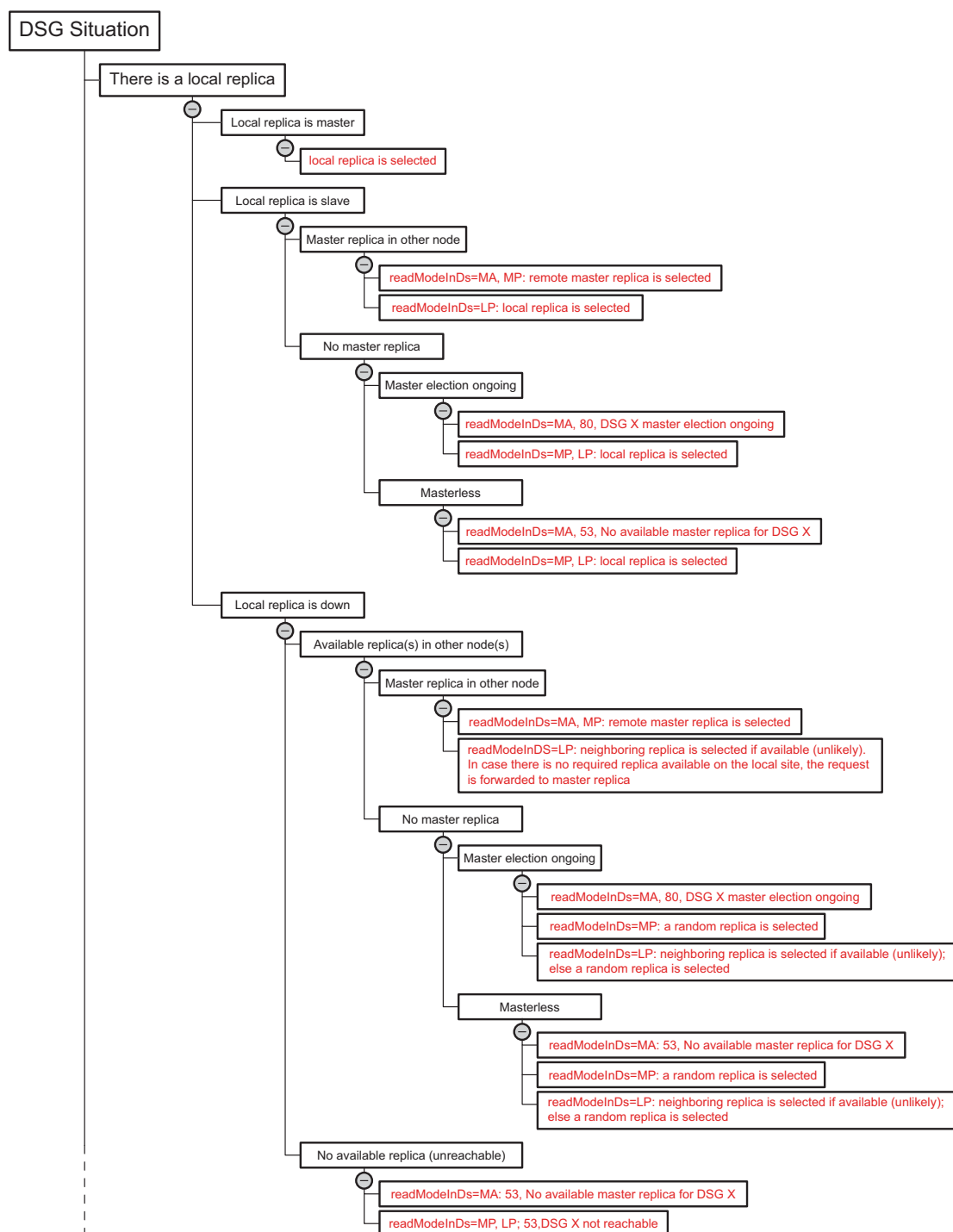


Figure 16 DSG Replica Selection for Reading: Criteria, Error Codes, and Diagnostic Messages



Write Requests on DS Entries

Figure 17 shows the replica selection criteria, the LDAP result code, and diagnostic messages when no replica can be selected.

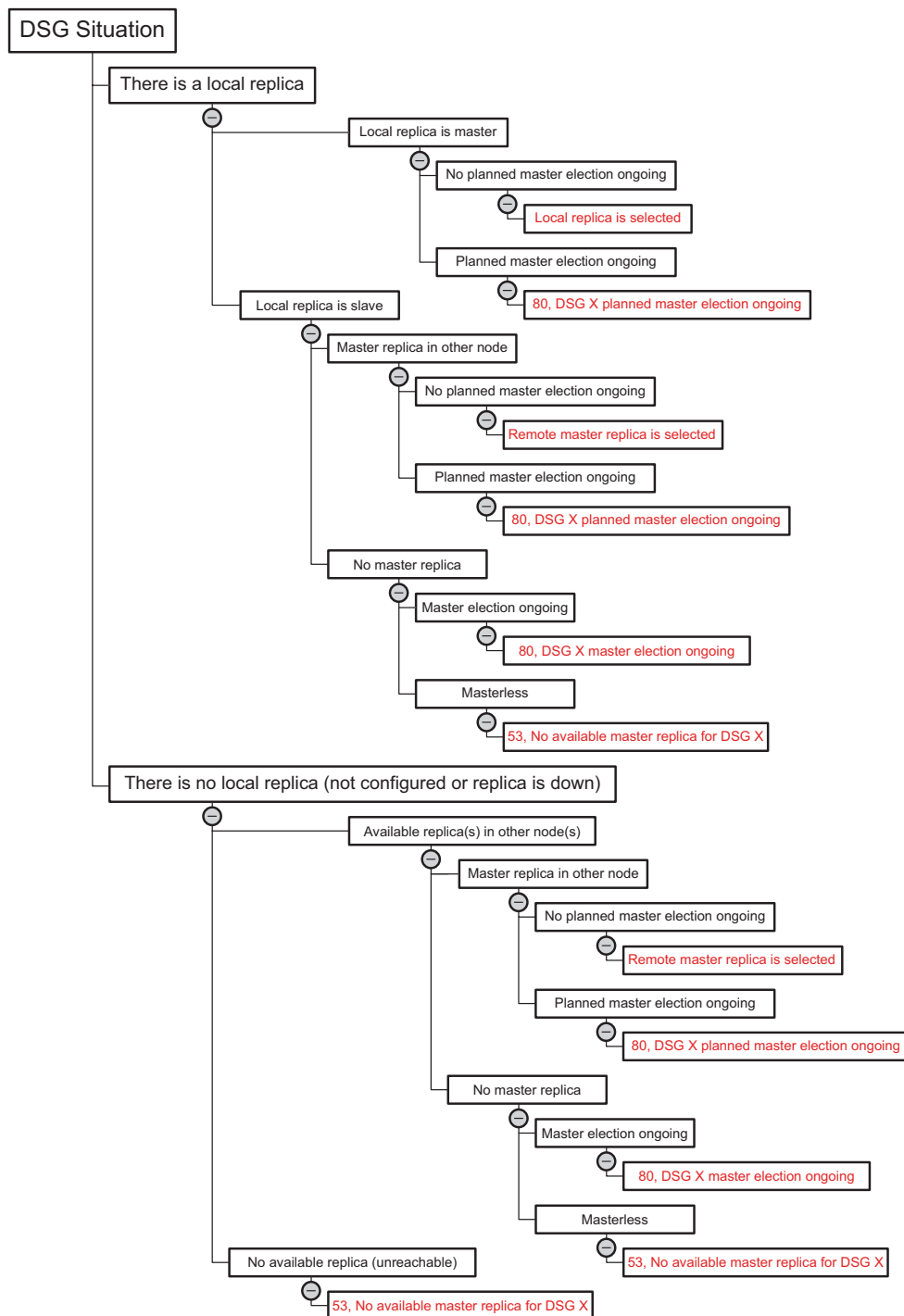


Figure 17 DSG Replica Selection for Writing: Criteria, Error Codes, and Diagnostic Messages



2.3.6.2 Special Scenarios

Provisioning Write Requests While in Potential Symmetrical Split Scenarios

Provisioning operations are identified by the LDAP user that issues them: all LDAP requests coming from users configured with `isProvisioningUser=TRUE` (see Section 3.1.3 on page 41) are considered to be provisioning operations.

Write operations are forbidden for LDAP users whose `isProvisioningUser` parameter is set to `TRUE`, while the CUDB system is in a potential symmetrical split situation in the sites within the network partition not hosting the master PLDB replica. In these cases, the 53, Potential split-brain situation. Write operations forbidden for provisioning users error message is returned. For more information on symmetrical split situations, refer to *CUDB High Availability*, Reference [8].

Provisioning Write Requests While in Minority with at Least One DSG Master Replica

A minority partition may still hold DSG masters if any of the following scenarios occur:

- The Service Continuity function is activated in the minority partition, and therefore DSG masterships have been acquired in the minority partition.
- For some DSGs, all their replicas are located within the minority partition, and none of their replicas exist in the majority side (refer to the *Subsystem in a Minority Situation* bullet of *CUDB High Availability*, Reference [8] for further details).

In both cases, write requests coming from an LDAP provisioning user (that is, from users whose `isProvisioningUser` parameter is set to `"true"`) are not allowed all over the minority partition, even if it retains or acquires DSG masters. The rejected write operations return LDAP error code 53, with the message "Minority with master DSGs situation. Write operations forbidden for provisioning users".

Operations from LDAP Users Configured with `readModeInPL` Access Node Set to "Master Always (MA)", and No PLDB Master Replica is Available within the Same Partition

In case of a minority situation and if the master PLDB replica is not in the current partition, all operations from users with `readModeInPL=MA` are rejected with LDAP error code 52, with the message "No available master replica for PLDB".



Provisioning Write Requests When Provisioning Assurance Function Is Configured

In case the Provisioning Assurance function is configured in the CUDB system (refer to *CUDB Node Configuration Data Model Description*, Reference [13] for further details), CUDB initiates a Provisioning Assurance event after any unplanned mastership change (refer to *CUDB High Availability*, Reference [8] for further details).

During Provisioning Assurance event execution:

- Write provisioning operations, those coming from LDAP users configured with `isProvisioningUser=TRUE` and `isReProvisioningUser=FALSE` (see Section 3.1.3 on page 41 for further details) are not allowed. In this case, LDAP error code 16384, Provisioning Blocking currently active. Write operations rejected for provisioning users is returned.
- Write re-provisioning operations, those coming from users configured with `isProvisioningUser=TRUE` and `isReProvisioningUser=TRUE` (see Section 3.1.3 on page 41 for further details) are allowed.

When Provisioning Assurance event is not running:

- Write re-provisioning operations (all LDAP write requests coming from users configured with `isReProvisioningUser=TRUE`, see Section 3.1.3 on page 41 for further details) are not allowed. In this case, LDAP error code 16385, Provisioning Blocking currently inactive. Write operations rejected for re-provisioning users is returned.
- All other write operations are allowed.

Minorities Without Any DSG Master Replica

In certain scenarios, even if the replica availability situation seen from a CUDB node would still allow the LDAP FEs to provide some level of service to the external LDAP clients, this provided level of service is too limited to be usable. Connecting external LDAP clients to a different CUDB node for a better level of service is recommended.

As explained in *CUDB LDAP Interwork Description*, Reference [6], the way to suggest external LDAP clients to connect to a different CUDB node is by sending error code 52 responses. Related scenarios are minority scenarios where no master replica for any DSG is reachable.

In these cases, LDAP FEs answer any LDAP request, including LDAP BIND, with the 52, No available master replica for any DSG error message, irrespective of the value of the `readModeInPL` and `readModeInDS` parameters.



Note: This behavior applies only to strict minority and neither to their Take-All-Masters (TAM) nor Take-Service-In-Minority equivalents. Refer to *CUDB High Availability*, Reference [8] for further details.

2.3.7 CUDB Node Overload Protection

Overload protection and load regulation in CUDB takes place in the following four separate but related areas:

- LDAP FE Self-Protection.
- PLDB Cluster Self-Protection.
- DSG Cluster Self-Protection.
- LDAP FE-to-Database Cluster Collaborative Protection.

The above mentioned four areas are described in detail in the following sections.

2.3.7.1 LDAP FE Self-Protection

If the LDAP FE receiving the LDAP request is overloaded, it immediately answers with an error message: 51, LDAP server overloaded in node <X>.

2.3.7.2 PLDB Cluster Self-Protection

If the local replica of PLDB is overloaded, LDAP requests that require an access to the local replica PLDB to be processed are answered with an error message: 51, PLDB overloaded in node <X>. DB Error Code <Y>: <NDB API error text>.

2.3.7.3 DSG Cluster Self-Protection

If the local replica of a DSG that must be accessed to process an LDAP request is overloaded, that LDAP request is answered with an error message: 80, DSG <X> overloaded in node <Y>. DB Error Code <Z>: <NDB API error text>.

There is a special scenario where a different error code is returned. If the local replica to access is the only existing DS replica in the node, and it is overloaded, the LDAP request is answered with the 51, DSG <X> overloaded in node <Y>. DB Error Code <Z>: <NDB API error text> error message. This signals the LDAP client that the node resources (one unique DS in the node) are overloaded.

2.3.7.4 LDAP FE-to-Database Cluster Collaborative Protection

LDAP FEs use feedback from local database clusters (PLDB, DSGs) to notice when they do not have enough available resources to process the



incoming traffic. LDAP FEs take action to ease the load on overloaded clusters by pre-emptively rejecting LDAP requests that target an overloaded local cluster. LDAP requests that would require accessing the local instance of the overloaded PLDB cluster are pre-emptively rejected by the LDAP FEs, and are answered with the 51, PLDB overloaded in node <X> error message.

LDAP requests that would require accessing a local, overloaded DSG cluster, and are pre-emptively rejected by the LDAP FEs are answered with the 80, DSG <X> overloaded in node <Y> error message.

There is a special scenario where a different error code is returned. LDAP requests to access the only existing local DSG cluster in the node, while it is overloaded and pre-emptively rejected by LDAP-FE, are answered with the 51, DSG <X> overloaded in node <Y>. DB Error Code <Z>: <NDB API error text> error message.

Due to the random distribution of the inter-arrival times of the incoming traffic, it can happen that the system scatters small bursts of 80, DSG <X> overloaded in node <Y> error messages when incoming rates are below the expected capacity. The current configuration parameters however would maintain the amount of errors below 0.01% with the normal traffic patterns when the traffic is up to the capacity of the clusters. However, if the incoming traffic and the inter-arrival time probability follows a highly burst pattern, more frequent bursts of 80, DSG <X> overloaded in node <Y> errors can occur, which can decrease the Quality of Service (QoS) below the expected value of 99.99%.

2.3.8 Traffic Prioritization per Application Under Overload

The Traffic Prioritization per Application Under Overload feature in CUDB is built on top of the foundation provided by LDAP FE-to-database cluster collaborative protection. Thus, traffic prioritization only kicks in when local database clusters are overloaded and only affects traffic that would require accessing those overloaded database clusters.

CUDB defines five priority levels –1 to 5– which can be assigned to LDAP users by the attribute `overloadRejectionWeight` (refer to *CUDB Node Configuration Data Model Description*, Reference [13]). Top priority is priority number 1. If no priority is assigned to an LDAP user, the default priority for that user is 1. LDAP root user cannot be assigned a priority, it is hardwired to priority 1.

Note: The priority change can be performed at any time without latency, **even during an overload situation**. The effects of changing the priority level online are immediate.

When database clusters are overloaded, LDAP FEs will be more aggressive at pre-emptively rejecting LDAP requests from lower-priority LDAP users, effectively prioritizing down the traffic from these users as compared to traffic from higher-priority LDAP users.



Under overload situation, the ratio of processed LDAP requests vs. total incoming LDAP requests for the higher-priority application FEs is always higher than the ratio for the lower-priority application FEs.

LDAP requests rejected by LDAP FEs due to traffic prioritization get the same error messages used for LDAP FE-to-database cluster collaborative protection (see Section 2.3.7.4 on page 34).

See Section 4 on page 47 for more information on configuring traffic prioritization.

2.3.9 Unresponsive Remote Node

Proxy LDAP operations in CUDB are synchronous, which means that resources in the proxying LDAP FE stay busy until the proxy operation ends. Proxy LDAP requests addressed to unresponsive nodes take a long time to finish (until they time out), and keep resources at the proxying LDAP FE busy during the process. To prevent resource exhaustion at the proxying LDAP FE, no more than one execution thread will be allocated to process LDAP proxy requests addressed to a CUDB node that is considered to be unresponsive.

When that one execution thread is busy, the LDAP FE immediately answers further incoming LDAP FEs meant to be processed at the unresponsive node with result code 80, `Unresponsive remote node <X>`.

The LDAP FE considers a remote node unresponsive if 20 LDAP requests addressed to it time out in a row. A remote node is considered to stay unresponsive while succeeding LDAP requests addressed to it time out.

The LDAP FE considers a remote node to be no longer unresponsive when one LDAP request addressed to it is processed on time (that is, it does not time out).

Note: The unresponsive remote node information does not affect the replica availability situation used on replica selection.

Therefore, if the LDAP FE processes requests from LDAP users configured with user modes that allow a certain degree of flexibility in replica selection (such as Master Preferred or Local Preferred modes), the LDAP FE does not use the information it may have on unresponsive remote nodes to discard certain replicas from election.

2.3.9.1 Time Limits

LDAP FEs set an administrative time limit of 2 seconds to all received LDAP requests, including add, delete and modify requests. For search requests, this administrative time limit overrides the `timeLimit` setting in the original request if the `timeLimit` specified in the search request is higher than the administrative time limit.

The administrative time limit does not apply to distributed search operations.



2.3.10 Fairness Control and Database Cluster Watchdog Function

The fairness control and database cluster watchdog function helps the CUDB system keep processing traffic in case of trouble with a local database cluster. This function prevents LDAP FEs and traffic from asynchronous connections from being bogged down if one of the database clusters in the CUDB node is slow to respond by limiting the number of execution threads assigned to these tasks. Following are different cases when maximum number of threads are reached:

- If the maximum number of threads that access the local PLDB in an LDAP FE is reached, that LDAP FE answers further requests to access the local PLDB with the 51, Max number of threads reached for PLDB error message.

Note: Error 51, which is reserved for overload, is used in this case even though the CUDB node is not necessarily overloaded. The reason is that overload in the PLDB is a probable cause for reaching the maximum number of threads for the PLDB.
- If the maximum number of threads that access a local DSG replica in an LDAP FE is reached, that LDAP FE answers further requests to access that DSG replica with the 11, Max number of threads reached for DSG <X> error message.
- If the maximum number of threads that access a local DSG replica through an LDAP connection is reached, the LDAP FE handling that connection answers further requests to access that local DSG replica with the 11, Max number of threads reached for connection <X> and DSG <Y> error message.

2.3.11 CUDB Import and Export

Import and export procedures are used to load or extract high amounts of data in CUDB. The import and export procedures are executed on specific data partitions – PLDB, DSGs. These procedures use LDAP Interchange Format (LDIF) files as input and output, respectively, instead of using the LDAP protocol.

2.3.11.1 Import and Export Commands

The LDAP commands used to import and export data are as follows:

- `slapadd`: Import command, used to add entries to a database. It is executed on master replicas of data partitions. The input of the `slapadd` command is an LDIF file. Import operations can run simultaneously on several data partitions.
- `slapcat`: Export command which extracts data from a specific data partition – PLDB or one of the DSGs. The `slapcat` command supports search with filters, scope and other options. The output of this command



is an LDIF file. Export operations can run simultaneously on several data partitions.

For more information about import and export procedures in CUDB, refer to *CUDB Import and Export Procedures*, Reference [9].

2.3.12 LDAP Error Code Precedence

The general criteria about LDAP error code precedence is as follows:

- a LDAP FE overloaded.
- b Local PLDB is down, or no local PLDB in the node and site PLDB is down.
- c Minorities without any DSG master replica.
- d Provisioning write requests while in potential symmetrical split scenarios.
- e Provisioning write requests during Provisioning Assurance event.
- f Re-provisioning write requests in case Provisioning Assurance event is not running.
- g Issues found while choosing the PLDB replica to access (if needed).
- h Issues found while accessing the PLDB replica (if needed) which can also include the following:
 - Issues with the fairness control and database cluster watchdog function.
 - Issues with request processing.
- i Issues found while choosing the DSG replica(s) to access (if needed).
- j Issues found while accessing the DSG replica to access (if needed) which can also include the following:
 - Issues with the fairness control and database cluster watchdog.
 - Issues with request processing.

2.4 Dependencies and Interactions

The CUDB LDAP data access feature has interactions mainly with the following features:

- LDAP Schema Management: LDAP schemas are used by the LDAP data access feature and the LDAP schemas loaded in CUDB are considered to allow or reject LDAP operations implying adding or modifying data (object class and attributes) into a data entry. For more information on schema management, refer to *CUDB LDAP Schema Management*, Reference [2].



- **Data distribution:** It is responsible for the data distribution along the CUDB system. When an LDAP add operation is received to create a new DE in the DIT, the LDAP distribution feature responsible for selecting the corresponding DS group where the distributed data is located. For more information on data distribution, refer to *CUDB Data Distribution*, Reference [10].
- **Availability:** It is responsible for the monitoring of the availability of the CUDB local resources including fault handling, recovery and data reconciliation. The availability feature handles the status of the PLDB and DS unit groups and the location of the needed master or replica copy in the CUDB system. In case the needed resources are available, LDAP data access feature proceeds with the received LDAP operation in the corresponding PL or DS resource. Otherwise, the LDAP data access feature answers back the request with the appropriate LDAP error code as detailed in Section 2.3 on page 9. For more information on availability, refer to *CUDB High Availability*, Reference [8].
- **Storage:** It is responsible for the data storage, its redundancy and replication. When an LDAP operation is received acting on the DIT (create or delete any data entry or to add, modify, or delete any attribute) the storage is responsible for the handling of the request related to the data storage and the data replication. In case it is not possible to store the requested data, the LDAP data access feature answers back to the request with the appropriate LDAP error code. For more information on storage, refer to *CUDB Data Storage Handling*, Reference [11].
- **LDAP View:** The LDAP Data Views function allows applications that access CUDB nodes through the LDAP protocol to use a custom data model, and perceive the data model stored in CUDB as not being fixed to a unique DIT and schema. LDAP users that have a view assigned to them by configuration see the data through a custom schema when building a DIT. Once a user with an assigned view establishes a new LDAP bind, all the following LDAP operations are performed against the LDAP view selected for this user. For more information on LDAP Data Views, refer to *CUDB LDAP Data Views*, Reference [3].

Note: The LDAP Data Views function can only be used if the Application Facilitator Value Package is available.





3 Operation and Maintenance

This section describes the operation and maintenance of the LDAP data access feature.

3.1 Configuration

This section describes the configuration mechanisms related to LDAP data access. In addition, schema configuration is considered as well, as described in *CUDB Node Schema Update*, Reference [5], due to its effects on LDAP data access flow.

3.1.1 Configuration Files

ACLs file must be available. The file includes the access control rules for all LDAP users. For details, refer to *CUDB Security and Privacy Management*, Reference [12].

3.1.2 LDAP Data Access Configuration

The different LDAP users and optionally the LDAP user groups must be configured for LDAP data access, including the authentication data.

CUDB provides the possibility to configure a node authentication default type for all LDAP users. It is also possible to configure the authentication type on a per LDAP user basis. Authentication data (hash or password) must be configured for the different LDAP users.

Transport Layer Security (TLS) configuration of Certification Authority (CA) must be configured in case TLS is used. This includes public certificate and private key. For more information, refer to *CUDB Node Configuration Data Model Description*, Reference [13].

In case secure LDAP is used on the proxy traffic between CUDB nodes, certificates for TLS are also needed in every CUDB node in the network. It is also necessary to configure the secure proxy option.

3.1.3 LDAP Data Access Parameters

The following parameters must be configured:

- DN for the main directory entry in the DIT of the LDAP FE.
- List of LDAP attributes to be managed as `searching optimization indexes` for distributed searches.



Note: These indexes must be defined before any entry with the indexed attributes is populated in CUDB.

LDAP User Parameters

Parameter values are received during the LDAP bind operation from the LDAP users (dynamic configuration). The behavior of CUDB regarding LDAP data access can be configured on a per LDAP user basis using the following parameter values:

- `isProvisioningUser`: two parameter values are supported (`TRUE` and `FALSE`). In potential symmetrical split situations, the write operations sent by LDAP users with this flag set to `TRUE` are not allowed by CUDB in the partition not hosting the PLDB Master Replica. This is to assure that provisioning operations are never lost even in symmetrical split situations when modifications on different replicas of the same data can occur. For more details on CUDB behavior in potential even split situations, refer to *CUDB High Availability*, Reference [8]. When Provisioning Assurance is running, the write operations sent by LDAP users with this flag set to `TRUE` and `isReProvisioningUser` flag set to `FALSE` are not allowed by CUDB. For more details on the Provisioning Assurance function, refer to *CUDB High Availability*, Reference [8].
- `isReProvisioningUser`: two parameter values are supported (`TRUE` and `FALSE`). Set to `TRUE` for LDAP users dedicated to re-provisioning operations related to the Provisioning Assurance function that can be configured in the CUDB system. During Provisioning Assurance event, the write operations sent by LDAP users with this flag set to `TRUE` are allowed by CUDB. When Provisioning Assurance event is not running, the write operations sent by LDAP users with this flag set to `TRUE` are not allowed. For more details on the Provisioning Assurance function, refer to *CUDB High Availability*, Reference [8].
- `cudbLdapViewId`: Due to the LDAP Data Views function, the `cudbLdapViewId` optional parameter can be used to assign a custom data view to an LDAP user. Refer to *CUDB LDAP Data Views*, Reference [3] for more information about data views.

Note: The LDAP Data Views function can only be used if the Application Facilitator Value Package is available.

- `readModeInPL`: three parameter values are supported (`MA` for Master Always, `MP` for Master Preferred, and `LP` for Local Preferred). Used to determine which PLDB replica is used for read LDAP requests when an access to PLDB is required.
- `readModeInDS`: three parameter values are supported (`MA` for Master Always, `MP` for Master Preferred, and `LP` for Local Preferred). Used to determine which DSG replica is used for read LDAP requests when an access to a DSG is required.



- `cudbUserGroup`: optional parameter that specifies to which user group the LDAP user belongs to, so Access Control Configuration Rules defined for the group apply to this user.
- `overloadRejectionWeight`: this parameter sets the LDAP traffic priority under PL or DS overload for this user. The lower value for this parameter means higher priority and a lower rejection rate.

CUDB supports the following value combinations for the `readModeInPL` and `readModeInDS` LDAP user attributes:

- `readModeInPL=LP` and `readModeInDS=MP` for "traffic" applications and application FEs (such as HSS and HLR).
- `readModeInPL=MA` and `readModeInDS=MA` for "provisioning" applications and application FEs (such as the Provisioning Gateway).
- `readModeInPL=LP` and `readModeInDS=LP` for "traffic" applications with local read.

Note: The `readModeInDS=LP` can be configured only if the Deployment Flexibility Value Package is available.

Any other combination of `readModeInPL` and `readModeInDS` is currently unsupported. The LDAP user settings for `readModeInPL` and `readModeInDS` configuration parameters can be overridden per single search request using the ReadMode LDAP control. For more information on ReadMode LDAP control, refer to *Section 6.4.1 ReadMode Control, CUDB LDAP Interwork Description*, Reference [6].

For more information on LDAP user configuration parameters, refer to *CUDB Node Configuration Data Model Description*, Reference [13].

For more information about the installation and the operation and maintenance of CUDB, refer to *CUDB System Administrator Guide*, Reference [1].

3.2 Provisioning

CUDB data provisioning can be performed by the LDAP data access service. CUDB supports the possibility to define LDAP users accessing to authoritative data so they always access the master copy of the data. This option can be used for data provisioning in CUDB. As CUDB offers a single point of access through LDAP clients, the LDAP users responsible for CUDB data provisioning can benefit from this data access functionality. In this way, provisioning of CUDB data is available in real-time.

For initial provisioning of CUDB data, provisioning can be performed using CUDB import tools loading a high amount of data directly into the CUDB local storage resources.

In both cases CUDB performs data validation. The added data must comply with the loaded schemas available in the CUDB system.

3.3 Fault Management

The CUDB system can raise the following alarms when monitoring the availability of the LDAP related local resources:

- Processing Redundancy Lost

For more information, refer to *LDAP Front End, Processing Redundancy Lost*, Reference [14].

- Processing Capacity Below Minimum

For more information, refer to *LDAP Front End, Processing Capacity Below Minimum*, Reference [15].

- Server Down

For more information, refer to *LDAP Front End, Server Down*, Reference [16].

- High Load

For more information, refer to *LDAP Front End, High Load in LDAP Processing Layer*, Reference [17].

For more information about fault management in CUDB, refer to *CUDB Node Fault Management Configuration Guide*, Reference [18].

3.4 Performance Management

The following events related to LDAP data access are counted by CUDB:

- Number of LDAP operations per second being carried out by each LDAP server.
- Number of LDAP requests handled by a CUDB node.
- Number of LDAP operations accessing the database clusters.

For more information about the counters, refer to *CUDB Counters List*, Reference [19].

3.5 Security

The following security mechanisms are supported by CUDB concerning the LDAP data access service:



- LDAP user authentication
- LDAP over SSL (LDAPS)
- Access Control List (ACL) mechanisms to set the LDAP user access privileges

CUDB supports simple LDAP user authentication following the LDAP standard. Authentication is supported by user credentials (user name and password) included in the LDAP bind request by the client (LDAP user).

CUDB supports Simple Authentication and Security Layer (SASL), which allows the LDAP user to send the hash of the password to the server during the authentication protecting password confidentiality in the network.

The CUDB system administrator can set LDAP user access privileges by the ACL file. This file includes the rules to be applied to the LDAP Users accessing CUDB. It is possible to apply different granularity to the rules: to define general or default ACL rules where CUDB allows a write access to all LDAP Users and it is also supported to define specific or restricted access per LDAP user to a certain branch and attributes.

Note: The `rootdn` user gets write access to any part of the DIT, its access can not be restricted by using ACLs.

Secure LDAP is LDAP over TLS/SSL. LDAP user authentication and traffic confidentiality can be protected by configuring TLS/SSL security layer for communications between client and server. CUDB supports client side authentication of TLS/SSL LDAP interface, and optionally server side authentication.

For more details, refer to *CUDB Security and Privacy Management*, Reference [12].

3.6 Logging

CUDB provides `OpenLdap syslog` as logging system. For more information on logging, refer to *CUDB Node Logging Events*, Reference [20].





4 Appendix A: Traffic Prioritization Configuration Guidelines

The following sections provide some examples to explain how the Traffic Prioritization feature works, and to help operators configure the feature according to their specific needs.

4.1 Traffic Prioritization Configuration

The Traffic Prioritization function is configured by assigning the traffic priority to LDAP users with the attribute `overloadRejectionWeight` in the `CudbLdapUser` data model (refer to *CUDB Node Configuration Data Model Description*).

4.2 Traffic Prioritization Examples

The following sections provide examples for some possible configuration cases.

Note: The below examples graphics serve as illustration only. Such graphics are not generated or provided by the CUDB system.

4.2.1 Two Users Configuration

Page 47 and Page 49 show different examples when two users are running LDAP traffic over a CUDB node.

Scenario 1

In this scenario, two different application FEs (LDAP users) are running LDAP traffic.

One of the application FEs has higher priority than the other one.

The incoming traffic load distribution is 50% higher-priority application FE, 50% lower-priority application FE.

The suggested Traffic Prioritization configuration is as follows:

- Higher-priority application FE/user with `overloadRejectionWeight` equal to 1.
- Lower-priority application FE/user with `overloadRejectionWeight` equal to 2.

In this scenario, the lower-priority application user still processes a bit of traffic of the total incoming traffic, during the overload situation.

Figure 18 shows LDAP traffic input in Scenario 1 in case of a sustained overload test, which increases the traffic amount from 80% of the Engineering Capacity (EC), through 100% of the EC until 300% of the EC. As seen in Figure 18, the total incoming traffic starts to be rejected from 150% of the EC, due to cluster overload.

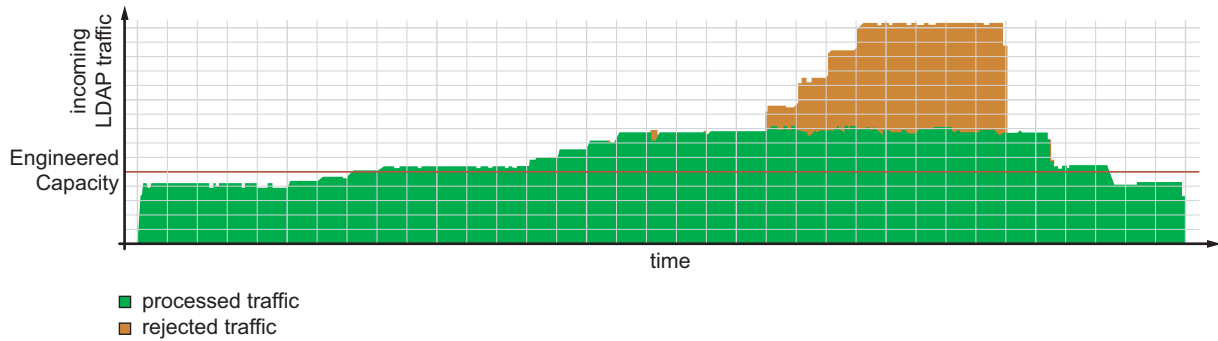


Figure 18 LDAP Traffic Input, Scenario 1

Figure 19 shows the individual processed operations for each user in Scenario 1. As long as there is no overload situation, all the traffic of both users is processed. When the traffic is increased above the overload level, CUDB processes more operations from the higher priority User 1, while CUDB processes less operations from the lower priority User 2.

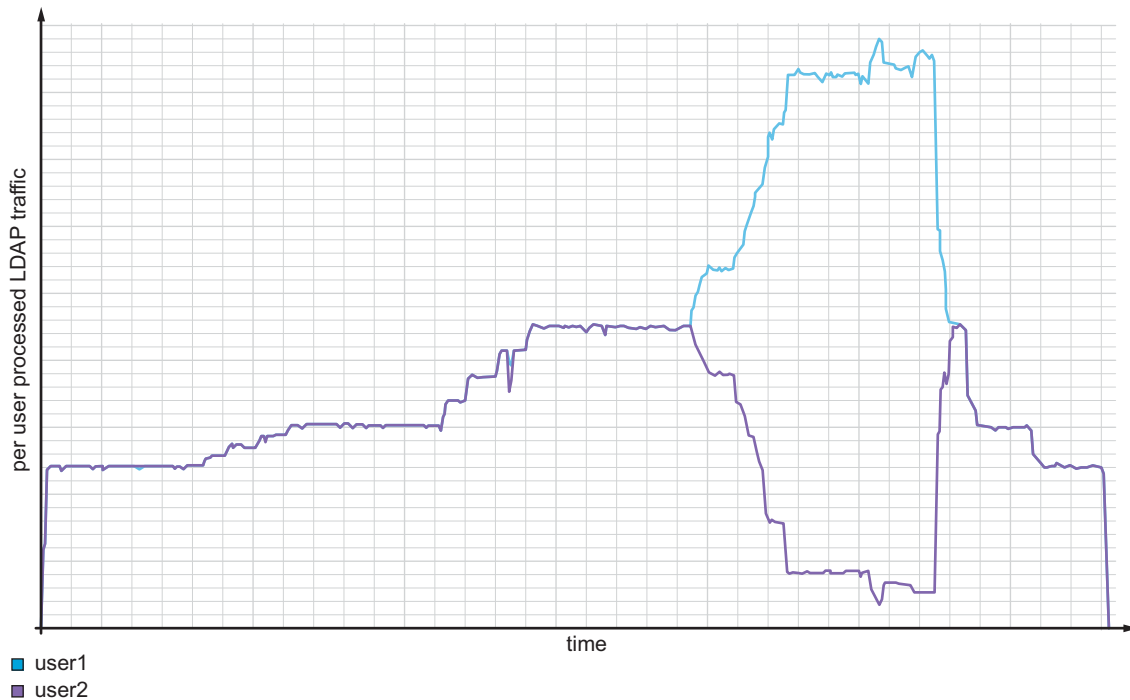


Figure 19 Processed LDAP Traffic for Each User, Scenario 1



The way to measure the priority is by checking the ratio between the processed LDAP traffic of each user and the total LDAP traffic of each user, as shown in Figure 20.

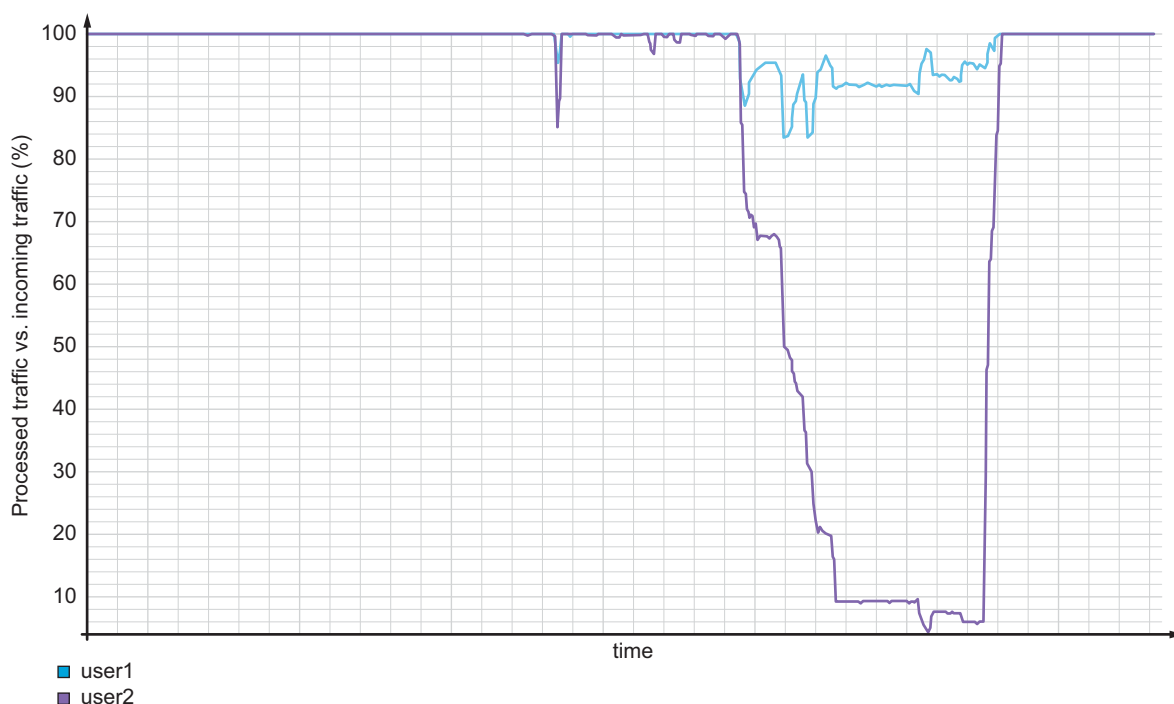


Figure 20 Processed LDAP Traffic Percentage for Each User, Scenario 1

As shown in Figure 20, the ratio of processed LDAP requests vs. total incoming LDAP requests for the higher-priority application FE is always as good (when not in overload situation) or better (during overload situation) than the ratio for the lower-priority application FE.

Scenario 2

In Scenario 2, two different application FEs (LDAP users) are running LDAP traffic.

One of the application FEs has higher priority than the other one.

The incoming traffic load distribution is 50% higher-priority application FE, 50% lower-priority application FE.

The suggested Traffic Prioritization configuration is as follows:

- Higher-priority application FE/user with `overloadRejectionWeight` equal to 1.
- Lower-priority application FE/user with `overloadRejectionWeight` equal to 5.

In this scenario, the lower-priority application user barely processes traffic of the total incoming traffic, during the overload situation.

Figure 21 shows LDAP traffic input in Scenario 2 in case of a sustained overload test, which increases the traffic amount from 80% of the EC through 100% of the EC until 300% of the EC. As seen in Figure 21, the total incoming traffic starts to be rejected from 150% of the EC, due to cluster overload.

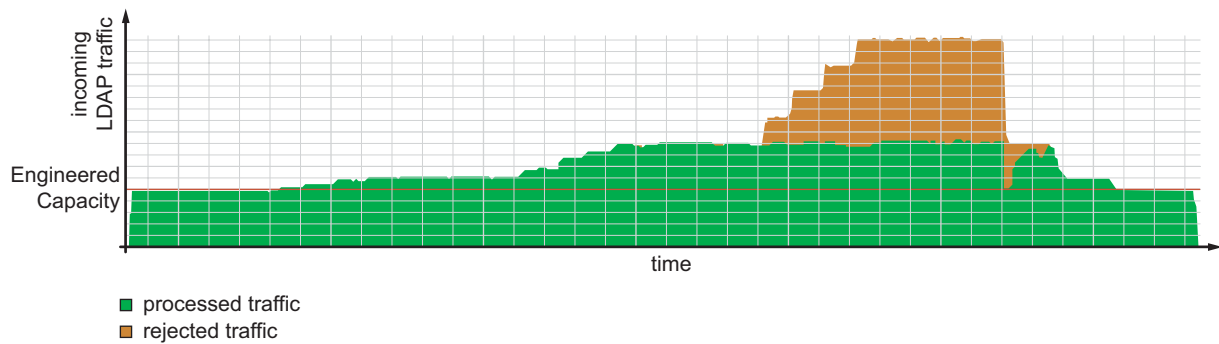


Figure 21 LDAP Traffic Input, Scenario 2

Figure 22 shows the individual processed operations for each user in Scenario 2. As long as there is no overload situation, all the traffic of both users is processed. When the traffic is increased above the overload level, CUDB processes more operations from the higher priority User 1, while CUDB processes less operations from the lower priority User 2.

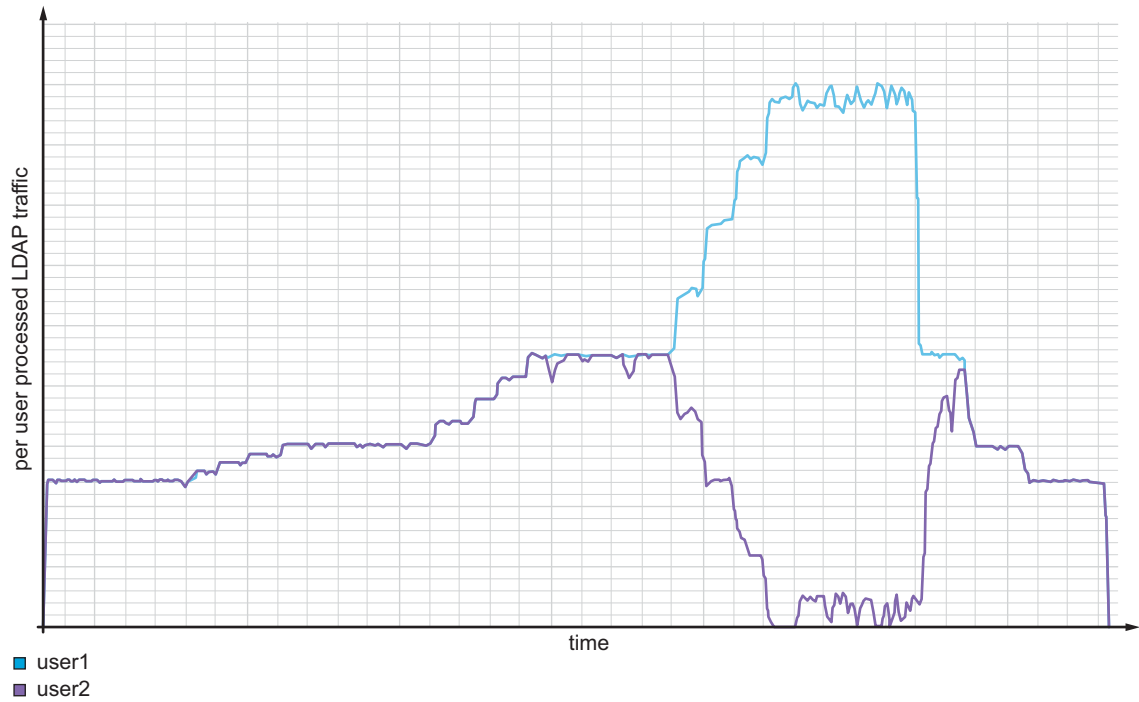


Figure 22 *Processed LDAP Traffic for Each User, Scenario 2*

Figure 23 shows the processed LDAP traffic percentage for each user in Scenario 2.

User 2 (with lower priority) rejects more traffic to allow User 1 (with higher priority) to process almost all its traffic.

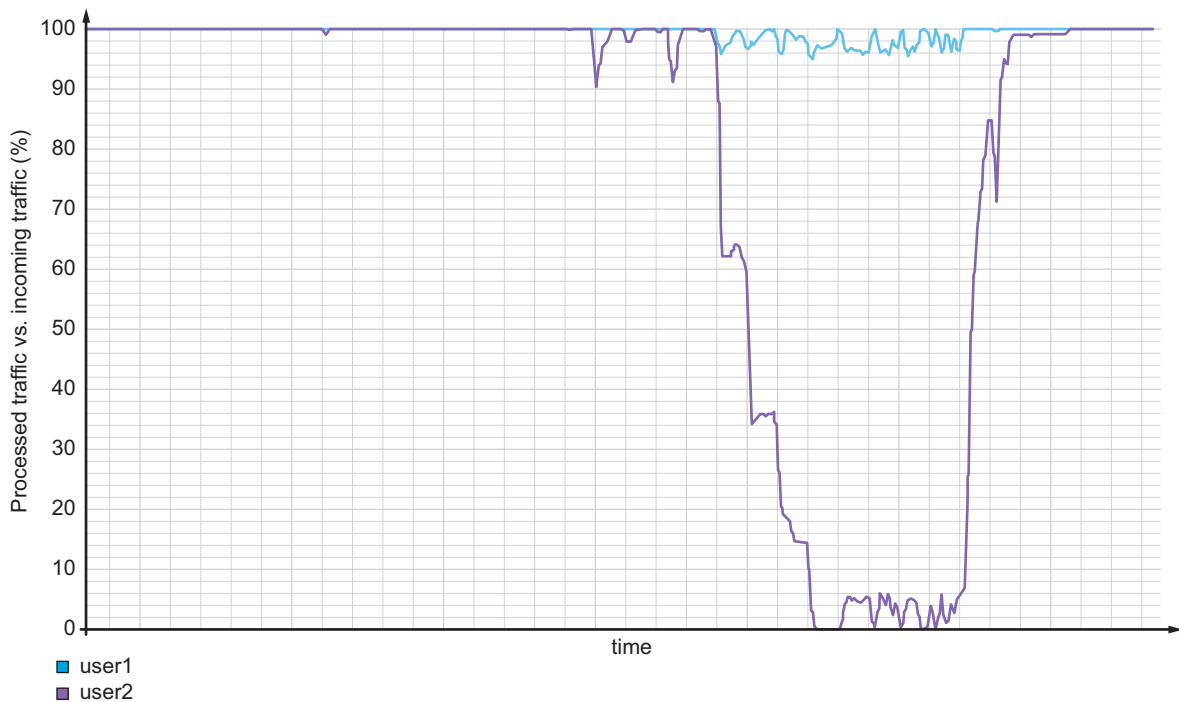


Figure 23 Processed LDAP Traffic Percentage for Each User, Scenario 2

In Figure 19 and Figure 20, the processed traffic ratio for User 2 is higher than the processed traffic ratio for User 2 in Figure 22 and Figure 23.

4.2.1.1 Unbalanced Traffic Examples

In case the different users send different percentage of the traffic (unbalanced LDAP traffic), as expected, high priority LDAP users keep a higher ratio of processed LDAP traffic, while lower priority LDAP users have a lower percentage of processed LDAP traffic.

Scenario 1

In this scenario, two different application FEs (LDAP users) are running LDAP traffic.

One of the application FEs has higher priority than the other one.

The incoming traffic load distribution is 10% higher-priority application FE, 90% lower-priority application FE.

The suggested Traffic Prioritization configuration is:

- Higher-priority application FE/user with `overloadRejectionWeight` equal to 1.



- Lower-priority application FE/user with `overloadRejectionWeight` equal to 3.

Figure 24 shows LDAP traffic input in Scenario 1 in case of a sustained overload test, which increases the traffic amount from 80% of the EC, through 100% of the EC until 300% of the EC. As seen in Figure 24, the total incoming traffic starts to be rejected from the level of 150% of the EC, due to cluster overload. In this case, the processed traffic is less smooth compared to earlier examples.

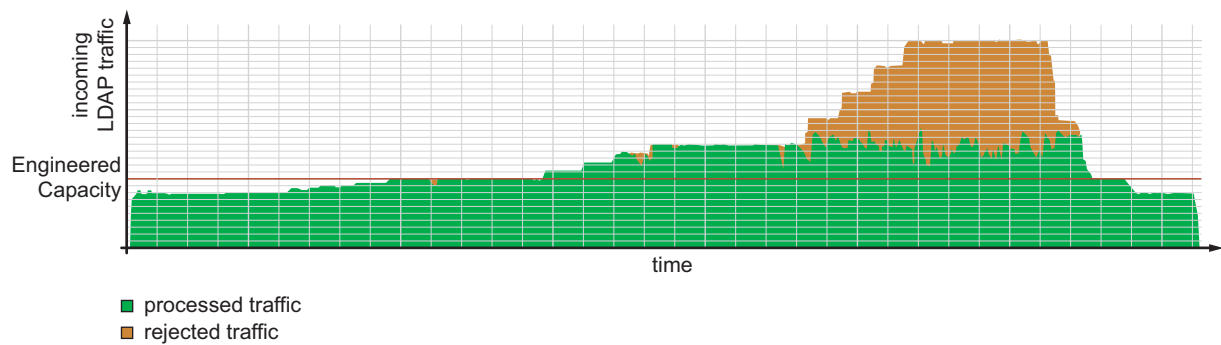


Figure 24 LDAP Traffic Input, Scenario 1

Figure 25 shows the total amount of successful traffic for each user in Scenario 1. As shown in Figure 25, the amount of the processed operations for each user corresponds with Scenario 1, as `User 1` is sending just 10% of the entire traffic, while `User 2` sends the other 90%.

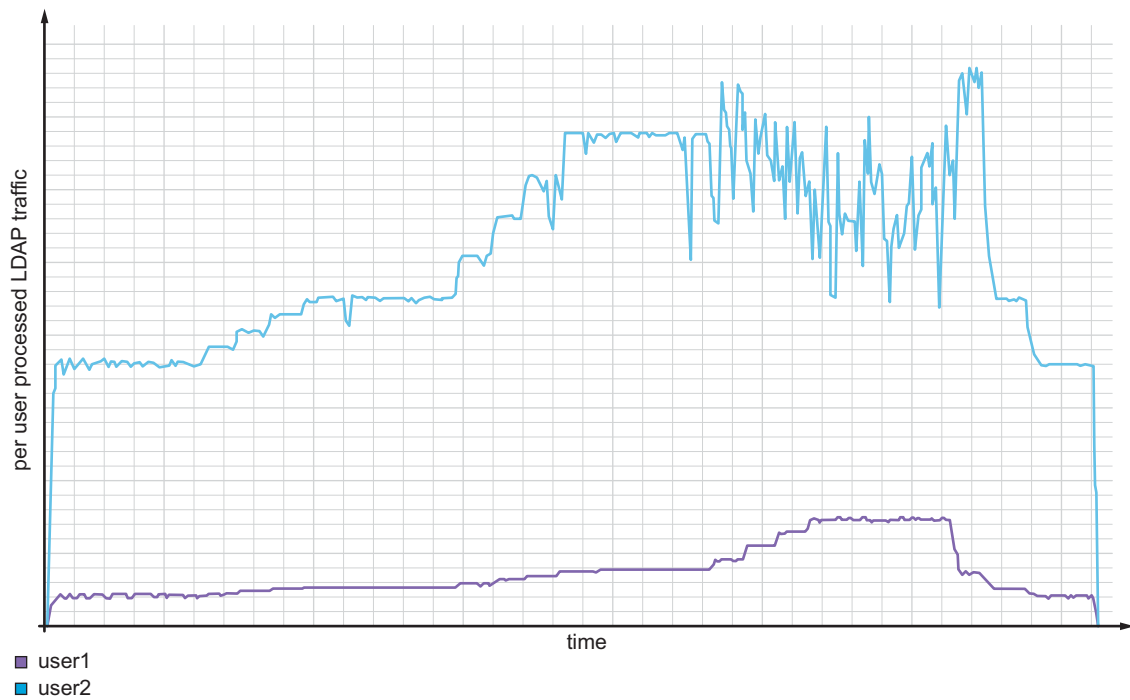


Figure 25 Processed LDAP Traffic for Each User, Scenario 1

However, compared to the total amount of LDAP traffic percentages per user, the success rate of User 1 is higher than the rate of User 2, as shown in Figure 26.

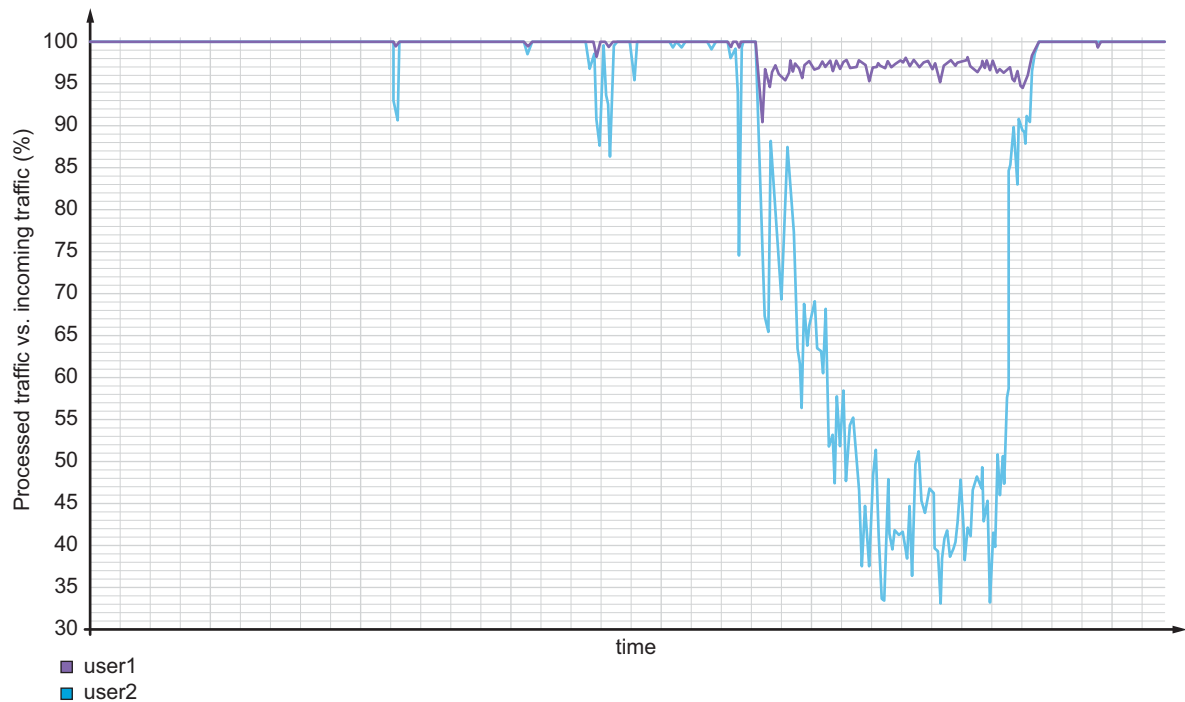


Figure 26 Processed LDAP Traffic Percentage for Each User, Scenario 1

Scenario 2

In this scenario, two different application FEs (LDAP users) are running LDAP traffic.

One of the application FEs has higher priority than the other one.

The incoming traffic load distribution is 10% higher-priority application FE, 90% lower-priority application FE.

The suggested Traffic Prioritization configuration is as follows:

- Higher-priority application FE/user with `overloadRejectionWeight` equal to 4.
- Lower-priority application FE/user with `overloadRejectionWeight` equal to 4.

Figure 27 shows LDAP traffic input in Scenario 2 in case of a sustained overload test, which increases the traffic amount from 80% of the EC, through 100% of the EC until 300% of the EC. As seen in Figure 27, the total incoming traffic starts to be rejected from the level of 150% of EC, due to cluster overload. However, in this case, the rejection rate is smoother than in Page 52, as both users now have the same priority.

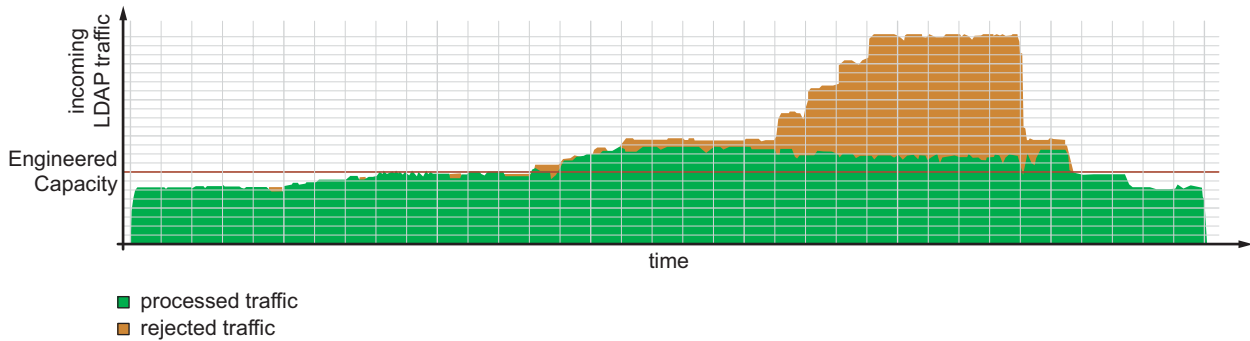


Figure 27 LDAP Traffic Input, Scenario 2

Figure 28 shows the processed LDAP traffic for each user in Scenario 2. As shown in Figure 28, the amount of the processed operation for each user corresponds with Scenario 2, as User 1 is sending just 10% of the total amount of traffic, while User 2 sends 90%.

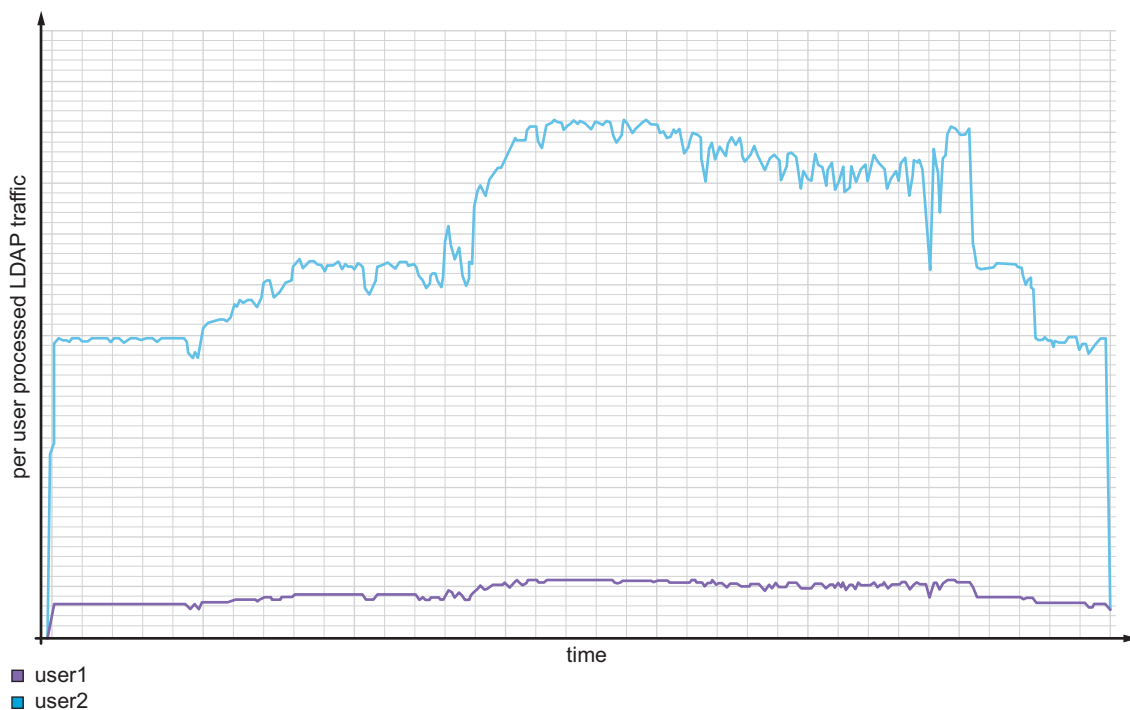


Figure 28 Processed LDAP Traffic for Each User, Scenario 2

Figure 29 shows that the success rate is the same for both users when compared to the LDAP traffic percentage for each user in Scenario 2.

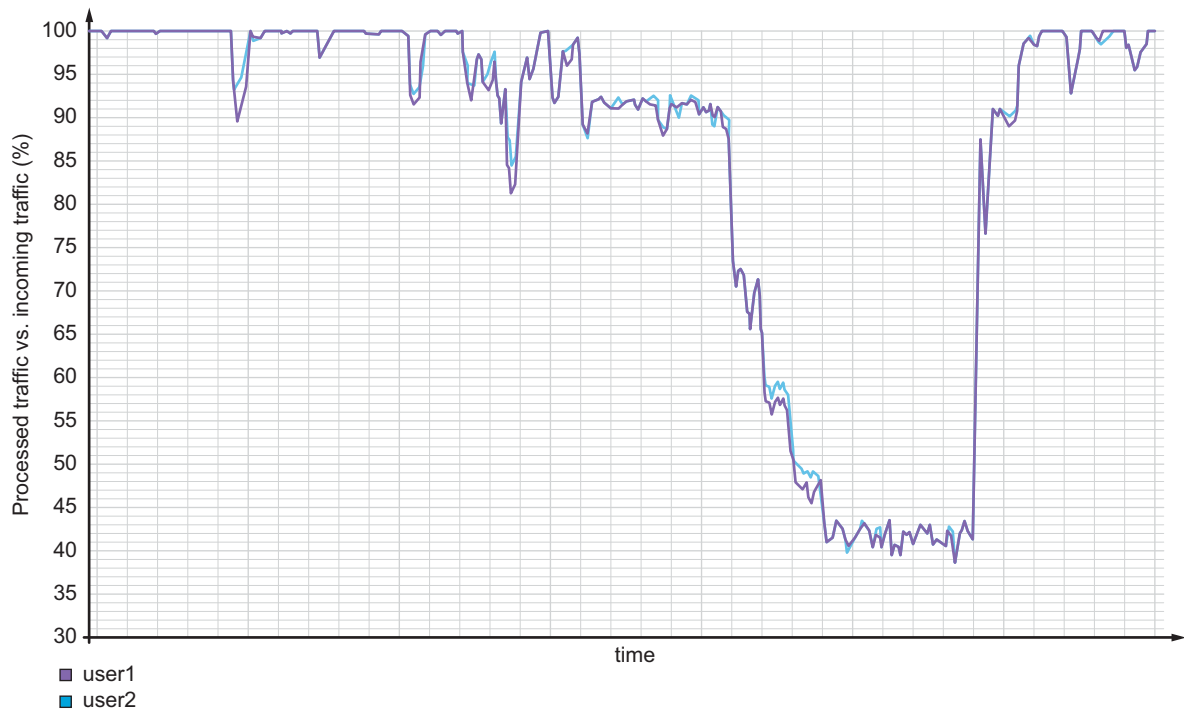


Figure 29 Processed LDAP Traffic Percentage for Each User, Scenario 2

4.2.2 Three Users Configuration

This section shows examples of three-user configurations.

Scenario 1

In this scenario, three different application FEs (LDAP users) are running LDAP traffic with different priorities.

The incoming traffic load distribution is equally distributed among the three LDAP users.

The suggested Traffic Prioritization configuration is as follows:

- Higher-priority application FE/user with `overloadRejectionWeight` equal to 1.
- Medium-priority application FE/user with `overloadRejectionWeight` equal to 2.
- Lower-priority application FE/user with `overloadRejectionWeight` equal to 3.

Figure 30 shows the LDAP traffic input for the CUDB node in Scenario 1. As in the earlier sections, the example shows a sustained overload situation.

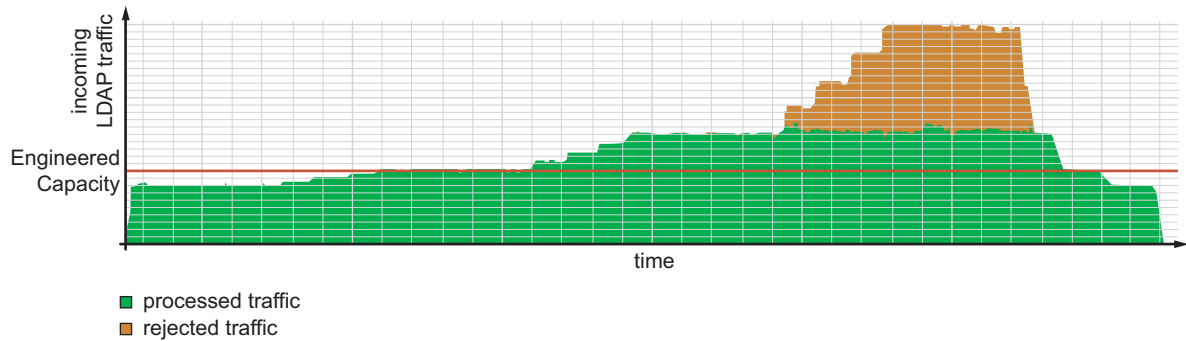


Figure 30 LDAP Traffic Input, Scenario 1

Figure 31 shows the amount of processed LDAP traffic for each user in Scenario 1. As each user is sending the same amount of LDAP traffic, they are processed equally until the overload situation. However, during the overload situation, the amount of processed traffic corresponds to Scenario 1: most of the processed traffic belongs to the highest priority user (User 1), User 2 has less amount of processed traffic (as it has medium priority), while User 3 does not even have any processed traffic for some time, since it has the lowest priority.

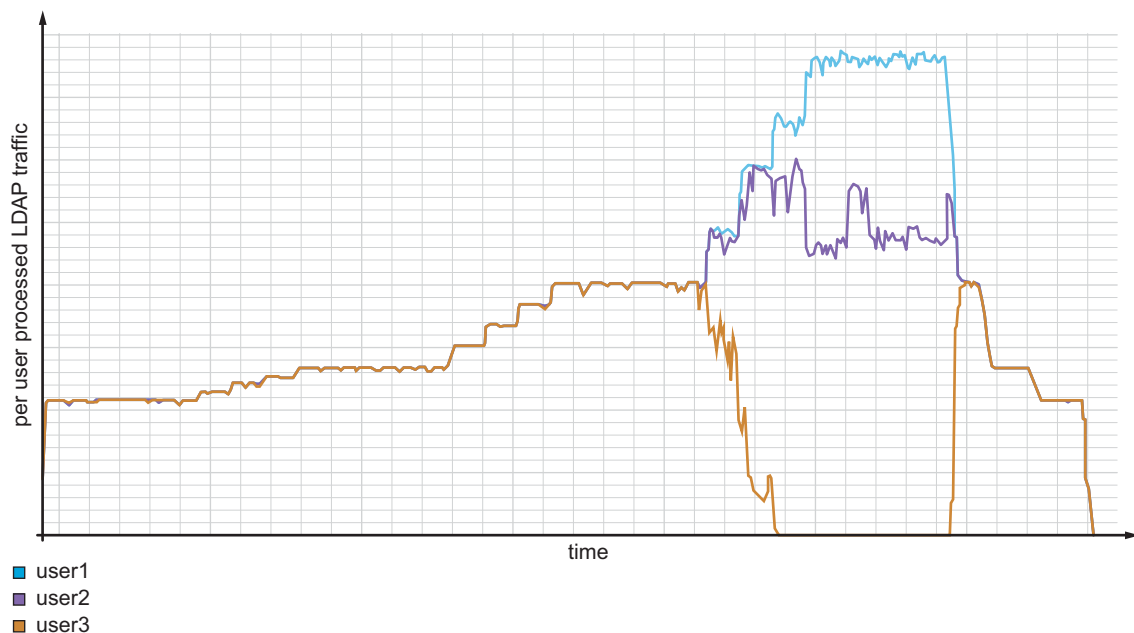


Figure 31 Processed LDAP Traffic for Each User, Scenario 1

Figure 32 shows the percentage of processed LDAP traffic for each user in Scenario 1. The highest percentage of processed traffic belongs to the highest priority user (User 1), User 2 has a lower percentage of processed traffic (as it has medium priority), while User 3 does not even have any processed traffic for some time, since it has the lowest priority.

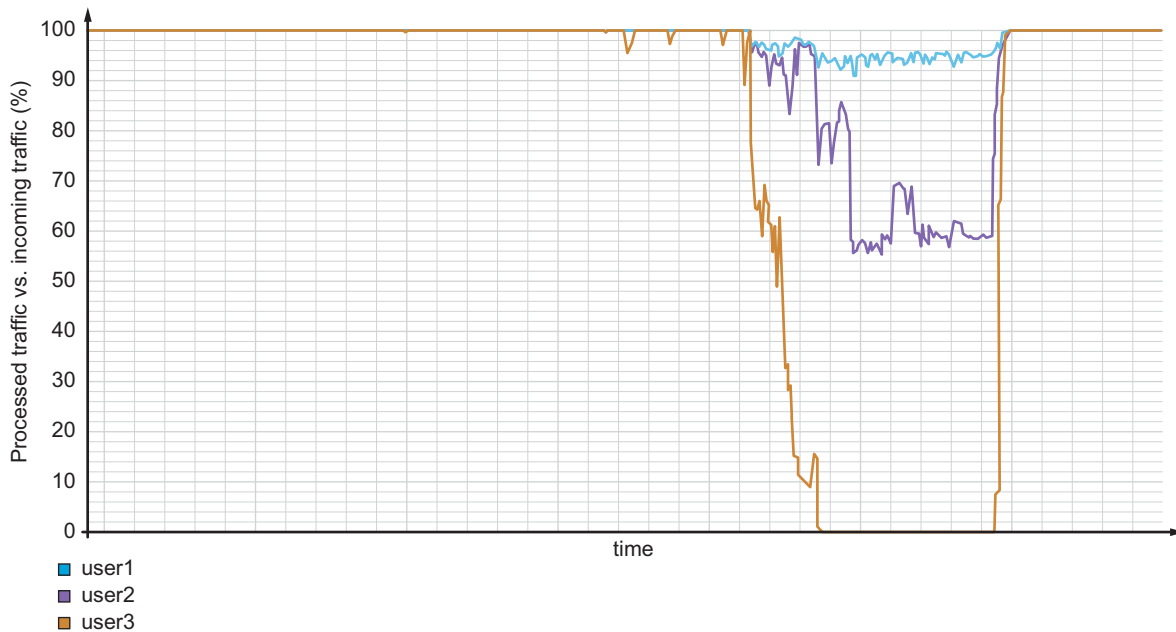


Figure 32 Processed LDAP Traffic Percentage for Each User, Scenario 1

Scenario 2

In this scenario, three different application FEs (LDAP users) are running LDAP traffic with different priorities. The incoming traffic load distribution is equally distributed among the three LDAP users.:

The suggested Traffic Prioritization configuration is as follows:

- Higher-priority application FE/user with `overloadRejectionWeight` equal to 1.
- Medium-priority application FE/user with `overloadRejectionWeight` equal to 3.
- Lower-priority application FE/user with `overloadRejectionWeight` equal to 5.

Figure 33 shows the LDAP traffic input in Scenario 2. As in the earlier sections, the example shows a sustained overload situation.

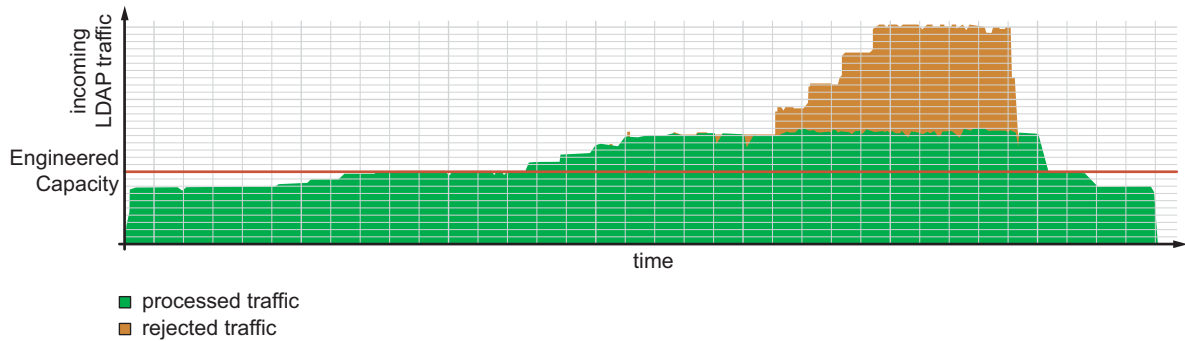


Figure 33 LDAP Traffic Input, Scenario 2

Figure 34 shows the processed LDAP traffic for each user in Scenario 2. As each user is sending the same amount of traffic, they are processed equally until the overload situation. During the overload situation, the processed traffic amount corresponds to Scenario 2: most of the processed traffic belongs to the highest priority user (User 1), User 2 has less amount of processed traffic (as it has medium priority), while User 3 does not even have any processed traffic for some time, since it has the lowest priority. As User 3 has a lower priority now than in Page 57, the amount of its processed traffic is also worse than in Page 57.

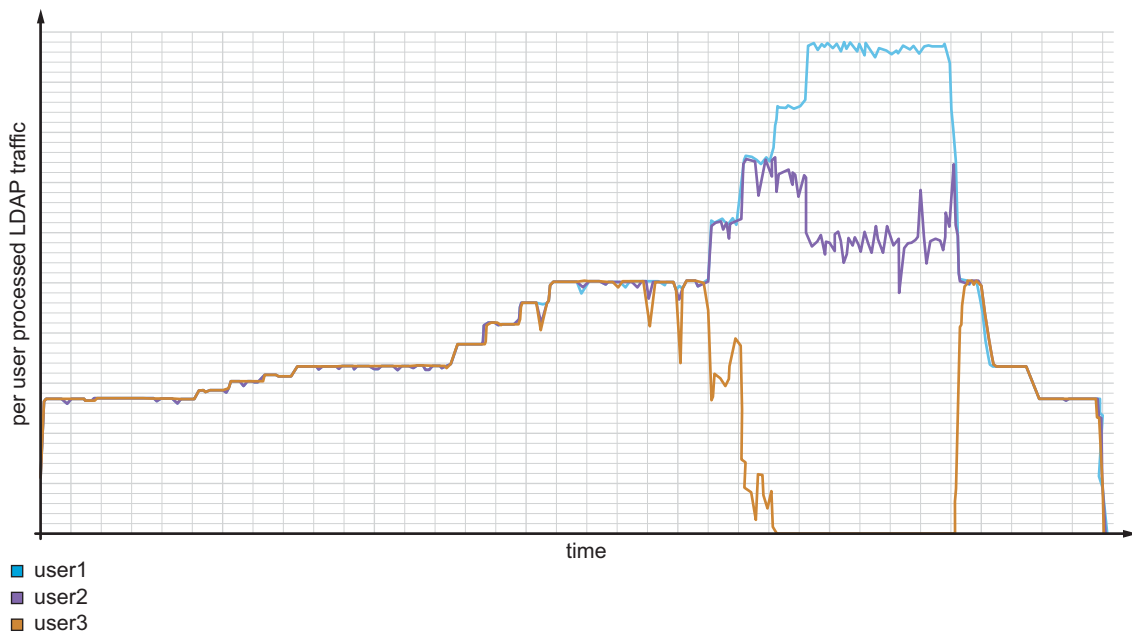


Figure 34 Processed LDAP Traffic for Each User, Scenario 2

Figure 35 shows the percentage of processed LDAP traffic for each user in Scenario 2. As User 3 has lower priority now than in Page 57, the percentage of processed traffic for User 3 is also lower in this scenario.

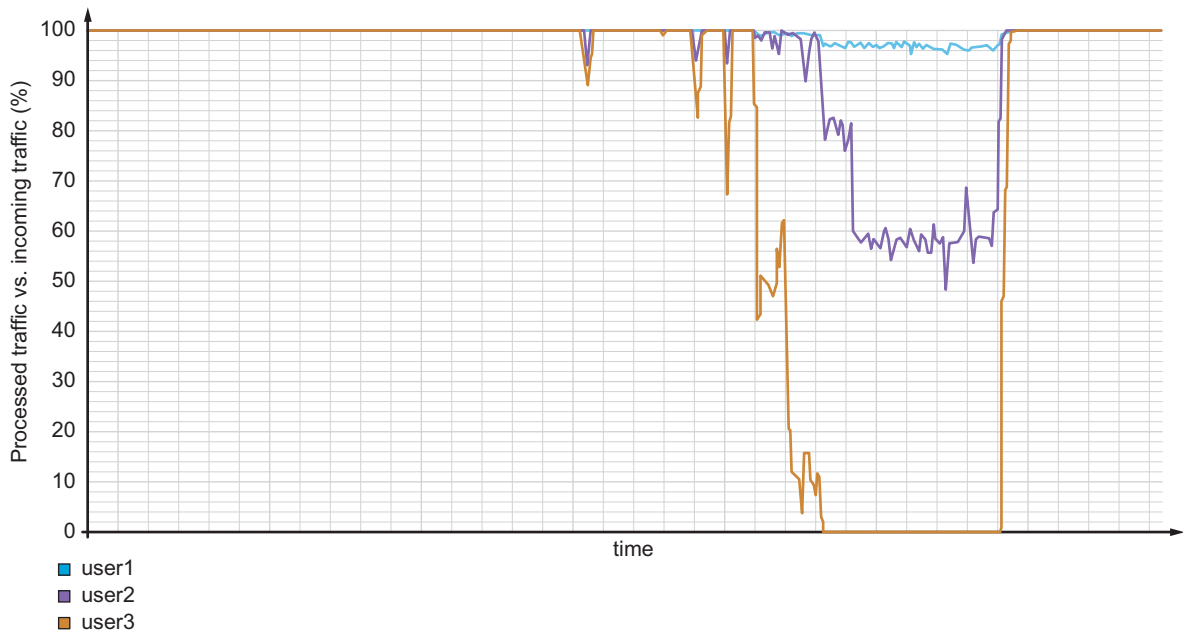


Figure 35 Processed LDAP Traffic Percentage for Each User, Scenario 2

In case of Scenario 2, the incoming LDAP traffic of User 3 under overload situation is starting to reject earlier than in Scenario 1.

4.2.3 Online Priority Change

The priority of a CUDB LDAP User can be changed by modifying `overloadRejectionWeight` in the data model.

Figure 36 and Figure 37 show examples when two users are running LDAP traffic over a CUDB node with `overloadRejectionWeight` set to 1. User 1 changes from 1 to 5 and then, back to 1 again. Later, User 2 goes through exactly the same procedure.

Figure 36 shows the LDAP traffic input for the above example during a sustained overload situation.

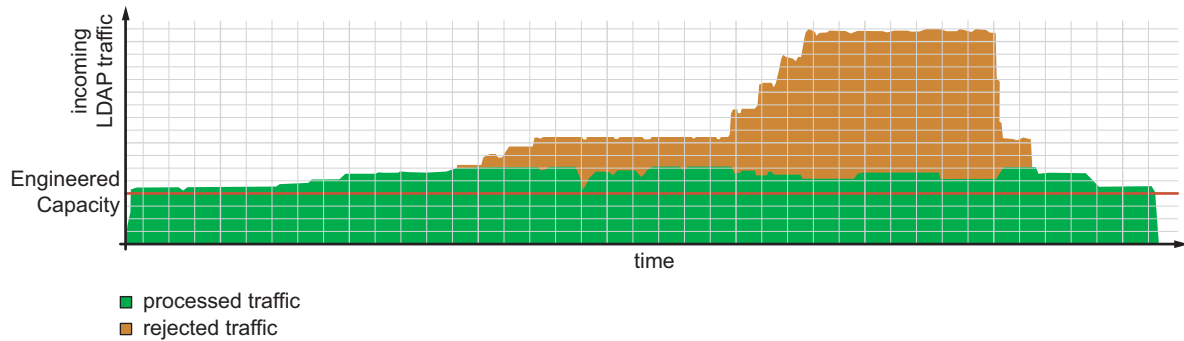


Figure 36 LDAP Traffic Input, Priority Change Example

Figure 37 shows the percentage of processed LDAP traffic for each user. As the users are sending the same amount of traffic until an overload situation, the processed amount of traffic is the same for both users. However, at 150% of the EC level, when the traffic priority of `User 1` is changed from 1 to 5, the processed traffic is falling down drastically for this user, allowing to process more traffic for the other user. If the priority is changed back from 5 to 1, the processed traffic is normalized, and reverts to the same way as before.

At 300% of the EC level, when the traffic priority of `User 2` is changed from 1 to 5, the amount of processed traffic is decreasing drastically for `User 2`, allowing to process more traffic for the other user. If the priority is changed back from 5 to 1, the processed traffic is normalized, and reverts to the same way as before.

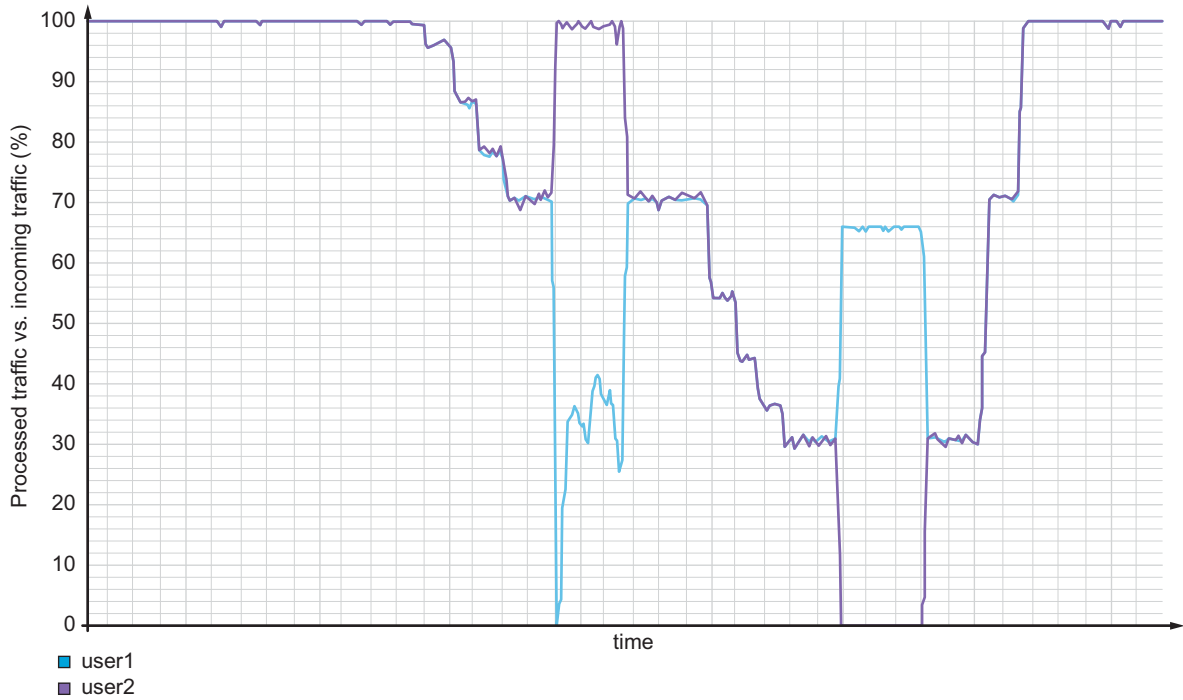


Figure 37 Processed LDAP Traffic Percentage for Each User, Priority Change Example

As seen from the above examples, the priority change can be executed during an overload situation without traffic loss, and without the need to restart any components.

4.3 Performance Management

There are no specific counters developed in CUDB for measuring the results of Traffic Prioritization. However, if the operator is interested in the statistic information of the Traffic Prioritization, it is possible to make correspondence between the `overloadRejectionWeight` and the `countersGroups` of the CUDB LDAP users required (refer to *CUDB Counters List*, Reference [19]). Following is a sample for a possible correspondence-configuration:

```
overloadRejectionWeight countersGroup = 1
eight = 1
overloadRejectionWeight countersGroup = 2
eight = 2
overloadRejectionWeight countersGroup = 3
eight = 3
overloadRejectionWeight countersGroup = 4
eight = 4
overloadRejectionWeight countersGroup = 0
eight = 5
```



Note: The `countersGroup = 0` is the default `counterGroup`. Any traffic which is not assigned to a specific `countersGroup` will store its traffic in this `countersGroup`.



Glossary

For the terms, definitions, acronyms and abbreviations used in this document, refer to *CUDB Glossary of Terms and Acronyms*, Reference [21].





Reference List

CUDB Documents

- [1] *CUDB System Administrator Guide*
- [2] *CUDB LDAP Schema Management*
- [3] *CUDB LDAP Data Views*
- [4] *CUDB Application Integration Guide*
- [5] *CUDB Node Schema Update*
- [6] *CUDB LDAP Interwork Description*
- [7] *CUDB Multiple Geographical Areas*
- [8] *CUDB High Availability*
- [9] *CUDB Import and Export Procedures*
- [10] *CUDB Data Distribution*
- [11] *CUDB Data Storage Handling*
- [12] *CUDB Security and Privacy Management*
- [13] *CUDB Node Configuration Data Model Description*
- [14] *LDAP Front End, Processing Redundancy Lost*
- [15] *LDAP Front End, Processing Capacity Below Minimum*
- [16] *LDAP Front End, Server Down*
- [17] *LDAP Front End, High Load in LDAP Processing Layer*
- [18] *CUDB Node Fault Management Configuration Guide*
- [19] *CUDB Counters List*
- [20] *CUDB Node Logging Events*
- [21] *CUDB Glossary of Terms and Acronyms*

Other Documents and Online References

- [22] *LDAP: Technical Specification Road Map. IETF RFC 4510*
<http://www.rfc-editor.org/rfc/rfc4510.txt>



- [23] *LDAP: The Protocol. IETF RFC 4511* <http://www.rfc-editor.org/rfc/rfc4511.txt>
- [24] *LDAP: Directory Information Models. IETF RFC 4512*
<http://www.rfc-editor.org/rfc/rfc4512.txt>
- [25] *LDAP: Authentication Methods and Security Mechanisms. IETF RFC 4513* <http://www.rfc-editor.org/rfc/rfc4513.txt>
- [26] *LDAP: String Representation of Distinguished Names. IETF RFC 4514*
<http://www.rfc-editor.org/rfc/rfc4514.txt>
- [27] *LDAP: String Representation of Search Filters. IETF RFC 4515*
<http://www.rfc-editor.org/rfc/rfc4515.txt>