ERICSSON

Toolbox Description

TECHNICAL PRODUCT DESCRIPTION

12/1551-AXB 250 12-V3 Uen EY

Contents

1 Introduction

The EPG Toolbox is a collection of tools used to simplify maintenance and troubleshooting of the EPG. The Toolbox is included in the EPG delivery.

This document contains a description of the tools that are supported in the current software release.

Do!

Use only those Toolbox commands and parameters that are listed in this document. Any other Toolbox commands and parameters are for internal Ericsson use. Incorrect use of internal Ericsson Toolbox commands or parameters may cause an overload in an active EPG or other unwanted disturbances.

1.1 Scope

This document describes the Toolbox as well as the tools it contains and their syntax, and provides usage examples.

The following tools are described in this document:

— Performance Data Collection (PDC) tools

— Support Package

— EPG Automatic Log Package

— System storage cleanup

— Charging Data Record (CDR) decoding

— EPG Software Version

— UE Trace to PCAP Converter

— UE Trace Parser

— Loggd Joiner

— Flatten XML

1.2 Target Groups

This document is an introduction to the Toolbox for network operators, network and service planners, and system engineers and administrators.

1.3 Prerequisites

The person using the Toolbox should have solid knowledge of and training in the following areas:

— Working in UNIX™ and similar systems

— Packet-switching communication

— Operation of the EPG

2 General Description

The Toolbox is automatically installed during the deployment of EPG or software update procedure; however, tools can be started only after the EPG application has started. The local tools are ready for use on the EPG and are used from a Linux shell on the node. All Toolbox commands assume that the user first starts a shell with the command **`start shell`**.

2.1 Remote Toolbox

Two remote Toolbox tarballs contain the tools to be executed on platforms running Linux® or Microsoft Windows® with Cygwin package installed. The names and locations of the tarballs are as follows:

EPG on the Virtual Routing Engine (vRE), in directory `/opt/services/epg/bin/ttx/`:
`remote-epg-ssr-toolbox-linux_iX86_<ver>.tgz`
`remote-epg-ssr-toolbox-cygwin_<ver>.tgz`

To install the remote tools, execute the following steps:

1. Connect from the remote host to the EPG.
2. Fetch the appropriate Toolbox tarball (Linux or Cygwin, depending on the platform where installation is to be performed), for example, through an established Secure File Transfer Protocol (SFTP) connection.

3. Unpack the file in the desired location. The Toolbox is deployed under the subdirectory named as follows:

   Cygwin Toolbox:
   `epg-ssr-toolbox-cygwin`
   Linux Toolbox:
   `epg-ssr-toolbox-linux_iX86`

2.2 Syntax Options

In the syntax of the commands, the following notation is used:

`[ ]`	Operands and variables in brackets are optional.
`...`	The variable or structure preceding the ellipsis can be repeated once or several times.
`|`	The operands or variables separated by the pipe are mutually exclusive. Only one of the alternatives can be entered.

3 Tools

This section describes the tools provided by the Toolbox and their use.

The following tools are part of the Toolbox:

— Local tools - tools that run on the EPG

- PDC tools
- Support Package
- EPG Automatic Log Package
- System Storage Cleanup
- EPG Healthcheck KPIs
- EPG Delta Statistics
- EPG Software Version
- Loggd Joiner
- PISC Security Log
- Flatten XML

— Remote tools - tools that execute on remote workstations running Linux or MS Windows with installed Cygwin package. Remote Linux Toolbox was tested on a Red Hat 5.5, 32-bit machine. Remote Cygwin Toolbox was tested on a Cygwin 1.7, 32-bit machine.

  - CDR Decoder
  - EBM Data Parser (available only on Linux)
  - UE Trace to PCAP Converter
  - UE Trace Parser
  - ACR Decoder

3.1 Performance Data Collection

Performance Data Collection (PDC) is a set of tools used to collect performance management information from EPG, and package all relevant data into PDC packages.

During the Toolbox installation on EPG, a PDC instance, with specific options, is scheduled for periodic execution. This PDC instance is referred to as the default PDC run. Users can also reconfigure PDC and manually control the execution of PDC tools.

PDC uses an internal Telnet session, which is not using the Telnet service over the Ethernet management port. Without this internal Telnet session, PDC cannot collect data. To ensure that the internal Telnet session executes properly, it is recommended to keep the default SSH and Telnet configuration. If the default configuration needs to be adjusted, maximum 29 SSH sessions can be allowed, leaving the remaining three sessions for the internal Telnet sessions, refer to Using the CLI.

Information provided by PDC is not intended to replace performance monitoring statistics gathered from the EPG as described in Counters and Gauges for the GGSN and PGW and Counters and Gauges for the SGW.

This document does not provide interpretation of the logs handled by PDC, which means that the user should be familiar with the content of various system logs and command printouts.

Note: It is recommended to have at least 1 GB of free space on the partition where the PDC runs. This size is only a recommendation. The actual amount of free space required depends on the node configuration.

PDC collection must be stopped before each software upgrade and update. PDC collection restarts automatically after the upgrade and update procedure is finished. For more information, refer to Software Upgrade for Virtual EPG.

A portion of PDC configuration is persistent and survives a node reload. This includes interval setting, administrator username, the list of APNs, and a definition of a custom run. Remaining configuration is reset to default values at reload.

Note: Some PDC configuration cannot be changed while statistics collection is ongoing. Use the command **`stop_sched`** to stop the collection temporarily, then modify necessary configuration, and finally restart collection with the start command. For more details, see Section 3.1.4 on page 8.

3.1.1 PDC Monitor User Account

The default PDC run communicates with the Ericsson Core CLI (ECCLI) using a PDC Monitor user account. Configure the PDC Monitor user account when the EPG is first installed. For the PDC to work, the PDC Monitor user accounts must be configured according to the instructions, refer to Security Management.

If the user account is not manually configured, the PDC automatically selects a user account, otherwise the PDC cannot collect the required statistics. The automatically selected user account is active until a manual configuration is done.

Select the PDC Monitor user account for PDC use during the Toolbox installation by the **`epg_pdc_control user=<username>`** command. The configured PDC user account must be assigned to required NACM groups. PDC user account verification is triggered by the **`epg_pdc_control start`** command or if the PDC daemon process is restarted, for example, after reboot. If the current PDC user account does not meet the PDC user account requirements, PDC selects a new user account as follows:

— If a PDC user is already configured, this user is used to run the **`show running-config contexts context local`** command. If the PDC user cannot run the command, the first user in the `ericsson.xml` file is used to run the **`show running-config contexts context local`** command. If that user cannot either execute the command PDC fails to select a PDC user and thus has to be set manually.

— If a PDC Monitor user account with username `pdcuser` is configured with the required NACM groups, this user is selected as PDC user. If there is no PDC Monitor user account with username `pdcuser` or if the required NACM groups of this user are not correct, the first user that is assigned to the required NACM groups is selected as PDC user.

— If PDC fails to select a new PDC user, it falls back to the previous PDC user.

The same mechanism is used to verify the PDC user when it is configured manually for the first time.

3.1.2 Default PDC Run

The PDC detects the node type; SGW, PGW, or combined based on the running configuration. The default value is combined.

The node role can also be configured manually by using the following command:

```
epg_pdc_control nodetype=type
```

Type(s)	Description
pgw	PGW
sgw	SGW
combined	Execute both the sgw and the pgw commands.
pgw,sgw	same as combined
sgw,pgw	same as combined

The node type is verified and updated at the following events:

— PDC start triggered by the `epg_pdc_control start` command

— The first PDC run, for example, after start or a reboot

— The first daily PDC run after midnight

Note: The manually set node type is automatically changed if it does not match the detected node type.

The default PDC run collects statistics generated by the following EPG ECCLI commands, depending on the node type (sgw, pgw or combined), see Section 3.1.2.1 on page 6, Section 3.1.2.2 on page 7, and Section 3.1.2.3 on page 8.

By default, statistics are collected every 15 minutes. For information on configuring a collection interval, see Section 3.1.4 on page 8.

Caution!

To preserve PM log integrity, the recommended sampling interval is 15 minutes or more.

Collected statistics (raw PDC data) are parsed, and the information is rearranged into different output files (parsed PDC data). Both raw PDC data and parsed results are stored in the `/var/log/services/epg/pdc/work/tmp/` directory.

3.1.2.1 nodetype=sgw

— EPG per-board statistics – OLP Status

```
epg node olp status value detail
```

— Global SGW statistics

```
epg sgw statistics
```

— ARP statistics

```
epg sgw statistics-arp
```

— QCI statistics

```
epg sgw statistics-qci qci all interface all detail-level detail
```

— CPU and memory utilization statistics for active Node Management Board (NMB); obtained from the platform:

`show chassis`
`show_chassis_routing-engine` (a shell script that collects relevant CPU and memory utilization)

3.1.2.2 nodetype=pgw

— EPG per-board statistics – OLP Status

```
epg node olp status value detail
```

— Global PGW statistics

```
epg pgw statistics of brief
```

— Charging statistics

```
epg pgw statistics of charging
```

— Per-PPB dropped packets statistics

```
epg pgw statistics of ppb
```

— DPI flow statistics

```
epg pgw internal-debug pisc-flows level detail
```

— DPI counters

```
epg pgw internal-debug inspection cmd-string "show performance-counters"
```

— QCI statistics

```
epg pgw statistics of qci
```

— High availability statistics

```
epg pgw statistics of high-availability
```

— PPB-Drops statistics

```
epg node user-plane show-drops
```

— Per-APN statistics

  For more information, see Section 3.1.6.1 on page 15.

— CPU and memory utilization statistics for active Node Management Board (NMB); obtained from the platform:

  show chassis
  show_chassis_routing-engine (a shell script that collects relevant CPU and memory utilization)

3.1.2.3 nodetype=combined

Collects statistics for the SGW and for the PGW, as described in Section 3.1.2.1 on page 6 and Section 3.1.2.2 on page 7.

3.1.3 PDC Package

To get a package of performance data for each month, every first day of the month, at 01:00 AM, parsed result files of the previous month are generated and packaged into a tarball referred to as a PDC package. PDC package names have the following format: <Month>_<hostname>_<yyyy>-<mm>-monthly_ggsn_pdc.tar.gz, where <Month> is a three-letter abbreviation of the month for which the PDC data is collected (for example, Aug for August), <hostname> is the EPG node name, and <yyyy>-<mm> is the year and month when PDC package is generated.

To list all available PDC packages, use the following ECCLI command:

```
ls /md/services/epg/pdc/archive/
```

PDC packages should be fetched from an external management server every month. Only six monthly packages are kept on the EPG.

3.1.4 Managing PDC Collection

Under normal conditions, there is no need to run PDC manually, as the default PDC run collects essential statistics. This section should be used only if necessary, and under the supervision of Ericsson personnel.

Stop!

The execution of some EPG commands can lead to an increase in Central Processing Unit (CPU) load, especially if done periodically. Consider the effects of the commands before using this tool.

The PDC user interface is provided by the command `epg_pdc_control`. It can be used to manually make a PDC package before the end of the month, to check the status of the statistics collection, and for configuration changes.

The EPG configuration information is retrieved at the end of the PDC log session. The EPG configuration and traffic or load data must be correlated to ensure accurate representation of the EPG characteristics after parsing the PDC log using the PDC tool.

To ensure the correlation between the EPG configuration and the PDC log session, stop the PDC log, perform the EPG configuration changes, and then re-start the new PDC log session.

Note: The user account, from which PDC is manually run, must have the right user privileges. For more information, refer to Security Management.

The syntax of the control command is as follows:

```
epg_pdc_control [-h|-s] command
```

-h	Display tool help
-s	Display tool summary

The following commands are available:

Command	Description
status	Print status information about PDC.
start	Start scheduling PDC runs. See interval for details on how often statistics collection takes place.
runinfo	List commands executed in the previous run, with the result of execution (`ok` or `not ok`).
user	Show which user is defined for PDC.
user=<username>	Username of an administrator account that can execute ECCLI commands.(1)
nodetype=<type>	Manual setting of the node type (sgw, pgw, or combined).

Command	Description
interval=num	Set the interval (in seconds). The user may append `m` or `h` to indicate minutes or hours. For example, `interval=5m`. The interval is always counted in time periods shifted by 3 minutes in order to avoid collision with other periodic jobs. For example, if at 15:12 a user sets `interval=1h`, then the next run will be at 16:03; if at 15:12 a user sets `interval=5m`, then the next run will be at 15:18. If an interval is not set, the default interval is 15 minutes. Information is then collected at 3, 18, 33, and 48 minutes past the hour. It is recommended that only intervals of 5, 15, or 60 minutes are used on live nodes. If the interval is too low, it is possible that all requested commands may not be printed within the given interval. If that occurs, PDC automatically increases the interval to the next predefined interval. For example, if an interval of 5 minutes was used, and it was not enough, PDC increases the interval to 15 minutes.
package	Makes a PDC package. User generated PDC packages are named `<month>_<hostname>_<timestamp>_ggsn_pdc.tar.gz`, where `<month>` is the current month, `<hostname>` is the EPG node name, and `<timestamp>` is the date and time when the package is generated. For example, `Apr_epg104-1_2012-04-11-2159_ggsn_pdc.tar.gz`. A maximum of six user-initiated PDC packages are saved on the EPG.
stop	Stop scheduling PDC runs, make a package, and flush all old statistics. Consider using `stop_sched` to preserve data.
stop_sched	Stop scheduling PDC runs and preserve collected data.
store_cust_run =/somewhere/file.txt	Configure a new list of commands to run in addition to default run. By adding a custom run configuration, the PDC starts to collect additional information at the next run. A new configuration replaces the old one.
delete_cust_run	Delete custom run.

Command	Description

feature_add=[pmip|l2tp|ebm]	Activate a specific feature.(2)
	
	Set up a pdc job for a specific feature.
	For PMIP:
	`epg pgw statistics of pmip`
	`epg pgw statistics of gre`
	For L2TP:
	`epg pgw statistics of l2tp-tunnel`
	For EBM:
	`epg sgw ebm statistics`
	`epg pgw ebm statistics`

feature_del=[pmip|l2tp|ebm]	Delete an activated feature.
feature[s]	List the activated feature(s).

(1) The configuration of `username` is persistent, meaning it survives EPG reload.
(2) This command is introduced to activate some frequently used features in an easy way.

For an example of the output from the command **epg_pdc_control status**, see Example 1.

```
bash-3.2$ epg_pdc_control status
current date and time: 2018-06-01_10:57:02
pdc node type: combined
status: running
current interval: 900 sec
configured interval: 900 sec
next run in: 358 sec
next run at: 11:03:00
custom-run: off
keep raw logs: off
accumulated data starts: 2018-06-01 01:03:01
monthly package: 1st of month, 1:00am
disk space used by PDC: 22344 Kbytes, partition 9% full
cli user: erv
result of last run: ok
duration of last run: 6sec
```

Example 1 Output from the Command epg_pdc_control status

For an example of the output from the command **epg_pdc_control runinfo**, see Example 2.

```
bash-3.2$ epg_pdc_control runinfo
Statistics run started at: 2018-05-31 17:18:01
Type Result => Command
----------------------
d ok => show_chassis_routing-engine
d ok => show chassis
d ok => epg node internal-debug status
d ok => epg pgw statistics of brief
d ok => epg pgw statistics of ppb
d ok => epg pgw statistics of charging
d ok => epg pgw internal-debug pisc-flows level detail
d ok => epg pgw statistics of qci
d ok => epg node internal-debug execute cmd-string "bdcu all show ggsn driver x
d ok => epg pgw statistics of high-availability
d ok => epg pgw internal-debug inspection cmd-string "show performance-counters
d ok => epg node user-plane show-drops
d ok => epg node olp status value detail
d ok => epg sgw statistics
d ok => epg sgw statistics-arp
d ok => epg sgw statistics-qci qci all detail-level detail
a ok => epg pgw apn apn1.com statistics
a ok => epg pgw apn apn2.com statistics
a ok => epg pgw apn apn3.com statistics
a ok => epg pgw apn apn4.com statistics
a ok => epg pgw apn apn5.com statistics
----------------------
Types: "a" auto selection, "c" custom run, "d" default run, "m" manual selectio
```

Example 2 Output from the Command epg_pdc_control runinfo

3.1.5 Configuring PDC for Custom Statistics Data Collection

To configure collection of additional statistics for special scenarios, custom commands can be included in the PDC. The ECCLI commands and parsers to run are stored in the PDC collect engine.

Note: PMIP, GRE, L2TP protocol, and EBM can be configured by the **`feature_add=[pmip|l2tp|ebm]`** command directly.

Use the command `epg_pdc_control store_cust_run=/somewhere/file.txt` t o configure custom commands. This command replaces all previously configured custom commands with the ones specified in the file.

It is recommended to keep a copy of the installed file on a persistent place, such as `/flash/` for future reference.

Use the command **`epg_pdc_control runinfo`** to see which custom run commands that are active.

The following procedure describes how to enable the statistics collection:

1. Create a custom run file to install (store) in the PDC.

   `cp /flash/services/epg/ttx/run-cust.config.example /flash/services/epg/ttx/run-cust.config`

   Note: If custom run is already configured, edit the existing file saved from the latest configuration.

   To edit the existing file, use the following command:

   `vi /flash/services/epg/ttx/run-cust.config`

   In the file, remove the comment hashes (#) at the beginning of the statements, for example:

   com;config epg sgw statistics;sgw-statistics.txt;epg_pdc_parse_sgw_statistics.pl;

   Note: An alias cannot be configured for `run-cust.config`.

2. Add custom run to the PDC.

   `epg_pdc_control store_cust_run=/flash/services/epg/ttx/run-cust.config`

   Note: It is recommended to keep a copy of the `/flash/services/epg/ttx/run-cust.config` file on a persistent storage, `/flash/`, as a reference for later changes.

3. Check that custom run is enabled.

   Run **`epg_pdc_control status`** in a shell. For an example of the output from the command with custom run enabled (on), see Example 3.

4. Check for no errors in command list.

   Run **`epg_pdc_control runinfo`** in a shell. For an example of the output from the command, see Example 4.

```
bash-3.2$ epg_pdc_control status
current date and time: 2018-05-31_17:24:32
pdc node type: combined
status: running
current interval: 900 sec
configured interval: 900 sec
next run in: 508 sec
next run at: 17:33:00
custom-run: off
keep raw logs: off
accumulated data starts: 2018-05-01 01:03:01
monthly package: 1st of month, 1:00am
disk space used by PDC: 40536 Kbytes, partition 9% full
cli user: erv
result of last run: ok
duration of last run: 7sec
```

Example 3 Output from epg_pdc_control status

```
bash-3.2$ epg_pdc_control runinfo
Statistics run started at: 2018-05-31 17:18:01
Type Result => Command
----------------------
d ok => show_chassis_routing-engine
d ok => show chassis
d ok => epg node internal-debug status
d ok => epg pgw statistics of brief
d ok => epg pgw statistics of ppb
d ok => epg pgw statistics of charging
d ok => epg pgw internal-debug pisc-flows level detail
d ok => epg pgw statistics of qci
d ok => epg node internal-debug execute cmd-string "bdcu all show ggsn driver x
d ok => epg pgw statistics of high-availability
d ok => epg pgw internal-debug inspection cmd-string "show performance-counters
d ok => epg node user-plane show-drops
d ok => epg node olp status value detail
d ok => epg sgw statistics
d ok => epg sgw statistics-arp
d ok => epg sgw statistics-qci qci all detail-level detail
a ok => epg pgw apn apn1.com statistics
a ok => epg pgw apn apn2.com statistics
a ok => epg pgw apn apn3.com statistics
a ok => epg pgw apn apn4.com statistics
a ok => epg pgw apn apn5.com statistics
----------------------
Types: "a" auto selection, "c" custom run, "d" default run, "m" manual selectio
```

Example 4 Output from epg_pdc_control runinfo

3.1.6 APN Statistics Collection

3.1.6.1 Manually Selected APN Statistics Collection

The PDC can collect statistics for up to 20 of the configured APNs. The APNs, for which the statistics are to be generated, must be configured in the PDC. Although automatic APN selection exists, it is recommended to make an active selection of which APNs the PDC shall generate statistics for. If no manual selection is made, the PDC command `epg_pdc_control status` issues a warning, even if an automatically selected list of APNs exists.

To collect per-APN statistics, a list of APN names must first be generated. To generate a list of APN names, use the following command:

```
epg_pdc_apn_config
```

This list can contain up to 20 APN names.

To generate a list automatically, use the following command:

```
epg_pdc_apn_config `epg_pdc_apn_config –L`
```

Note: Perform this configuration during software installation procedures. For more information, refer to Deploying Virtual EPG, and Software Upgrade for Virtual EPG.

The APN configuration list is persistent, meaning it is unaffected by an EPG restart.

If an APN configuration is changed, for example, an APN is unconfigured, or a new one is configured, the list must be regenerated.

To display a list of all available APNs, use the following command:

```
epg_pdc_apn_config -l
```

Note: In the above command, `-l` uses the letter l, and not the digit 1.

To collect statistics for a set of APNs, use the following command format:

```
epg_pdc_apn_config <apn-name1> <apn-name2> ...
```

For example, to collect statistics for a set of APNs that includes APN 1, APN 2, and APN 3, use the following command:

```
epg_pdc_apn_config apn-name1 apn-name2 apn-name3
```

This includes the following commands in the data collection:

```
epg pgw apn apn-name1 statistics
epg pgw apn apn-name2 statistics
epg pgw apn apn-name3 statistics
```

3.1.6.2 Automatic APN Statistics Collection

During normal operation, the PDC tries to automatically configure statistics collection for the configured APN or APNs up to the PDC maximum of 20 APNs. The APNs are selected in the order they appear in the running configuration. This functionality is a complement to the manual configuration using the command `epg_pdc_apn_config`, see Section 3.1.6.1 on page 15, where the APNs can be selected based on importance, amount of traffic, and so on.

If there is a mix of manually and automatically selected APNs, the PDC first generates statistics for the manually configured APNs. The PDC then continues to generate statistics for the automatically configured APNs up to the PDC maximum. If the same APN has been selected both manually and automatically, the PDC only generates statistics for the specific APN once.

The lists of the APNs are stored as below:

— The list of manually configured APNs is stored in the file `/flash/services/epg/ttx/run-apn.cfg`

— The list of automatically selected APNs is stored in the file `/tmp/run-apn-list.cfg`

The lists are updated or created at:

— PDC startup (about 5-10 minutes after reboot, restart, or startup)

— PDC statistics interval collection, if the file `/tmp/run-apn-list.cfg` does not exist

— Change of month

Note: The old file including manually defined APNs, `/flash/services/epg/ttx/run-apn.config`, is obsolete due to the new format. If such a file exists in the system, the file is automatically converted into a new file with the new format of the manually configured APNs.

3.2 Support Package

If a Customer Service Request (CSR) is raised for an EPG problem, it is recommended that a support package is generated and attached to the CSR. To generate the support package, use the **`support_package`** command.

The administrator account used for PDC is also used for the creation of a support package. For more information on the configuration of administrator accounts, see Section 3.1 on page 4 and refer to Security Management.

The support package contains the following:

— Node configuration

— System log files

- PDC packages for previous months
- PDC packages for current month
- Hardware version
- Software version
- Node status

The EPG includes different files and logs for the support package based on the features that are configured and activated on the node. For more information on CSRs, refer to Data Collection Guideline.

Support packages are created in the following directory:

```
/var/log/services/epg/ttx/sp/
```

The support packages are named `support_package-<nodename>-<timestamp>.tar.gz`, where `<nodename>` is the EPG node name, and `<timestamp>` is the date and time when the package is created. For example, for an EPG with node name `epg75`, the support package is named `support_package-epg75-2015-11-21-2323.tar.gz`.

Note: The execution of `support_package` may fail under some directories such as CDR due to access restriction.

3.2.1 ECCLI Commands Used for Support Package

Note: Some commands that include `internal-debug` can be traffic affecting and should not be executed individually outside the support package script, unless specific instructions/permissions to do so are provided by Ericsson.

The following commands can be used in the generation of the support package:

- **`show rfm disk-usage`**
- **`show rfm statistics`**
- **`epg node internal-debug session-resilience lim clients`**
- **`epg node internal-debug session-resilience lim master-table pe-type all`**
- **`epg node internal-debug session-resilience lim history-li pe-type sgwc load-index all`**
- **`epg node internal-debug session-resilience lim history-li pe-type pgwc load-index all`**
- **`epg node internal-debug session-resilience lim history-li pe-type diameterendpoint load-index all`**

- `epg node internal-debug session-resilience lim history-li pe-type u load-index all`
- `epg node internal-debug session-resilience lim manager-status`
- `show running-config epg pgw diameter`
- `show running-config epg node diameter`
- `show icr client`
- `show icr history`
- `show icr prefixes`
- `show icr state`
- `show icr statistics`
- `show icr log`
- `epg node fault-management active-notifications`
- `show fm alarm`
- `epg pgw dns status`
- `epg node status`
- `epg pgw internal-debug session-resilience ppb-statistics-all`
- `epg pgw service-identification configuration`
- `show chassis`
- `show card detail`
- `show hardware detail`
- `show tech-support export`
- `show crashfiles`
- `epg pgw flow-cache status`
- `'epg node internal-debug execute cmd-string "show istats gsc"'`
- `'epg node internal-debug execute cmd-string "show istats dhcp"'`
- `'epg node internal-debug execute cmd-string "show istats dbp"'`
- `'epg node internal-debug execute cmd-string "show istats sacc"'`
- `'epg node internal-debug execute cmd-string "show istats gtp"'`

```
— 'epg node internal-debug execute cmd-string "show istats
  gtpv2"'
— 'epg node internal-debug execute cmd-string "show istats pmip"'
— 'epg node internal-debug execute cmd-string "show istats
  radius"'
— 'epg node internal-debug execute cmd-string "show istats pdp"'
— 'epg node internal-debug execute cmd-string "show istats li"'
— 'epg node internal-debug execute cmd-string "show istats urm"'
— 'epg node internal-debug execute cmd-string "show istats ebm"'
— 'epg node internal-debug execute cmd-string "show istats data"'
— 'epg node internal-debug execute cmd-string "show istats gccp"'
— 'epg node internal-debug execute cmd-string "show istats gucp"'
— 'epg node internal-debug execute cmd-string "show istats gtcp"'
— 'epg node internal-debug execute cmd-string "show istats gnsp"'
— 'epg node internal-debug execute cmd-string "show istats gpcp"'
— 'epg node internal-debug execute cmd-string "show istats gtpb"'
— 'epg node internal-debug execute cmd-string "show istats
  apnmgt"'
— 'epg node internal-debug execute cmd-string "show istats
  charging"'
— 'epg node internal-debug execute cmd-string "show istats rpd"'
— 'epg node internal-debug execute cmd-string "show istats core"'
— 'epg node internal-debug execute cmd-string "show istats
  etfctrl"'
— 'epg node internal-debug execute cmd-string "show istats
  pgw.sessionengine*"'
— 'epg node internal-debug execute cmd-string "show istats
  pgw.service*"'
— 'epg node internal-debug execute cmd-string "show istats
  pgw.ppb*"'
— 'epg node internal-debug execute cmd-string "show istats
  pgw.admissioncontrol*"'
```

```
— 'epg node internal-debug execute cmd-string "show istats l2tp"'

— 'epg node internal-debug execute cmd-string "show istats ppb"

— 'epg node internal-debug execute cmd-string "show istats
  pgw.l2tp*"

— 'epg node internal-debug execute cmd-string "bdcu all show
  contentfiltering"'

— 'epg node internal-debug execute cmd-string "bdcu all show ggsn
  driver"'

— 'epg node internal-debug execute cmd-string "bdcu all show ggsn
  ipudp"'

— 'epg node internal-debug execute cmd-string "bdcu all show ggsn
  ipudp fragments"'

— 'epg node internal-debug execute cmd-string "bdcu all show ggsn
  gtp payload"'

— 'epg node internal-debug execute cmd-string "bdcu all show ggsn
  gtp gucp statistics"'

— 'epg node internal-debug execute cmd-string "bdcu all show ggsn
  gtp charging"'

— 'epg node internal-debug execute cmd-string "bdcu all show ggsn
  l2tp payload"'

— 'epg node internal-debug execute cmd-string "bdcu all show ggsn
  l2tp signaling"'

— 'epg node internal-debug execute cmd-string cmd-string "bdcu
  all show ggsn l2tp gtcp statistics"'

— 'epg node internal-debug execute cmd-string "bdcu all show
  piscapi"'

— 'epg node internal-debug execute cmd-string "bdcu all show
  sgwu"'

— 'epg node internal-debug execute cmd-string "show log *"'

— lm refresh-license-inventory

— lm publish-license-inventory

— show lm

— /opt/services/epg/bin/gcdc_IPOS_rp gifmd printtsm

— /opt/services/epg/bin/gcdc_IPOS_rp gifmd print_ipc_history
```

- **/opt/services/epg/bin/gcdc_IPOS_rp gifmd printroutes**
- **/opt/services/epg/bin/gcdc_IPOS_rp gifmd print_subnet_routes**
- **/opt/services/epg/bin/gcdc_IPOS_rp gifmd print_traffic_slice_info**
- **/opt/services/epg/bin/gcdc_IPOS_rp gifmd print_gateways**

The following commands can be used to analyze Route Processor Switch (RPSW) cards:

- **show card**
- **show tsm tsft-map**
- **show card card fabl tsm papt**
- **show card card fabl tsm tsft-tbl 2**
- **show card card fabl tsm tsft-tbl 3**
- **show card card fabl tsm tsft-tbl 4**
- **show card card fabl tsm tsft-tbl 30**
- **show card card fabl tsm tsft-tbl 31**
- **show card card fabl tsm tsft-tbl 32**
- **show card card fabl tsm tsft-tbl 33**
- **bdcu internal data**

3.2.2 Logs and Files Collected in the Support Package

The following files are collected in a support package:

- List of available core dumps in `/md`
- List of files in the `/md` directory
- List of files in the `PDC` directory
- `show_epg_version`
- `health_check data`
- The PDC package of the previous month, if it exists
- PDC status
- PDC runinfo

— OS data:
  - Process list
  - System uptime
  - Kernel settings

The following logs and files are collected for the support package:

— `/flash/.isp.log`
— `/flash/ericsson.cfg`
— `/flash/services/epg/var/data/*`
— `/flash/services/epg/heuristics/Heur*`
— `/flash/services/epg/ttx/*`
— `/md/loggd_persistent.log*`
— `/md/loggd_startup.log*`
— `/md/autid.log`
— `/opt/services/epg/bin/ttx/ssr_ttx_inst.version`
— `/opt/services/epg/etc/*`
— `/tmp/candidateCfg`
— `/tmp/ttxd.log`
— `/tmp/zonefile`
— `/var/log/auth.log`
— `/var/log/cli_commands`
— `/var/log/cli_startup.out`
— `/var/log/daemon.log`
— `/var/log/debug`
— `/var/log/dmesg`
— `/var/log/kern.log`
— `/var/log/messages`
— `/var/log/rfm_2*.log`
— `/var/log/aspd_2*.log`

- /var/log/servmon_1_2*.log
- /var/log/services/epg/fm/alarm-history
- /var/log/services/epg/fm/alarm-history[0-9]
- /var/log/services/epg/licd/*
- /var/log/services/epg/pdc/epg_pdc_messages.log[0-9]*
- /var/log/services/epg/pdc/kpi/*
- /var/log/services/epg/ttx/alp.log
- /var/log/services/epg/ttx/alp.log.*
- /var/log/syslog
- /var/log/user.log

The last megabyte of these files is collected:

- /var/log/com.log
- /var/log/com.trace
- /opt/com/comea/log/snmpd.log

If the RPSW card is a standby card, the following files from the /md directory are also collected:

- loggd_persistent
- loggd_startup

3.3 Automatic Log Package

The EPG Automatic Log Package (ALP) collects log files and command output files after a core dump. The ALP can be used to collect additional information that is not collected by the Support Package.

The log collection starts 30 seconds after a core dump event. A script runs selected statistics commands and collects the output. This output and the log files are stored in an ALP.

During the collection of the ECCLI command printouts, the ECCLI commands are executed as the configured PDC user. If no PDC user is defined, the ECCLI commands are not executed during the log collection.

The ALP is stored in the /md directory and is named <yyyymmdd_HHMMSS>_<hostname>_alp.tgz. For example, for an EPG with node name epg75, the ALP is named 20150813_103225_epg75_alp.tgz.

The ALP log file is stored in the `/var/log/services/epg/ttx/` directory and is named `alp.log`.

A sequence diagram for the ALP is shown in Figure 1.

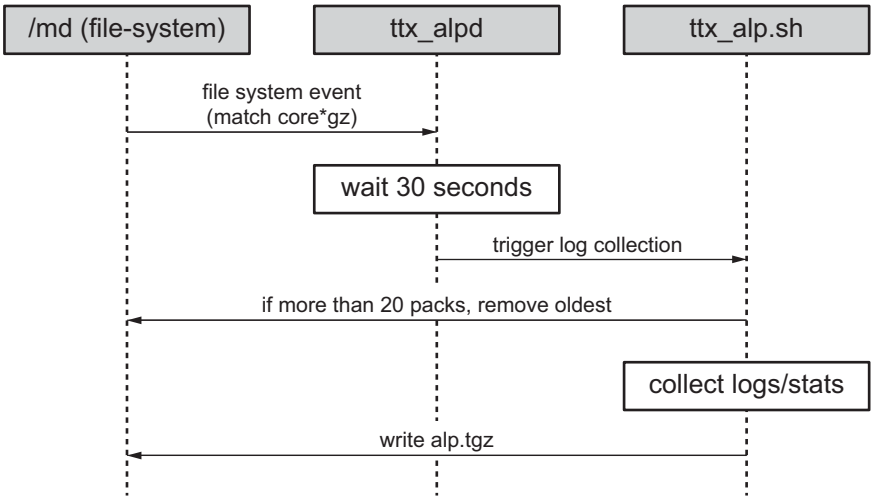

Figure 1 Sequence Diagram for the ALP

3.4 System Storage Cleanup

The command **`system_storage_cleanup`** can be used to facilitate the cleanup of disk space on the Route Processor (RP). The following files can be removed:

— coredumps

— Custom PDC packages (all PDC packages with `<timestamp>` in the name)

— PDC accumulated raw files (if the `epg_pdc_control` parameter `keep` is set to 1)

— Support Packages

— Archived system log files (`loggd_persistent` and `loggd_startup`)

3.5 CDR Decoder

The CDR decoder is used to decode CDR files into human readable text format.

CDR files encoded in Abstract Syntax Notation One (ASN.1) Basic Encoding Rules (BER) are converted to plain text or Distinguished Encoding Rules (DER) encoded CDRs.

Decoders are available for each supported release of the charging standard (Releases 6, 7, 8, 9, 13, and 15). For more information, refer to CDR Format for the SGW or CDR Format for the GGSN and PGW.

Note: This tool is available only in the remote Toolbox, since it is not safe to decode large CDR files on the EPG.

The syntax of the CDR Decoder command is:

```
cdr_decoder -h | -s | -o format | -r cdrrelease infilename ...
```

Option	Description
-r cdrrelease	Indicates 3GPP CDR release used for CDR decoding. It must be the same as the format configured on the node. Available values for **cdrrelease**: • R6 • R7 • R8 • R9 • R13-pgw • R13-sgw • R15
infilename	Specifies the name of the CDR input file.
-h	Displays the tool help.
-s	Displays the tool summary.
-o format	Specifies the output format, which can be one of the following: • **text** for plain semi-structured text format • **der** for DER • **xer** for XML Encoding Rules (XER) (default) • **null** for verifying the input, without decoding

For example, to decode the CDR file `epg2-2_20120611103807_481` and save output to the file `epg2-2_20120611103807_481.decoded`, enter the following command:

```
cdr_decoder -r R7 epg2-2_20120611103807_481 > epg2-2_201206111038
07_481.decoded
```

3.6 ACR/CCR Decoder

The ACR/CCR decoder is used to decode EPG Rf ACR files and EPG GY CCR-Terminate messages that are written to file, into human readable text format.

The tool supports both SGW and PGW generated RF ACR files, and PGW generated GY CCR-T files.

Note: This tool is available only in the remote Toolbox, since it is not safe to decode large ACR/CCR files on the EPG.

Attention!

This program is only intended for debugging purposes and must not be used in any live environment as part of a product chain.

The ACR/CCR decoder is written in Java®, and JRE version 1.8 must be installed on the machine that wants to run it.

The syntax of the ACR/CCR Decoder command is as follows:

```
java -jar acrccrdecoder.jar acr/ccrdirectorypath/acr/ccrfilepath
| -d / --debug | -o / --outputfolder outputfolderpath | -v /
--version |
```

Option	Description
`-d, --debug`	Enables debug logs from the program.
`acr/ccrdirectorypath`	Specify the full path to the folder where the ACR/CCR files that are subject to be decoded are being stored.
`acr/ccrfilepath`	Specify the full path to the specific ACR/CCR file that is subject to be decoded.
`-v, --version`	Displays the current version of the ACR/CCR Decoder. Provide the version number in inquiries.
`-o, --outputfolder outputfolderpath`	By default, the tool creates a folder named `DecodedAcrsCcrs` in the same folder as the ACR/CCR files to be decoded. By specifying the output folder, the textual representation of the ACR/CCR files is stored in the output folder.(1)

(1) If the ACR/CCR decoder is invoked with the option `-o`, the custom output folder must already exist.

For example, to decode the ACR/CCR file `epg2-2_20120611103807_481` and save output to the file `DecodedAcrsCcrs/epg2-2_20120611103807_481.txt`, enter the following command:

```
java -jar acrccrdecoder.jar epg2-2_20120611103807_481
```

For example, to decode all the /CCR files in the folder **c:\Ccrs** on a windows system, and to save the result to **c:\DecodedCcrs**, enter the following command:

```
java -jar acrccrdecoder.jar c:\Ccrs -o c:\DecodedCcrs
```

3.7 EBM Data Parsing

Warning!

EBM data parsing is intended to be used as a troubleshooting tool in a limited way.

The following tools can be used on a remote platform for recording and parsing binary Event-Based Monitoring (EBM) data:

— The EBM stream recorder (`ebm_stream_recorder` or `ebm_stream_recorder64`) is used to record streamed EBM data into a binary EBM log file.

  Note: The `ebm_stream_recorder` file is 32-bit. The `ebm_stream_recorder64` file is 64-bit for Linux OS.

  A server that runs the EBM stream recorder tool can be treated as an EBM server.

— The EBM decoder (`parse_ebm_log.pl`) is used to parse binary EBM log files into readable text files.

For general information about EBM, refer to Event-Based Monitoring.

For information about the EBM output, refer to Event-Based Monitoring Output.

For a list of EBM supported events, refer to Event-Based Monitoring Events and Parameters

For a complete list of EBM cause codes and sub cause codes, refer to Event-Based Monitoring Cause Codes.

For information about how to operate EBM, refer to Event-Based Monitoring Configuration.

3.7.1 EBM Data Parsing Workflow

Binary EBM data streams can be saved to file by using the EBM stream recorder. The file can then be parsed into a readable text file by using the EBM decoder.

Warning!

To avoid capacity degradation, the EBM stream recorder and EBM decoder should only be executed on a remote platform

EBM Data Parsing workflow:

1. Install the Toolbox.
2. Save EBM data stream to file.
3. Parse binary EBM log files

EBM Data Parsing is provided as a part of the remote Toolbox tarball. This package must be fetched from the EPG and unpacked on a remote workstation which has Perl™ installed.

3.7.2 Saving an EBM Data Stream

Use the EBM stream recorder to save one or several EBM data streams simultaneously from an EPG to a binary EBM log file. When using other tools to save EBM data streams to a file, ensure that the output file contains EBM data only; otherwise, the file cannot be parsed.

The EBM stream recorder saves the binary EBM log files in the directory where it is executed. When the file size reaches 10 MB, the recorder closes the file and opens a new file for the same Reporting Output Period (ROP). It saves the files using the following naming convention:

```
A<Startdate>.<Starttime><UTC>-<Enddate>.<Endtime><UTC>_<ROPindex
>_ebme.<Index>
```

The filename parts are defined as follows:

Part	Description
`Startdate`	The date when the file recording is started, with the format `<yyyy><mm><dd>`.
`Starttime`	The time when the file recording is started, with the format `<hh><mm>`.
`UTC`	The local time differential from Coordinated Universal Time (UTC).

Part	Description
Enddate	The date when the file recording is stopped, with the format <yyyy><mm><dd>.
Endtime	The time when the file recording is stopped, with the format <hh><mm>.
ROPindex	The sequence number of the log file in the ROP. It specifies the recording order of the log files in one ROP.(1)
Index	The sequence number of the log file. It specifies the recording order of the log files.(2)

(1) The ROPindex is restarted from one after the end of each ROP or after reaching 255.
(2) The index is restarted from one after reaching 255.

To save an EBM data stream in a file, use the ebm_stream_recorder or ebm_stream_recorder64 command. The syntax of the ebm_stream_recorder command is as follows:

```
ebm_stream_recorder -h | -s | -i <addr> -p <port> [-r <rop>]
```

Option	Description
-h	Displays the tool help.
-s	Displays the tool summary.
-i <addr>	Specifies the IP address to monitor incoming EBM data streams, that is, the IP address to which the EPG sends the EBM data stream.
-p <port>	Specifies the port number to monitor incoming EBM data streams, that is, the port number to which the EPG sends the EBM data stream.
-r <rop>	Specifies the ROP in minutes. The ROP for the EBM log file can be set to any of the following values: 1, 5, 15, 30, and 60 minutes. The default ROP is 15 minutes.

The following example shows saving an EBM Data Stream to File:

```
> ./ebm_stream_recorder -i 10.10.10.1 -p 58001 -r 30
```

The following example shows an Output File from EBM Stream Recorder:

```
A20110406.1630+0200-20110406.1700+0200_1_ebme.5
```

3.7.3 Parsing Binary EBM Log Files

Ensure that the following prerequisites are met before parsing binary EBM log files:

— Perl™ installed on a remote workstation where EBM Data Parsing takes place.

— The `ebm_event_specification.xml` file and the `parse_ebm_log.pl` file are located in the same directory on the remote platform.

— The binary EBM log files only contain EBM data, and are named according to the naming convention described in previous section.

Use the following command to parse a binary EBM log file into a readable text file:

```
parse_ebm_log.pl -h | -s

parse_ebm_log.pl [-f <log_file>] [-d <dir>] [-S] [-o
<output_file>] [-l [-p <delimiter>]] filters

[-u] [-e '<event-type>[,...]'] [-v '<event-trigger>[,...]'] [-c <cause_code>[,
```

Option	Description
`-h`	Displays the tool help.
`-s`	Displays the tool summary.
`-f <log-file>`	Specifies the EBM log file to parse. If the EBM log file to parse is not specified, the EBM decoder parses all the log files in the current working directory, or the directory specified using option -d.
`-d <dir>`	Specifies the directory where binary EBM log files are located. If no directory is specified, the log files in the current working directory are parsed.
`-o <output-file>`	Specifies the output file. The output is displayed on the console if no output file is specified.
`-S`	Displays a summary of found events in the parsed log files.
`-l`	Prints as table. If the option -p is not specified, then the default delimiter ";" is used.
`-p <delimiter>`	Prints event records separated by the specified delimiter, and overwrites the default delimiter. Option -l must be specified for the option -p to work.
`-u`	Filters unsuccessful events.

Option	Description
`-e <event-type>`	Filters the event type specified. "*" can be used to match any character.
`-v <event-trigger>`	Filters the event trigger specified. "*" can be used to match any character.
`-c <cause-code>`	Filters the cause code (decimal value) specified.
`-z <sub-cause-code>`	Filters the sub-cause code (decimal value) specified.
`-i <imsi-number>`	Filters the IMSI number specified.
`-n`	Filters unauthenticated IMSI numbers.
`-t <tac>`	Filters the Type Approval Code (TAC) specified. The TAC is the first eight digits in the IMEI number.
`-a <apn>`	Filters the Access Point Name (APN) specified.

Example – Filter by Unsuccessful Events:

```
> ./parse_ebm_log.pl -u -f /tmp/A20110406.1630+0200-20110406.1700+0200_1_ebm
```

Example – Filter by Create Session Events and All Events Ending with Deletion:

```
> ./parse_ebm_log.pl -e 'session_creation,*deletion' f /tmp/A20110406.1630+0
```

Example – Filter by IMSI Number 240999800416785 and 240999802602314:

```
> ./parse_ebm_log.pl -i 240999800416785,240999802602314 -f /tmp/A20110406.16
```

Example – EBM Summary of All Logs Located in the Directory /tmp/ebm_logs:

```
> ./parse_ebm_log.pl -S -d /tmp/ebm_logs
```

An optional parameter that is not present in the event stream is displayed as 'undefined' by the EBM decoder. For more information on optional parameters, refer to Event-Based Monitoring.

3.8 UE Trace Tools

All UE Trace session data is stored in the XML format according to SoC for 3GPP TS 32.423. The UE Trace log files for the control plane are stored on the Node Management Board (NMB) in the `/var/log/services/epg/uetrace` directory.

The following tools can be used on a remote platform for converting control plane UE Trace log files:

— UE Trace to PCAP Converter

— UE Trace Parser

If support has been activated for the following protocols, they can be used to retrieve the UE trace log files from an external server or transfer UE trace log files to an external server:

— Secure File Transfer Protocol (SFTP)

— Secure Copy Protocol (SCP)

For more information on UE Trace, refer to UE Trace.

For more information on the UE Trace log file format, refer to UE Trace Log.

3.8.1 UE Trace to PCAP Converter

UE Trace log files can be converted into Packet Capture (PCAP) format by using the UE Trace to PCAP converter, which consists of the `epg_uetrace2pcap.pl` script.

UE Trace log files converted into PCAP format can be viewed using Wireshark (version 1.10.7 or later).

In the converted output files, the User Datagram Protocol (UDP) headers contain the real UDP ports and the IP headers contain the real Internet Protocol (IP) addresses. The other data fields in the UDP headers and IP headers, and all the data fields in the link layer headers contain dummy data constructed by the UE Trace to PCAP converter.

Warning!

To avoid capacity degradation, the UE Trace to PCAP converter should only be executed on a remote platform.

Note: The UE Trace to PCAP converter is verified to work on Red Hat Enterprise Linux Server 5.3.

Figure 2 shows the workflow for converting UE Trace log files.

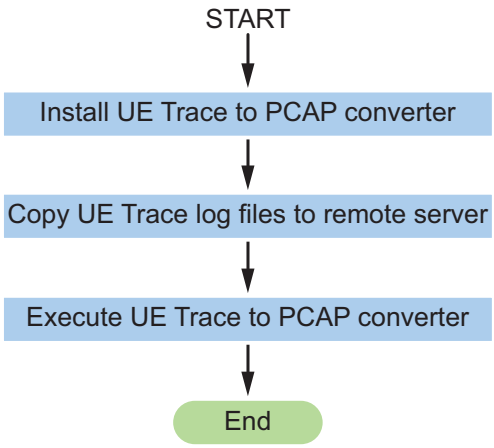

Figure 2 UE Trace Log File Conversion Workflow

For information about the UE Trace log files, refer to UE Trace Log.

3.8.1.1 Installing the UE Trace to PCAP Converter

The UE Trace to PCAP converter is provided as a part of the remote Toolbox tarball. This package must be fetched from the EPG and unpacked on a remote workstation which has Perl™ (version 5.6.1 or later) installed. For more information, see Section 2.1 on page 2.

3.8.1.2 Copying the UE Trace Log Files to the Remote Server

Copy the UE Trace log files to the remote server where the UE Trace to PCAP converter is located.

3.8.1.3 Executing the UE Trace to PCAP Converter

In addition to the prerequisites described in Section 3.8.1 on page 32, ensure that the UE Trace log files are located on the same remote platform as the UE Trace to PCAP converter.

For information on how to fetch UE Trace log files and where they are stored, see Section 3.8 on page 31.

Convert UE Trace log files into PCAP format by using the following command:

```
user@host>epg_uetrace2pcap.pl [-h | -s | -p | -r | [-f logfile |
-d input-directory] [-o output-directory]]
```

Syntax

Option	Description
-h	Displays the UE Trace to PCAP converter help text.
-s	Displays information about the UE Trace to PCAP converter.
-p	Creates one PCAP file per UE device. If both option **-p** and option **-r** are specified, the PCAP file is created per UE Trace session. If neither of the option **-p** nor option **-r** is specified, the PCAP file is created per UE Trace log file.
-r	Creates one PCAP file per UE Trace session.
-f logfile	Specifies the UE Trace log file to convert into PCAP format. If no UE Trace log file is specified, all log files in the current working directory, or the directory specified using option **-d** are converted.
-d input-directory	Specifies the input directory where UE Trace log files are located. If no input directory is specified, the UE Trace log files in the current working directory are converted.
-o output-directory	Specifies the output directory for the PCAP files. If no output directory is specified, the output files are stored in the current working directory.

```
user@host>./epg_uetrace2pcap.pl -f B20141205.081502+0100-PGW.epg18-1-pgw –p
Processing: B20141205.081502+0100-PGW.epg18-1-pgw
Result:     B20141205.081502+0100-PGW.epg18-1-pgw-IMSI-100559002335945.pcap
Result:     B20141205.081502+0100-PGW.epg18-1-pgw-IMSI-100559002335946.pcap
Result:     B20141205.081502+0100-PGW.epg18-1-pgw-IMSI-100559002335947.pcap
```

Example 5 Converting a UE Trace Log File to PCAP Format per UE Device.

```
user@host>./epg_uetrace2pcap.pl -f B20141205.081502+0100-PGW.epg18-1-pgw –r
Processing:  B20141205.081502+0100-PGW.epg18-1-pgw
Result:     B20141205.081502+0100-PGW.epg18-1-pgw-TraceSessionRef-1000000000
Result:     B20141205.081502+0100-PGW.epg18-1-pgw-TraceSessionRef-1000000000
Result:     B20141205.081502+0100-PGW.epg18-1-pgw-TraceSessionRef-1000000000
```

Example 6 Converting a UE Trace Log File to PCAP Format per UE Trace Session.

3.8.2 UE Trace Parser

UE Trace log files can be parsed into plain text format by using the UE Trace Parser, which consists of the `epg_uetracer_parser.pl` script.

UE Trace log files in plain text format can be viewed using a plain text editor.

Warning!

To avoid capacity degradation, the UE Trace Parser should only be executed on a remote platform.

For information about the UE Trace log files, refer to UE Trace Log.

3.8.2.1 Installing the UE Trace Parser

The UE Trace Parser is provided as a part of the remote Toolbox tarball. This package must be fetched from the EPG and unpacked on a remote workstation which has Perl™ (version 5.6.1 or later) installed. For more information, see Section 2.1 on page 2.

3.8.2.2 Copying the UE Trace Log Files to the Remote Server

Copy the UE Trace log files to the remote server where the UE Trace parser is located.

3.8.2.3 Executing the UE Trace Parser

In addition to the prerequisites described in Section 3.8.2 on page 35, ensure that the UE Trace log files are located on the same remote platform as the UE Trace Parser.

Parse UE Trace log files into plain text format by using the following command:

```
user@host>epg_uetracer_parser.pl [-h | -s | [-f logfile | -d
input-dir] [-o output-dir] [-i identity] [-c interface] [-r
node-type] ]
```

Syntax

Option	Description
-h	Displays the UE Trace parser help text.
-s	Displays information about the UE Trace parser.
-f logfile	Specifies the UE Trace log file to decode If no UE Trace log file is specified, all log files in the current working directory, or the directory specified using option **-d** are decoded.
-d input-dir	Specifies the input directory where UE Trace log files are located. If no input directory is specified, the UE Trace log files in the current working directory are decoded.
-o output-dir	Specifies the output directory for the decoded files. If no output directory is specified, the output files are stored in the current working directory.
-i identity	Filter on IMSI.
-c interface	Filter on communication over an interface, such as S11 or S5.
-r node-type	Real-time print-out of ongoing UE Trace log with selected node type SGW, PGW, or EPG (both SGW and PGW). The options **-d**, **-f**, or **-o** cannot be used together with the option **-r**. The real-time print-out can be interrupted by pressing **Ctrl+C**.

```
user@host > epg_uetracer_parser.pl -f B20140207.135443+0100-SGW.epg128-6-sgw
processing: B20140207.135443+0100-SGW.epg128-6-sgw
result:     B20140207.135443+0100-SGW.epg128-6-sgw.par
```

Example 7 Parsing a UE Trace Log File.

```
user@host > epg_uetracer_parser.pl -f B20140207.135443+0100-SGW.epg128-6-sgw
processing: B20140207.135443+0100-SGW.epg128-6-sgw
result:     B20140207.135443+0100-SGW.epg128-6-sgw.par
```

Example 8 Filtering on IMSI in a UE Trace Log File.

```
Date: 2016-08-04  Time zone: UTC+02:00
04:46:46.175  IMSI   24001372293744    SGW           ->  GGSN/PGW      S5/S8
04:46:46.277  IMSI   24001372293744    GGSN/PGW      ->  PCRF          Gx
04:46:46.297  IMSI   24001372293744    GGSN/PGW      <-  PCRF          Gx
04:46:46.297  IMSI   24001372293744    GGSN/PGW      ->  OCS           Gy
04:46:46.308  IMSI   24001372293744    GGSN/PGW      <-  OCS           Gy


04:46:46.311  IMSI   24001372293744    SGW           <-  GGSN/PGW      S5/S8
04:46:46.312  IMSI   24001372293744    GGSN/PGW      ->  PCRF          Gx
04:46:46.316  IMSI   24001372293744    GGSN/PGW      <-  PCRF          Gx
04:46:46.354  IMSI   24001372293744    SGW           <-  GGSN/PGW      S5/S8
04:46:46.364  IMSI   24001372293744    SGW           ->  GGSN/PGW      S5/S8
04:46:46.366  IMSI   24001372293744    GGSN/PGW      ->  PCRF          Gx
04:46:46.373  IMSI   24001372293744    GGSN/PGW      <-  PCRF          Gx
04:46:46.439  IMSI   24001372293744    GGSN/PGW      ->  PCRF          Gx
04:46:46.448  IMSI   24001372293744    GGSN/PGW      <-  PCRF          Gx
04:46:46.452  IMSI   24001372293744    GGSN/PGW      ->  PCRF          Gx
04:46:46.452  IMSI   24001372293744    GGSN/PGW      <-  PCRF          Gx
04:46:46.460  IMSI   24001372293744    GGSN/PGW      ->  3GPP AAA ServerS6b
04:46:46.468  IMSI   24001372293744    3GPP AAA Server->  GGSN/PGW       S6b
04:46:46.469  IMSI   24001372293744    GGSN/PGW      ->  OCS           Gy
04:46:46.470  IMSI   24001372293744    SGW           <-  GGSN/PGW      S5/S8
04:46:46.476  IMSI   24001372293744    SGW           ->  GGSN/PGW      S5/S8
04:46:46.484  IMSI   24001372293744    GGSN/PGW      <-  OCS           Gy
04:46:47.135  IMSI   24001372293744    SGW           ->  GGSN/PGW      S5/S8
04:46:47.135  IMSI   24001372293744    SGW           <-  GGSN/PGW      S5/S8
04:46:47.140  IMSI   24001372293744    SGW           ->  GGSN/PGW      S5/S8
04:46:47.141  IMSI   24001372293744    GGSN/PGW      ->  PCRF          Gx
04:46:47.148  IMSI   24001372293744    GGSN/PGW      <-  PCRF          Gx
04:46:47.158  IMSI   24001372293744    GGSN/PGW      ->  OCS           Gy
04:46:47.158  IMSI   24001372293744    GGSN/PGW      ->  3GPP AAA ServerS6b
04:46:47.176  IMSI   24001372293744    GGSN/PGW      <-  OCS           Gy
04:46:47.177  IMSI   24001372293744    3GPP AAA Server->  GGSN/PGW       S6b
04:46:47.177  IMSI   24001372293744    SGW           <-  GGSN/PGW      S5/S8
04:46:47.232  IMSI   24001372293744    GGSN/PGW      <-  PCRF          Gx
04:46:47.233  IMSI   24001372293744    GGSN/PGW      ->  PCRF          Gx
04:46:50.741  IMSI   24001372293744    SGW           ->  GGSN/PGW      S5/S8
04:46:50.741  IMSI   24001372293744    GGSN/PGW      ->  PCRF          Gx
04:46:50.749  IMSI   24001372293744    GGSN/PGW      <-  PCRF          Gx
04:46:50.749  IMSI   24001372293744    GGSN/PGW      ->  OCS           Gy
04:46:50.760  IMSI   24001372293744    GGSN/PGW      <-  OCS           Gy
04:46:50.879  IMSI   24001372293744    SGW           <-  GGSN/PGW      S5/S8


```

Example 9 Printing Ongoing UE Trace Log.

3.9 EPG Healthcheck KPIs

The EPG Healthcheck KPIs script is used to show Key Performance Indicators (KPIs).

Note: The KPIs calculated by this tool do not correspond to the official EPG KPIs. The KPIs in Toolbox are to be used only for troubleshooting purposes.

Several sets of KPI values are displayed:

— Several most recent measurements (the default is three measurements, the range is from 1 to 39)

— One measurement from the previous day around the same time

— One measurement from the previous week around the same time

The KPI table consists of the following sections:

— PGW signalling KPIs

— PGW user plane KPIs

— SGW signalling KPIs

— SGW user plane KPIs

— CPU load

The user can select to see only certain sections. The following options are available:

Option	Description
1	PGW Signalling KPIs
2	PGW User plane KPIs
3	SGW Signalling KPIs
4	SGW User plane KPIs
5	CPU load average / peak

```
epg_healthcheck_kpi -n 10 4
```

Example 10 Display SGW User Plane KPIs

```
epg_healthcheck_kpi
```

Example 11 Display All KPIs

```
epg_healthcheck_kpi 5
```

Example 12 Display CPU Load Average / Peak

3.9.1 KPIs

The health check statistics table is calculated as described in the following sections.

3.9.1.1 PGW

— PgwGgsnSubscribers

  Based on the command `epg pgw statistics of brief`

  `statistics:subscriber-count`

— GgsnPdpContexts

  Based on the command `epg pgw statistics of brief`

  `pdp-context-statistics:pdp-active`

— PgwBearers

  Based on the command `epg pgw statistics of brief`

  `pdp-context-statistics:eps-active-bearer`

— PgwNonIpBearers

  Based on the command `epg pgw statistics of brief`

  `pdp-context-statistics:eps-active-non-ip-bearer`

— PgwNbIotBearers

  Based on the command `epg pgw statistics of brief`

— GgsnCreatePdpCtxFR [%]

  Based the on command `epg pgw statistics of brief`

  Failure ratio KPI:
  (pdp-context-statistics:pdp-create-attempted - pdp-context-statistics:pdp-created) / pdp-context-statistics:pdp-create-attempted * 100

— GgsnUpdatePdpCtxFR [%]

  Based on the command `epg pgw statistics of brief`

  Failure ratio KPI:
  (pdp-context-statistics:pdp-update-attempted - pdp-context-statistics:pdp-updates) / pdp-context-statistics:pdp-update-attempted * 100

— PgwS5CreateSessionFR [%]

Based on the command `epg pgw statistics of brief`

Failure ratio KPI:
(pdp-context-statistics:eps-session-creation-attempted - pdp-context-statistics:eps-session-creation) / pdp-context-statistics:eps-session-creation-attempted * 100

— PgwS5ModifyBearerFR [%]

Based on the command `epg pgw statistics of brief`

Failure ratio KPI:
(pdp-context-statistics:eps-bearer-modification-attempted - pdp-context-statistics:eps-bearer-modification) / pdp-context-statistics:eps-bearer-modification-attempted * 100

3.9.1.2 PGW User Plane

— PgwGgsnDlThroughput [Gbps]

Based on the command `epg pgw statistics of brief`

downlink:bytes * 8 / (interval_length * 1000 * 1000 * 1000)

— PgwGgsnUlThroughput [Gbps]

Based on the command `epg pgw statistics of brief`

uplink:bytes * 8 / (interval_length * 1000 * 1000 * 1000)

— PgwNonIpDlPacketThroughput [kpps]

Based on the command `epg pgw statistics of brief`

downlink:packets-non-ip / (interval_length * 1000)

— PgwNonIpUlPacketThroughput [kpps]

Based on the command `epg pgw statistics of brief`

uplink:packets-non-ip / (interval_length * 1000)

— PgwGgsnDlPacketThroughput [kpps]

Based on the command `epg pgw statistics of brief`

downlink:packets / (interval_length * 1000)

— PgwGgsnUlPacketThroughput [kpps]

Based on the command `epg pgw statistics of brief`

uplink:packets / (interval_length * 1000)

— PgwNonIpDlThroughput [Gbps]

Based on the command `epg pgw statistics of brief`

downlink:bytes-non-ip * 8 / (interval_length * 1000 * 1000 * 1000)

— PgwNonIpUlThroughput [Gbps]

Based on the command `epg pgw statistics of brief`

uplink:bytes-non-ip * 8 / (interval_length * 1000 * 1000 * 1000)

— PgwGgsnDlDataDropR [ppm]

Based on the command `epg pgw statistics of brief`

(downlink:dropped-packets * 1000 * 1000) / (downlink:dropped-packets + downlink:packets)

— PgwGgsnUlDataDropR [ppm]

Based on the command `epg pgw statistics of brief`

(uplink:dropped-packets * 1000 * 1000) / (uplink:dropped-packets + uplink:packets)

3.9.1.3 SGW

— SgwSubscribers

Based on the command `epg sgw statistics`

sgw-subscribers

— SgwNbIotSubscribers

Based on the command `epg sgw statistics`

sgw-nb-iot-subscribers

— SgwS11uSubscribers

Based on the command `epg sgw statistics`

sgw-s11u-subscribers

— SgwS11uNbIotSubscribers

Based on the command `epg sgw statistics`

sgw-s11u-nb-iot-subscribers

— SgwConnectedSubscribers

Based on the command `epg sgw statistics sgw-connected-subscribers`

- SgwNonIpSubscribers

  Based on the command `epg sgw statistics sgw-non-ip-subscribers`

- SgwNonIpPdnConnections

  Based on the command `epg sgw statistics sgw-non-ip-pdn-connections`

- SgwPdnConnections

  Based on the command `epg sgw statistics sgw-pdn-connections`

- SgwNbIotPdnConnections

  Based on the command `epg sgw statistics sgw-nb-iot-pdn-connections`

- SgwS11uPdnConnections

  Based on the command `epg sgw statistics sgw-nb-iot-pdn-connections`

- SgwS11uNbIotPdnConnections

  Based on the command `epg sgw statistics sgw-s11u-nb-iot-pdn-connections`

- SgwBearers

  Based on the command `epg sgw statistics sgw-bearers`

- SgwNbIotBearers

  Based on the command `epg sgw statistics sgw-nb-iot-bearers`

- SgwS11uBearers

Based on the command `epg sgw statistics`

sgw-s11u-bearers

— SgwS11uNbIotBearers

Based on the command `epg sgw statistics`

sgw-s11u-nb-iot-bearers

— SgwS4S11CreateSessionFR [%]

Based on the command `epg sgw statistics`

failure ratio kpi:
(sm-create-session-req-rcvd - sm-create-session-resp-acc-sent) /
sm-create-session-req-rcvd * 100

— SgwS4S11ModifyBearerFR [%]

Based on the command `epg sgw statistics`

Failure ratio KPI:
(sm-modify-bearer-req-rcvd - sm-modify-bearer-resp-acc-sent) /
sm-modify-bearer-req-rcvd * 100

— SgwS4S11DlDataNotificationFR [%]

Based on the command `epg sgw statistics`

Failure ratio KPI:
sgw-dl-data-notification-failure-ind-received / sgw-dl-data-notification-sent
* 100

3.9.1.4 SGW User plane

```
================SGW User plane Key Performance Indicators (KPIs)
The counters used to calculate the following KPIs are collected with the CLI co
epg sgw statistics
For each KPI, the formulas presented below are based on:
     PM-xml log counter names
     Statistics printout counter names
```

— SgwDlThroughputS1uS4S12 [Gbps]

(sgw-gtp-traffic-s1us4s12:out-data-byte * 8) / (interval_length * 1000 *
1000 * 1000)

— SgwUlThroughputS5S8 [Gbps]

(sgw-gtp-traffic-s5s8:out-data-byte * 8) / (interval_length * 1000 * 1000
* 1000)

— SgwUlThroughputS5 [Gbps]

  (sgw-gtp-traffic-s5:out-data-byte * 8) / (interval_length * 1000 * 1000 * 1000)

— SgwUlThroughputS8 [Gbps]

  (sgw-gtp-traffic-s8:out-data-byte * 8) / (interval_length * 1000 * 1000 * 1000)

— SgwDlPktThroughputS1uS4S12 [kpps]

  sgw-gtp-traffic-s1us4s12:out-data-pkt / (interval_length * 1000)

— SgwUlPktThroughputS5S8 [kpps]

  sgw-gtp-traffic-s5s8:out-data-pkt / (interval_length * 1000)

— SgwUlPktThroughputS5 [kpps]

  sgw-gtp-traffic-s5:out-data-pkt / (interval_length * 1000)

— SgwUlPktThroughputS8 [kpps]

  sgw-gtp-traffic-s8:out-data-pkt / (interval_length * 1000)

— SgwUlPktDropRS1uS4S12 [ppm]

  (sgw-gtp-traffic-s1us4s12:dropped-in-data-pkt * 1000000) / (sgw-gtp-traffic-s1us4s12:dropped-in-data-pkt + sgw-gtp-traffic-s1us4s12:in-data-pkt)

— SgwUlPktDropRS5S8 [ppm]

  (sgw-gtp-traffic-s5s8:dropped-out-data-pkt * 1000000) / (sgw-gtp-traffic-s5s8:dropped-out-data-pkt + sgw-gtp-traffic-s5s8:out-data-pkt)

— SgwUlPktDropRS5 [ppm]

  (sgw-gtp-traffic-s5:dropped-out-data-pkt * 1000000) / (sgw-gtp-traffic-s5:dropped-out-data-pkt + sgw-gtp-traffic-s5:out-data-pkt)

— SgwUlPktDropRS8 [ppm]

  (sgw-gtp-traffic-s8:dropped-out-data-pkt * 1000000) / (sgw-gtp-traffic-s8:dropped-out-data-pkt + sgw-gtp-traffic-s8:out-data-pkt)

— SgwDlPktDropRS1uS4S12 [ppm]

  (sgw-gtp-traffic-s1us4s12:dropped-out-data-pkt * 1000000) / (sgw-gtp-traffic-s1us4s12:dropped-out-data-pkt + sgw-gtp-traffic-s1us4s12:out-data-pkt)

— SgwDlPktDropRS5S8 [ppm]

(sgw-gtp-traffic-s5s8:dropped-in-data-pkt * 1000000) / (sgw-gtp-traffic-s5s8:dropped-in-data-pkt + sgw-gtp-traffic-s5s8:in-data-pkt)

— SgwDlPktDropRS5 [ppm]

(sgw-gtp-traffic-s5:dropped-in-data-pkt * 1000000) / (sgw-gtp-traffic-s5:dropped-in-data-pkt + sgw-gtp-traffic-s5:in-data-pkt)

— SgwDlPktDropRS8 [ppm]

(sgw-gtp-traffic-s8:dropped-in-data-pkt * 1000000) / (sgw-gtp-traffic-s8:dropped-in-data-pkt + sgw-gtp-traffic-s8:in-data-pkt)

— SgwDlDataDropR [ppm] (total DL drop rate)

(sgw-downlink-traffic:sgw-downlink-dropped-packets * 1000000) / (sgw-gtp-traffic-s5s8:in-data-pkt + sgw-gtp-traffic-s5:in-data-pkt + sgw-gtp-traffic-s8:in-data-pkt + sgw-gtp-traffic-s5s8:dropped-in-data-pkt + sgw-gtp-traffic-s5:dropped-in-data-pkt + sgw-gtp-traffic-s8:dropped-in-data-pkt)

— SgwUlDataDropR [ppm] (total UL drop rate)

(sgw-uplink-traffic:sgw-uplink-dropped-packets * 1000000) / (sgw-gtp-traffic-s1us4s12:in-data-pkt + sgw-gtp-traffic-s1us4s12:dropped-in-data-pkt)

3.9.1.5 CPU Load

— CPU load

Peak CPU load and average CPU load are for each board.

- peakCpuUsage [%]

  Based on the command `epg node status`

  cpu-utilization: Peak

- averageCpuUsage [%]

  Based on the command `epg node status`

  cpu-utilization: Average

3.9.2 Usage

The syntax of the control command is as follows:

```
epg_healthcheck_kpi [-hs] [-n NUM] [-d dir] [-o "yyyy-mm-dd
HH:MM:SS"] [sections]
```

Usage:

```
epg_healthcheck_kpi -h | -s
epg_healthcheck_kpi [-n NUM] [sections]
```

The command is used to display healthcheck related KPIs.

The following options are available:

Option	Description
`-h`	Prints the tool help text.
`-s`	Print a summary line. If both `-h` and `-s` are entered, `-h` overrides `-s`.
`-n NUM`	The number of the latest PDC measurement samples to show (the default is 3, range: 1–39).
`-d dir`	The directory where to read the KPI data from (offline/debug usage).
`-o "yyyy-mm-dd HH:MM:SS"`	Specifies the starting date and time of the displayed KPI data. The quotation marks are mandatory due to the space between the date and the time options.
`sections`	A comma-separated list of numbers with the sections to show, for example 1,2. If `sections` is omitted, all sections are displayed.

Comments:

— In data communications the metric definition of 1 kilobit = 1,000 bits is used. The binary definition of a kilobyte = 1,024 bytes is used in areas such as data storage, but not for expressing bandwidth and throughput.

— The `interval_length` is measured in seconds.

— (-) indicates that a counter is not available, or counter wrapped, or Failure Ratio denominator equals 0.

```
epg_healthcheck_kpi 4
```

Example 13 Display SGW User plane KPIs

```
epg_healthcheck_kpi
```

Example 14 Display All KPIs

```
4================SGW User plane Key Performance Indicators (KPIs)  [epg2-3]
                                                                              \
                                  2014-01-22     2014-01-22     2014-01-22     -
                                  18:03          17:48          17:33          -
SgwDlThroughputS1uS4S12 [Gbps]    1.0667         1.0667         1.0667         -
SgwUlThroughputS5S8 [Gbps]        1.0667         1.0667         1.0667         -
SgwUlThroughputS5 [Gbps]          1.0667         1.0667         1.0667         -
SgwUlThroughputS8 [Gbps]          1.0667         1.0667         1.0667         -
SgwDlPktThroughputS1uS4S12 [kpps]11.11           11.11          11.11          -
SgwUlPktThroughputS5S8 [kpps]     11.11          11.11          11.11          -
SgwUlPktThroughputS5 [kpps]       11.11          11.11          11.11          -
SgwUlPktThroughputS8 [kpps]       11.11          11.11          11.11          -
SgwUlPktDropRS1uS4S12 [ppm]       79.99          79.99          79.99          -
SgwUlPktDropRS5S8 [ppm]           79.99          79.99          79.99          -
SgwUlPktDropRS5 [ppm]             79.99          79.99          79.99          -
SgwUlPktDropRS8 [ppm]             79.99          79.99          79.99          -
SgwDlPktDropRS1uS4S12 [ppm]       79.99          79.99          79.99          -
SgwDlPktDropRS5S8 [ppm]           79.99          79.99          79.99          -
SgwDlPktDropRS5 [ppm]             79.99          79.99          79.99          -
SgwDlPktDropRS8 [ppm]             79.99          79.99          79.99          -
SgwDlDataDropR [ppm]              333.20         333.20         333.20         -
SgwUlDataDropR [ppm]              998.92         998.92         998.92         -
```

Example 15 Output from epg_healthcheck_kpi 4

3.9.3 Offline Usage

Offline usage of the KPI script included in the support package requires a workstation that has Perl™ installed.

The KPI script uses the `/usr/bin/perl` path. If the workstation does not have Perl™ installed in that path, run the script with the locally installed path.

To use the KPI script in the support package, do the following:

1. Unpack the support package.

2. Enter the following command:
   `cd node-name` (support pack directory)

The following steps assume that the user performing the steps is in the `node-name` directory which is the root of the support package.

The full node KPI data available in the support package is available in the following directories:

— Historic data in the directory `var/log/services/epg/pdc/kpi` (updated at monthly pdc pack and at `epg_pdc_control stop`)

— Current data in the current pdc pack

Optional: To make all the KPI data from the support package available for offline analysis, the historic and current data can be merged. To merge the historic and the current data, do the following:

1. Unpack the current pdc pack by using the following command:

   ```
   tar zxf pdc/current/<timestamp>_ggsn_pdc.tar.gz -C pdc/current
   ```

2. Merge the files with KPI data from pdc with the historic KPI data by using the following command:

   ```
   cat pdc/current/extra/pdc_pg >> var/log/services/epg/pdc/kpi/pdc_pgw_stat
   cat pdc/current/extra/pdc_pgw_statistics.csv >> var/log/services/epg/pdc/
   cat pdc/current/extra/pdc_sgw_statistics.csv >> var/log/services/epg/pdc/
   ```

The KPI data is now merged and is included in the `var/log/services/epg/pdc/kpi` directory.

If the KPI data is not merged, the directory `var/log/services/epg/pdc/kpi` includes historic data and the directory `pdc/current/extra` includes the current data.

The following example shows the commands to display the KPI data offline:

— **`scripts/epg_healthcheck_kpi -h`** (KPI script help)

— **`scripts/epg_healthcheck_kpi -d var/log/services/epg/pdc/kpi -o "yyyy-mm-dd HH:MM:SS"`** (quotes are mandatory for the `-o` option)
`-o "date time"` is used for start of the KPI data display, for example displaying data from the time when a certain problem occurred

The following example shows the commands to display the KPI data offline if the KPI data is not merged:

— **`scripts/epg_healthcheck_kpi -d var/log/services/epg/pdc/kpi -o "yyyy-mm-dd HH:MM:SS"`** (KPI from historic data)

— **`scripts/epg_healthcheck_kpi -d pdc/current/extra -o "yyyy-mm-dd HH:MM:SS"`** (KPI from current data)

3.10 EPG Delta Statistics

EPG Delta Statistics is a tool that enables the user to get delta statistics between two ECCLI printout samples.

Note: EPG Delta Statistics cannot be used when user login authorization is performed remotely through TACACS+.

The output is similar to the original printout of the ECCLI command and includes the current value of each counter or gauge, followed by the delta value within parenthesis (). The `-persec` option adds a per second value calculated from delta or delta-seconds.

The syntax for the command is: `epg_delta_statistics "cli-cmd-string"`

Sample 1: `epg_delta_statistics "cli-cmd-string"`
Sample 2: `epg_delta_statistics "cli-cmd-string"`

The output displays all numbers that have been modified between sample 1 and sample 2. `epg_delta_statistics` works for any ECCLI command that has numbers as output. The same command must be executed twice to get the delta.

```
"Create Session Request Received: 3338 (51) (1.31/s)"
Where:
3338     is the current counter value when the second command was executed
(51)     is the delta value
(1.31/s) is the delta value per-second (delta value divided by delta-seconds)
```

Example 16 Detailed Description of the Output from a Line, Executed with a Delta of 30 seconds

Comments on using the command:

— The delta value is calculated on any number on a line from the printout, where a number is represented by its own or followed by a `,` (comma) or a `;` (semi-colon). If the number is mixed with any other characters, the number is ignored during the delta calculation.

— Lines including a time syntax, such as `hh:mm:ss`, are ignored even if other numbers following the initial rule exist.

— Lines including an IP address syntax, such as `nnn.nnn.nnn.nnn` or `::`, are ignored.

— Ignored lines are printed unchanged.

— Lines including a delta value are prepended by a + (plus) sign.

— The delta value is calculated between the two last executions of a command.

— The command has the following options for executing in different modes:

  - By default, the command is executed in the operational mode. For an example of executing the command in the operational mode, see Example 17.
  - To execute the command in the configuration mode, add `-c` before `"cli-command-string"`, see Example 18.
  - To execute the command in the `platform exec mode`, add `-e` before `"cli-command-string"`, see Example 19.

— The data from the delta calculations is temporarily stored in a `/HOME-dir/epg_delta_statistics_tmp/` directory.

  To delete old sample files, use the following command:

```
bash-3.2$ epg_delta_statistics -C
```

Subdirectories might need to be removed manually.

— The script does not handle column widths. Therefore lines including several columns of numbers might not look exactly as the original printout.

— Most numeric printouts are calculated for a delta value, meaning that version numbers, interface IDs, and other static numeric information might also be printed as delta values. This is by design.

```
bash-3.2$ epg_delta_statistics "epg sgw statistics"
```

Example 17 Executing in operational mode

```
bash-3.2$ epg_delta_statistics -c "epg node internal-debug status"
```

Example 18 Executing in configuration mode

```
bash-3.2$ epg_delta_statistics -e "show ip statistics"
```

Example 19 Executing in platform exec mode

Usage:

```
epg_delta_statistics [-h|-s|-c] [-i sec] [-persec] [-e]
"cli-cmd-string"
```

The following options are available:

Option	Description
`-h`	Prints the tool help text.
`-s`	Prints a short summary description.
`-c`	Used to execute a command in configuration mode.
`-e`	Used to execute a command in platform exec mode.
`-i <sec>`	Interval mode, executes the command twice with a period given by `sec` in between.
`-persec`	Adds per-second values to the delta printout.
`-C`	Clears the temp directory (/usr-HOME-dir/epg_delta_statistics_tmp/) from old samples.
`"cli-cmd-string"`	Used to add a command string between quotation marks.

```
bash-3.2$ epg_delta_statistics "epg pgw statistics of brief"
== DELTA ========================================
CLI-command:  epg pgw statistics of brief
delta-seconds: 23 sec.
=================================================
    epg pgw statistics of brief
  ggsn-statistics:
      time-started: Thu May  3 14:09:16 2018
      time-sampled: Thu May 31 17:59:23 2018

      pdp-context-statistics:
+         eps-active-emergency-unath-ues: 0 (0)
+         pdp-active: 0 (0)
+         eps-active-bearer: 0 (0)
+         eps-active-bearer-wlan: 0 (0)
+         pdp-active-ipv6: 0 (0)
+         pdp-active-ipv4v6: 0 (0)
+         eps-active-ipv6-bearer: 0 (0)
+         eps-active-ipv4v6-bearer: 0 (0)
+         eps-active-qci1-bearer: 0 (0)
+         eps-active-bearer-nb-iot: 0 (0)
+         pdp-create-attempted: 0 (0)
+         pdp-created: 0 (0)
+         pdp-create-failed: 0 (0)
+         pdp-created-ipv6: 0 (0)
+         pdp-create-failed-ipv6: 0 (0)
+         pdp-secondary-created: 0 (0)
+         pdp-secondary-create-failed: 0 (0)
+         pdp-secondary-created-ipv6: 0 (0)
+         pdp-secondary-attempted-ipv6: 0 (0)
```

Example 20 The First Part of the Output of a Delta Statistics of 31 seconds between Two Executions of the Same Command

3.11 EPG Software Version

The command **`show swim`** displays the EPG version installed on both partitions. This command is used during the Software and Update procedure.

Executing the command provides complete information about the installed EPG software version, including OS software version and details of the software build.

```
swim
 active [ EPG_ACTIVE_R1A111 ]
 sw-version EPG_ACTIVE_R1A111
  administrative-data
   product-name     EPG
   product-number   "CXP 902 7374/27"
   product-revision R1A111
   production-date  2018-11-08T18:08:13+00:00
   description      "Evolved Packet Gateway 2.0"
   type             Release
  time-of-installation 2018-11-09T06:57:41+00:00
 sw-version EPG_ALTERNATE-2_R1A02
  administrative-data
   product-name     EPG
   product-number   "CXP 902 7374/27"
   product-revision R1A02
   production-date  2018-11-02T09:57:42+00:00
   description      "Evolved Packet Gateway 2.0"
   type             Release
  time-of-installation 2018-11-08T12:54:25+00:00
```

Example 21 Software Version for EPG

3.12 Loggd Joiner

The **loggd_joiner.py** tool is used to improve the log readability by joining log rows that are split in two or more rows in the log file. The tool is intended for the system log, `loggd_persistent.log`.

The syntax **loggd_joiner.py** is as follows:

```
loggd_joiner.py [-h] [-s] [-f FILE[FILE ...]]  [-o OUT][--single]
```

Option	Description
-h **--help**	Displays the tool help.
-s	Displays a short summary.
-f FILE[FILE...] **--file FILE[FILE...]**	Specifies the input file(s) for processing. The input filename may include a wildcard (*).

Option	Description
-o OUT **--out OUT**	Specifies the output directory name. If nothing is specified, the current directory is used as default directory. If the specified directory does not exist, an error message is displayed.
--single	Compiles all input files into one single output file in the current directory. If this option is used together with the **-o OUT** option, the single output file is stored in the directory specified in **OUT**.

The resulting output filename is the same as the input filename with the extension `.joined`. If the option `--single` is used, the output filename is the first file specified as input file, with the extension `.single.joined`.

When executing, `[RUNNING]` is printed together with filename-specific information.

The following example shows using the **loggd_joiner.py** tool to convert the file `infile.log` and save the output in the directory `Temp`:

```
> loggd_joiner.py -f infile.log -o Temp
```

The **loggd_joiner.py** tool is also provided as a part of the remote Toolbox tarball. This package must be fetched from the EPG and unpacked on a remote workstation which has Python™ installed. The Python™ version supported is the version included on the node. To check which Python™ version the node supports, log into the node, start a shell, and type `python -V`. For more information on remote Toolbox, seeSection 2.1 on page 2.

3.13 PISC Security Log

The `show_pisc_security_log` tool is used to show the PISC security log content with optional filters of attack type and IMSI. The tool is intended for the PISC security log named as `user.log-<hostname>` or `daemon.log-<hostname>` in the RP folder `/var/log/services/epg/piscsecurity/`. For more information about the PISC security log, refer to EPG Logs.

The syntax of `show_pisc_security_log` is as follows:

```
show_pisc_security_log [-a AttackType] [-i IMSI]
```

Option	Description
-a AttackType	Specifies the attack type to filter the log content. Currently only TCP_SYN_FLOOD_FROM_UE is supported as the attack type.
-i IMSI	Specifies the IMSI to filter the log content.

Example 22 shows how to retrieve the PISC security log content for TCP SYN flood attacks from UEs.

```
bash-3.2$ show_pisc_security_log -a TCP_SYN_FLOOD_FROM_UE
/var/log/services/epg/piscsecurity/user.log-vsfo-2:Nov  8 12:04:51 vsfo-2 ep
/var/log/services/epg/piscsecurity/user.log-vsfo-2:Nov  8 12:04:53 vsfo-2 ep
```

Example 22 PISC Security Log Content Filtered by Attack Type

3.14 Flatten XML

The `flatten_xml.py` tool is used to convert XML files such as the `ericsson.xml` file into plain texts.

The usage of `flatten_xml.py` is as follows:

```
flatten_xml.py <file-name>
flatten_xml.py <file-name> | grep logical-interface
flatten_xml.py <file-name> | tee <flatten-text-file-name>
```

```
bash-4.2$ flatten_xml.py ericsson.xml
asp default_asp_attr default_asp_attr management/1
asp default_asp_attr shutdown false
asp default_asp_attr default_asp_attr 1/1
asp default_asp_attr shutdown false
```

Example 23 Converting the ericsson.xml File into Plain Texts