

PDB Command Line Interface (CLI) Reference

Parameter Database

DESCRIPTION

Copyright

© Ericsson AB 2012 – 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

Ericsson

Ericsson is the trademark or registered trademark of Telefonaktiebolaget LM Ericsson. All other product or service names mentioned in this manual are trademarks of their respective companies.



Contents

1	About This Guide	1
1.1	Intended Audience	1
1.2	Conventions Used in This Guide	1
1.3	Prerequisites	3
1.4	Comments About the Documentation	3
2	Overview	3
2.1	CLI Help	6
3	compare-configuration-file	6
3.1	Usage Syntax	7
3.2	Options	7
3.3	Examples	9
3.4	Automated Live Node Configuration Comparison	9
4	compare-local-configuration-files	13
4.1	Usage Syntax	14
4.2	Options	14
4.3	Examples	17
4.4	Working with a Comparison Options File	17
5	configuration-list	21
5.1	Usage Syntax	21
5.2	Options	22
5.3	Examples	23
6	configuration-revision-set-status	24
6.1	Usage Syntax	24
6.2	Options	24
6.3	Examples	25
7	configuration-revision-set-tag	25
7.1	Usage Syntax	26
7.2	Options	26
7.3	Examples	27



8	configuration-revision-unset-tag	27
8.1	Usage Syntax	28
8.2	Options	28
8.3	Examples	29
9	convert-configuration	30
9.1	Usage Syntax	30
9.2	Options	31
9.3	Examples	33
10	create-variables-file	34
10.1	Usage Syntax	35
10.2	Options	35
10.3	Examples	36
11	export-configuration	36
11.1	Usage Syntax	37
11.2	Options	37
11.3	Examples	39
12	import-configuration	40
12.1	Usage Syntax	40
12.2	Options	41
12.3	Examples	43
13	import-configuration-revision	43
13.1	Usage Syntax	43
13.2	Options	44
13.3	Examples	45
14	import-schema-revision	46
14.1	Usage Syntax	46
14.2	Options	46
14.3	Examples	48
15	node-create-revision	48
15.1	Usage Syntax	48
15.2	Options	49
15.3	Examples	49
16	node-revision-link-configuration-revision	50



16.1	Usage Syntax	50
16.2	Options	50
16.3	Examples	51
17	node-revision-link-schema-revision	51
17.1	Usage Syntax	52
17.2	Options	52
17.3	Examples	52
18	schema-compare	53
18.1	Usage Syntax	53
18.2	Options	53
18.3	Examples	55
19	schema-list	55
19.1	Usage Syntax	55
19.2	Options	56
19.3	Examples	57
20	schema-revision-set-ivl	58
20.1	Usage Syntax	58
20.2	Options	58
20.3	Examples	59
21	schema-revision-set-status	59
21.1	Usage Syntax	60
21.2	Options	60
21.3	Examples	61
22	schema-revision-unset-ivl	61
22.1	Usage Syntax	61
22.2	Options	61
22.3	Examples	62
23	ssl-export	62
23.1	Usage Syntax	64
23.2	Options	64
23.3	Examples	65
24	ssl-list	65
24.1	Usage Syntax	65



24.2	Options	66
24.3	Examples	66
25	ssl-list-variable	67
25.1	Usage Syntax	67
25.2	Options	67
25.3	Examples	68
26	ssl-set-variable	68
26.1	Usage Syntax	69
26.2	Options	69
26.3	Examples	70
27	ssl-unset-variable	70
27.1	Usage Syntax	70
27.2	Options	70
27.3	Examples	71
28	validate-local-configuration-files	71
28.1	Usage Syntax	72
28.2	Options	72
28.3	Examples	73
29	validate-mim	74
29.1	Usage Syntax	74
29.2	Options	74
29.3	Examples	76
30	validate-mpvl	76
30.1	Usage Syntax	76
30.2	Options	77
30.3	Examples	78
Reference List		79



1 About This Guide

This document describes the functionality offered by each of the PDB Command Line Interface (CLI) commands.

1.1 Intended Audience

This guide is intended for end-users who interact with PDB using the CLI.

Personnel working on Ericsson products or systems must have the training and competence required to perform their work correctly.

1.1.1 Prerequisite Knowledge

Users of this document should have knowledge and experience of the following:

- Node configuration files (LDIF, NETCONF, and so on)
- MIM files
- Linux

1.2 Conventions Used in This Guide

Table 1 provides a list of typographic conventions that may be encountered in this document:

Table 1 Typographic Conventions

Convention	Description	Example
Code Examples	Code examples	<code>stat char* months[] = {"Jan", "Feb"}</code>
Command Variables	You need to supply the values within the <>	<code><home_directory></code>
Document and File Names	References to document titles or sections in a document and file names	For more information, refer to the System Administrator Guide. Check the local runlog files (xxx.runlog and xxa.runlog) in the /var/log/xxx directory.



Convention	Description	Example
GUI Objects	GUI objects, such as menus, fields, and buttons, dialog boxes, and options	On the File menu, click Exit .
Key Combinations	Key combinations	Press Ctrl+X to delete the selected value. ⁽¹⁾
Output Information	Text displayed by the system	System awaiting input
Parameter/Configuration Values	Parameter values (numbers, true/false, yes/no, and so on)	To use this feature, the parameter must be set to true
System Elements	Command and parameter names, program names, path names, URLs, and directory names	The files are located in E:\Test The files are located in /etc/opt/ericsson/bin. ⁽²⁾
User Input	In this document when you are required to input content, the input content is displayed using this bold mono-spaced font. The content must be added exactly as shown.	cd \$HOME
Line Break	The arrow symbol (⇒) can be used when an inappropriate line break has been made. An inappropriate line break occurs when the code lines are too long to fit on the page, and there is no appropriate place for a line break.	cd /opt/msmw-cds-⇒ cxp-<version> ⁽³⁾

(1) The plus sign (+) indicates that you must press the keys simultaneously.

(2) The use of the forward slash (/) is for Linux and UNIX systems; Windows systems use the backslash (\).

(3) The use of the ⇒ symbol (character entity ⇒) at the end of a line has a meaning to the human reader, but if copied and pasted from a CLI document to a command line interpreter the symbol must be cut from the code.



1.3 Prerequisites

The PDB CLI client is installed and properly configured.

For installation procedures, refer to the [Parameter Database \(PDB\) Installation Instructions](#) (Reference [1]).

For configuration procedures, refer to the [PDB System Administration Guide](#) (Reference [2]).

1.4 Comments About the Documentation

Ericsson encourages you to provide feedback, comments, or suggestions so that we can improve the documentation to better meet your needs. With your comments, provide the following:

- Document title
- Document number and revision
- Page number

Please send your comments to your local Ericsson Support.

2 Overview

The PDB CLI is a collection of command line tools that allows you to make use of the PDB functionality without having to log into a PDB server.

The CLI client is available on Linux and Microsoft® Windows® platforms. For detailed instructions on how to install the PDB CLI, refer to the [Parameter Database \(PDB\) Installation Instructions](#) (Reference [1]).

The CLI client acts as a front end for the CLI commands. You can retrieve the CLI version with the following option:

```
pdbcli --version
```

The following commands are available for use:

Table 2

Command	Description
compare-configuration-file	Compares a configuration file (extracted from a node) against a reference configuration in PDB.



Table 2

<code>compare-local-configuration-files</code>	Compares two local configuration files.
<code>configuration-list</code>	Lists configurations stored in PDB.
<code>configuration-revision-set-status</code>	Sets the status of a configuration revision.
<code>configuration-revision-set-tag</code>	Applies one or more tags to a configuration revision stored in PDB.
<code>configuration-revision-unset-tag</code>	Removes one or more tags from a configuration revision stored in PDB.
<code>convert-configuration</code>	Converts a configuration between the supported formats.
<code>create-variables-file</code>	Generates a template file containing all the parameter variables required by a given configuration.
<code>export-configuration</code>	Exports a configuration from the PDB server.
<code>import-configuration</code>	Imports a new node configuration to the PDB database.
<code>import-configuration-revision</code>	Imports a new revision of an existing node configuration to the PDB database.
<code>import-schema-revision</code>	Imports a new revision of an existing configuration schema to the PDB database.
<code>node-create-revision</code>	Creates a new revision of a node defined in PDB.



Table 2

node-revision-link-configuration-revision	Links a configuration revision to a node revision.
node-revision-link-schema-revision	Links a schema revision to a node revision.
schema-compare	Compares two sets of schema files and provides a comparison report in Comma-Separated Values (CSV) format.
schema-list	Lists configuration schemas stored in PDB.
schema-revision-set-ivl	Associates an IVL with a configuration schema stored in PDB.
schema-revision-set-status	Sets the status of a schema revision.
schema-revision-unset-ivl	Removes an IVL association from a configuration schema stored in PDB.
ssl-export	Exports a Site-Specific List (SSL) from the PDB server to an output file.
ssl-list	Lists SSLs stored in PDB.
ssl-list-variable	Lists the variables in a specified SSL.
ssl-set-variable	Sets a new value for an existing SSL variable or creates a new SSL variable with the specified value.
ssl-unset-variable	Deletes a SSL variable.
validate-local-configuration-files	Validates local configuration files and generates a validation report. Validation considers standard file syntax, format rules, and the limits set by the selected MIM files, including schematron rules.



Table 2

<code>validate-mim</code>	Validates MIM files and other support files like <code>index</code> and <code>model</code> . The validation considers both standard XML syntax and format rules that are part of the official MIM format definitions.
<code>validate-mpvl</code>	Validates configuration files written in PVL format. The validation considers standard XML syntax, format rules established by the official definition of the PVL format, and the limits set by the selected MIM files.

Refer to the following sections for command and usage information.

2.1 CLI Help

All CLI commands have a help option that prints usage information.

To request help for a specific command, enter the following at the command line:

```
pdbcli [<command>] -h, --help
```

3 compare-configuration-file

`compare-configuration-file` compares a configuration file extracted from a node against a reference configuration defined in PDB. The command outputs a comparison report in CSV format. If the configurations are identical, this report is empty and the exit status will be 0.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

PDB includes a UNIX script called `compare_live_node` that automatically extracts a live node configuration and compares it against a reference configuration defined in PDB. For more information on `compare_live_node`, refer to Section 3.4 on page 9.



3.1 Usage Syntax

```

compare-configuration-file -c configuration-file
compare-configuration-file -f format configuration-file
compare-configuration-file -n name configuration-file
compare-configuration-file -p password configuration-file
compare-configuration-file -r revision configuration-file
  
```

3.2 Options

Table 3 compare-configuration-file Options

Option	Description	Notes
-c, --configuration-files <arg>	Specifies the path and filename of a configuration file to use for the comparison.	Mandatory. Paths can be relative or absolute. This command supports the comparison of multiple LDIF files. Multiple files are specified using additional -c options.
-f, --format <arg>	Specifies the format of the node configuration file. Possible formats include: <ul style="list-style-type: none"> • cpp_csv • ldif • netconf • pvl 	Mandatory.
-n, --configuration-name <arg>	Specifies the name of a reference configuration in PDB.	Mandatory.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-r, --revision <arg>	Specifies a revision of the reference configuration. The command defaults to the latest revision if this option is not used.	Optional.



Table 3 compare-configuration-file Options

Option	Description	Notes
-R, --validation-report-file <arg>	Before PDB compares against an external configuration it first performs a validation check on that configuration. <code>compare-configuration-file</code> can store the results of the validation in a plain-text file. Specifies a path and filename for the validation report.	Optional. Paths can be relative or absolute.
-S, --report-file <arg>	Specifies a path and filename for the comparison report.	Mandatory. Paths can be relative or absolute. The comparison report is generated in CSV format.
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]) .
-v, --variables-file <arg>	Specifies the path and filename of a variables list that can be used to resolve any parameter value variables in the reference configuration before comparing.	Optional. Paths can be relative or absolute. The variables list is a user-defined CSV file that adheres to the following format, one variable per line: <code><name>,<value>,<description>,<usage></code> Use the <code>create-variables-file</code> CLI command to generate a file that you can populate with values.



3.3 Examples

Note: You must use double quotes <" "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
# Create a new variable called "total_bill" which is equal to the sum of "bill" and "tax"
total_bill = bill + tax

# Print out total_bill
print(total_bill)
```

This command compares the latest revision of the CSCF_11B_R4G configuration to the CSCF_11A_R4B.ldif file using the ss_list.lst variables file and output the results to Comparisons/ldif_cscf11a_r4b.csv.

3.4 Automated Live Node Configuration Comparison

Note: Automated live node configuration comparison is not offered through the PDB CLI.

PDB includes a UNIX script that can be used to automate the comparison of live node configurations in LDIF or NETCONF format. `compare_live_node` connects to a live node and automatically extracts the configuration data. Using the same logic as the `compare-configuration-file` CLI command, the script initiates a comparison of the live node configuration against a reference configuration defined in PDB and outputs a comparison report in CSV format. If the configurations are identical, this report is empty and the exit status will be 0.

`compare_live_node` requires connectivity with the designated live node as well as the PDB server. Live node connectivity is configured using a site-specific variables file that includes the destination IP address, port number, and login credentials. Use the `create-variables-file` CLI command to generate a file that you can populate with the necessary values.

Extracted node configurations are stored locally as follows:

- LDIF files are saved to the <pdftools>/cli directory with the following naming convention:

config_from_node_<timestamp>.ldif
- NETCONF files are saved to the <pdftools>/configtool directory with the following naming convention:

config from node <timestamp>.xml

`compare_live_node` uses external scripts to extract live node configurations.

LDAP nodes are processed using a set of Parameter Comparison Application (PCA) extractor scripts that must be installed separately. For more information on



installing the PCA extractors, refer to the [Parameter Database \(PDB\) Installation Instructions \(Reference \[1\]\)](#).

NETCONF nodes are processed using the PDB configtool and no further installation is necessary.

3.4.1 Usage Syntax

```
compare_live_node -a <node_application> -f <format> -n <configuration_name> -p <password> -r <revision>
```

3.4.2 Options

Table 4 compare_live_node Options

Option	Description	Notes
-a <node_application>	Specifies the node application name as defined in PDB and the site-specific variables file.	Mandatory. PDB prefixes LDAP and NETCONF variables with the associated application name. This name is necessary for the script to parse incoming variables.
-f <format>	Specifies the configuration format of the live node. Possible values include: <ul style="list-style-type: none">• Idif• netconf	Mandatory.
-n <configuration_name>	Specifies the name of a reference configuration in PDB.	Mandatory.
-p <password>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-r <revision>	Specifies a revision of the reference configuration. The command defaults to the latest revision if this option is not used.	Optional.



Table 4 compare_live_node Options

Option	Description	Notes
-R <report_file>	Specifies a path and filename for the comparison report.	Mandatory. Paths can be relative or absolute. The comparison report is generated in CSV format.
-s <script_application>	Specifies a script application to use for extracting the live configuration from an LDAP node. Possible values include: <ul style="list-style-type: none"> • cscf • hss • hss_slf • mrfc • mtas • pcrf • pm • slf 	Mandatory for the LDIF configuration format. The extractor scripts must be installed separately. For more information on installing the PCA extractors, refer to the Parameter Database (PDB) Installation Instructions (Reference [1]) .
-t <type>	Specifies the platform type of the live node. Possible values include: <ul style="list-style-type: none"> • is • omp 	Mandatory for the NETCONF configuration format.



Table 4 compare_live_node Options

Option	Description	Notes
-u <user>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).
-v <variables_file>	Specifies the path and filename of a variables list. compare_live_node uses the site-specific information contained in this file to resolve the destination IP address, port number, and login credentials for the live node as well as any site-specific variables used in the reference configuration before comparing.	Mandatory. Paths can be relative or absolute. The variables list is a user-defined CSV file that adheres to the following format, one variable per line: <name>, <value>, <description>, <usage> Use the <code>create-variables-file</code> CLI command to generate a file that you can populate with values.

3.4.3 Examples

Note: You must use double quotes ">" to surround entries that have spaces.

Sample Usage on a Linux Platform

```
compare_live_node -u admin -v ./cscf_comparison.csv
```

Compares the latest revision of the CSCF_11B_R4G configuration to the CSCF live node defined in `ldif.properties` and will output the results to `./cscf_comparison.csv`.

```
compare_live_node -u admin -v ./cscf_comparison.csv
```



Compares the PA3 revision of the SBG_R1A configuration to the SBG live node defined in netconf.properties and will output the results to ./sbg_comparison.csv.

4 compare-local-configuration-files

`compare-local-configuration-files` compares two local configuration files to identify the differences between them. The command outputs a comparison report in CSV format. If the configurations are identical, this report is empty and the exit status will be 0.

When performing a comparison, this command distinguishes between an Original (Left) Configuration and a Target (Right) Configuration. The Original Configuration serves as a starting point for the comparison and is displayed on the left side of the comparison report. Data from the Original Configuration is compared against a Target Configuration that is displayed on the right side of the comparison report.

`compare-local-configuration-files` allows you to customize the comparison report with options specified in an input file (yaml format). Customization options include:

- Specifying Site Specific Lists
- Falling Back to Default Values
- Ignoring Letter Case
- Ignoring ReadOnly Attributes
- Filtering by Category
- Filtering by Managed Object Class (MOC)
- Setting the Comparison Direction

A template file that specifies the default comparison options (`comparison-options.yaml`) is provided with the PDB CLI. For more information on working with a comparison options file, refer to Section 4.4 on page 17.



4.1 Usage Syntax

```
python pdbcli.py compare-local-configuration-files -l /etc/passwd
python pdbcli.py compare-local-configuration-files -L /etc/passwd
python pdbcli.py compare-local-configuration-files -l /etc/passwd -L /etc/passwd
python pdbcli.py compare-local-configuration-files -l /etc/passwd -L /etc/passwd -f netconf
```

4.2 Options

Table 5 compare-local-configuration-files Options

Option	Description	Notes
-l, --left-configuration- files <arg>	Specifies a single configuration file or a directory containing configuration files to serve as the Original configuration on the left side of the comparison report.	Mandatory. Paths can be relative or absolute. This option accepts only decompressed configuration files or a directory of decompressed files that constitute a single configuration. If selected, the directory is searched recursively for configuration files that may be present in subdirectories. Multiple files can be specified using additional -l options.
-L, --left-configuration- format <arg>	Specifies the format of the left configuration file. Possible formats include: <ul style="list-style-type: none">• cpp_csv• ldif• netconf• pvl	Mandatory.



Table 5 compare-local-configuration-files Options

Option	Description	Notes
-M, --moi-format <arg>	Specifies the MOI format of the PVL configuration files. Possible formats include: <ul style="list-style-type: none"> • tsp • is • cba 	Mandatory for PVL input files only. This selection applies to both the left and the right configuration. For more information on the PVL MOI format, refer to the Parameter List Template Description, EAB/FTI-08:0686 Uen.
-o, --comparison-options-file <arg>	Specifies a comparison options file (yaml format) to customize the comparison report. For more information on working with a comparison options file, refer to Section 4.4 on page 17.	Optional. Paths can be relative or absolute.
-r, --right-configuration-files <arg>	Specifies a single configuration file or a directory containing configuration files to serve as the Target configuration on the right side of the comparison report.	Mandatory. Paths can be relative or absolute. This option accepts only decompressed configuration files or a directory of decompressed files that constitute a single configuration. If selected, the directory is searched recursively for configuration files that may be present in subdirectories. Multiple files can be specified using additional -r options.
-R, --right-configuration-format <arg>	Specifies the format of the right configuration file. Possible formats include: <ul style="list-style-type: none"> • cpp_csv • ldif • netconf • pvl 	Mandatory.



Table 5 compare-local-configuration-files Options

Option	Description	Notes
-s, --schema-files <arg>	Specifies a single MIM file or a directory containing MIM files to use as a schema for the selected configurations.	Mandatory. Paths can be relative or absolute. This option accepts only decompressed MIM files or a directory of decompressed files that constitute a single schema. If selected, the directory is searched recursively for schema files that may be present in subdirectories. Multiple files can be specified using additional -s options.
-S, --comparison-report-file <arg>	Specifies a path and filename for the comparison report. The output defaults to stdout if this option is not set.	Optional Paths can be relative or absolute. The comparison report is generated in CSV format.
-T, --solution-type <arg>	Specifies which PVL parameter value set to compare. Possible solution types include: <ul style="list-style-type: none">• FIXED• MOBILE• CONVERGED• STANDALONE• INITIAL_VALUE	Mandatory for PVL input files only.
-V, --validation-report-file <arg>	Before comparing external configuration files, a validation check is performed on the configuration data. <code>compare-local-configuration-files</code> can store the results of the validation in a plain-text file. Specifies a path and filename for the validation report.	Optional. Paths can be relative or absolute.



4.3 Examples

Note: You must use double quotes <" "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
compare-local-configuration-files -f /path/to/tsp_format.xml  
-c /path/to/config1.xml -c /path/to/config2.xml  
-o /path/to/output.csv
```

This command compares the FIXED parameter value set of two PVL configuration files using `tsp_format.xml` as the schema. The comparison report will be output to `../reports/compare.csv`.

4.4 Working with a Comparison Options File

`compare-local-configuration-files` accepts an input file that allows you to customize the comparison report. A template file that specifies the default comparison options (`comparison-options.yaml`) is provided with the PDB CLI. Example 1 shows the contents of `comparison-options.yaml`.



```
# This file is to be used in conjunction with the following command: compare-local-configuration-files
# The following options are specific to each configuration under comparison
#
# site_specific_list:
#
# Site specific lists are a collection of key/value pairs used to support variable definition in
# configuration files. They are used to resolve any site-specific variables in the configuration.
# Enter the filename, absolute or relative path to the .CSV file. Leave empty if not used.
#
# fallback_to_default_values:
#
# If a configuration element exist in one configuration but is absent on the other, this
# option attempts a match by falling back to the default value for the absent element, if available.
# Available options are: false, true. Default is 'false'.

left_configuration:
  site_specific_list:
  fallback_to_default_values: false

right_configuration:
  site_specific_list:
  fallback_to_default_values: false

# The following configuration applies to both left and right configuration
common_options:
  # Consider the letter case when performing the comparison.
  # Available options are: false, true. Default is 'false'.
  case_insensitive: false

  # Consider read-only parameters when performing the comparison.
  # Available options are: false, true. Default is 'false'.
  ignore_read_only: false

  # Narrow down the comparison results by filtering based on parameter category.
  # Available categories are: [Internal, Operator Configurable, Site Specific, Solution Integration].
  # Note that 'include' acts as a white-list and 'exclude' as a black-list.
  # The value is an array. Leave it as '[]' if not used.
  filter_by_category:
    include: []
    exclude: []

  # Narrow down the comparison results by filtering based on MOC (managed object class) name.
  # Note that 'include' acts as a white-list and 'exclude' as a black-list.
  # The value is an array. Example: [SwInventory, SwM, Fm, Pm, ExtNetSelPoolTableEntryClass]
  # Leave it as '[]' if not used.
  filter_by_moc_name:
    include: []
    exclude: []

  # Specify in which direction the comparison is performed. Available options are:
  # . both_sides - This is the default behavior. PDB examines differences in both configurations
  # . left_to_right - PDB examines only what exist on the LEFT and is different on the RIGHT
  # . right_to_left - PDB examines only what exist on the RIGHT and is different on the LEFT
  comparison_direction: both_sides
```

Example 1 comparison-options.yaml

Table 6 describes the comparison options in comparison-options.yaml.



Table 6 comparison-options.yaml options

Option	Description	Notes
site_specific_list	Specifies the path and filename of a variables list that can be used to resolve any parameter value variables in the Original (Left) or Target (Right) configurations before comparing.	Optional. Paths can be relative or absolute. The variables list is a user-defined CSV file that adheres to the following format, one variable per line: <name>, <value>, <description>, <usage> Use the create-variables-file CLI command to generate a file that you can populate with values.
fallback_to_default_values	true/false When a configuration element is defined in one configuration but is absent from the other, this option attempts a match by falling back to the default value for the absent element, if available.	Optional. Default: false.
case_insensitive	true/false Ignore the letter case of parameter values.	Optional. Default: false.
ignore_read_only	true/false Ignore differences in ReadOnly parameters.	Optional. Default: false.
filter_by_category	Filters the comparison results to include or exclude specific categories of parameters and parameter groups. Available categories include: <ul style="list-style-type: none"> • Internal • Operator Configurable • Site Specific • Solution Integration 	Optional. Specified as an array of comma-separated values. Leave it as '[]' if not used.



Table 6 comparison-options.yaml options

Option	Description	Notes
filter_by_moc_name	Filters the comparison results to include or exclude specific Managed Object Classes.	Optional. Specified as an array of comma-separated values. Leave it as '[]' if not used.
comparison_direction	<p>Configures how the comparison is performed.</p> <ul style="list-style-type: none">• both_sides - Enabled by default, PDB examines all of the parameters and parameter groups within both configurations and identifies all differences in the comparison report.• left_to_right - PDB examines all of the parameters and parameter groups within the Original Configuration and compares them against the Target Configuration. Only those differences unique to the Original Configuration are identified in the comparison report.• right_to_left - PDB examines all of the parameters and parameter groups within the Target Configuration and compares them against the Original Configuration. Only those differences unique to the Target Configuration are identified in the comparison report.	Optional. Default: both_sides.

To customize the comparison options:

1. Locate the comparison-options.yaml template file in the <pdbtools>/cli directory.
2. Make a copy of comparison-options.yaml for editing.
3. Update the copy of comparison-options.yaml as needed. See Table 6.
4. Save your changes and quit the editor.

- 5. Use the updated copy of `comparison-options.yaml` when executing `compare-local-configuration-files`.

A customized comparison report is generated.

5 configuration-list

`configuration-list` connects to the PDB server and retrieves a list of node configurations that match the optional filters. If required, the list of configurations can be exported to a plain-text file for future use.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

The text file that is output by this command has the following format, one configuration per line:

```
/<root configuration>[/<delta configuration>[/...]], <revision>
```

By default, the `configuration-list` command will list all of the configurations that are visible to the selected user on the PDB server. This list can be filtered with the following options:

Name	Only configurations with names that match the selected substring will be included.
Owner	Only configurations with owners that match the selected substring will be included.
Revision	Only configurations with revisions that match the selected substring will be included.

5.1 Usage Syntax

```
usage: configuration-list [-h] [-v] [-u USER] [-p PASS] [-n NAME] [-o OWNER] [-r REVISION] [-f FILE]
```



5.2 Options

Table 7 create-variables-file Options

Option	Description	Notes
-n, --config-name <arg>	Specifies a substring to filter the list of configurations from PDB by name.	Optional. Partial matches are supported. If any part of the configuration name matches this substring, it will be included.
-o, --output <arg>	Specifies the path and filename for a new plain-text output file containing the list of configurations.	Optional. Paths can be relative or absolute. By default, the command outputs the matching configurations on screen.
-O, --config-owner <arg>	Specifies a substring to filter the list of configurations from PDB by owner.	Optional. Partial matches are supported. If any part of the configuration owner matches this substring, it will be included.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.



Table 7 create-variables-file Options

Option	Description	Notes
-r, --config-revision <arg>	Specifies a substring to filter the list of configurations from PDB by revision level. This option can be set to "latest" to return only the latest revisions of the available configurations.	Optional. Partial matches are supported. If any part of the configuration revision matches this substring, it will be included.
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. To access node configurations with this command, the selected user account must have the necessary permissions to view the configurations. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).

5.3 Examples

Note: You must use double quotes "<" ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```

$ cat CSCF_Configs.txt

```

This command lists latest revision of node configurations in PDB that have a name containing "CSCF_13A". The configuration list will be output to CSCF_Configs.txt in the current directory.



6 configuration-revision-set-status

`configuration-revision-set-status` can be used as an alternative to the PDB GUI to set the document state of a configuration revision. Document states must proceed through the following sequence:

PREL > FROZ

Note: Only PDB System Administrators can set a configuration to a previous state.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

6.1 Usage Syntax

.....
.....
.....

6.2 Options

Table 8 configuration-revision-set-status Options

Option	Description	Notes
<code>-n, --name <arg></code>	Specifies the name of a configuration in PDB.	Mandatory.
<code>-p, --password <arg></code>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
<code>-r, --revision <arg></code>	Specifies a revision of the selected configuration. The command defaults to the latest revision if this option is not used.	Optional.



Table 8 configuration-revision-set-status Options

Option	Description	Notes
-s, --status <arg>	Specifies a document state for the selected configuration revision. Valid states are: <ul style="list-style-type: none"> • PREL • FROZ 	Mandatory.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).

6.3 Examples

Note: You must use double quotes "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```

-----
-----

```

This command advances the document state of the HSS 15B Lab1 configuration , revision PB6, from PREL to FROZ.

7 configuration-revision-set-tag

`configuration-revision-set-tag` can be used as an alternative to the PDB GUI to apply one or more tags to a configuration revision stored in PDB. The command labels the selected configuration revision with one or more descriptive tags. These tags are used to categorize and add prominence to the configuration in PDB.



Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

7.1 Usage Syntax

```
usage: configuration-revision-set-tag -n CONFIGURATION_NAME [-r REVISION] [-t TAGS]
```

7.2 Options

Table 9 configuration-revision-set-tag Options

Option	Description	Notes
-n, --configuration-name <arg>	Specifies the name of a configuration in PDB.	Mandatory.
-r, --configuration-revision <arg>	Specifies a revision of the selected configuration to work with. The command defaults to the latest revision if this option is not used.	Optional.
-t, --tag <arg>	Specifies a comma-separated list of tags to add to the selected configuration revision. The following options are available: <ul style="list-style-type: none">• IVL - The IVL tag marks the configuration as an Initial Value List.• IFN - The IFN tag marks the configuration as Imported From Node.• MPVL - The MPVL tag marks the configuration as a Master Parameter Value List.• DPVL - This DPVL tag marks the configuration as a Delta Parameter Value List.• ALL - This option applies all tags to the selected configuration.	Mandatory. Tags are not case-sensitive.



Table 9 configuration-revision-set-tag Options

Option	Description	Notes
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]) .

7.3 Examples

Note: You must use double quotes "<" ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
#####
#####
```

This command tags the PA5 revision of the CSCFv 15A node configuration with the **IVL** and **MPVL** tags.

8 configuration-revision-unset-tag

configuration-revision-unset-tag can be used as an alternative to the PDB GUI to remove one or more tags from a configuration revision stored in PDB.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

8.1 Usage Syntax

```

ghetto <- list(ghettos = names, det_tag = "multiquantiles_names")

f <- multiquantiles_names(det_tag) <- (get_tag(tag),...)

#> factor_names [5] (factor_names)

```

8.2 Options

Table 10 configuration-revision-unset-tag Options

Option	Description	Notes
-n, --configuration-name <arg>	Specifies the name of a configuration in PDB.	Mandatory.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-r, --configuration-revision <arg>	Specifies a revision of the selected configuration to work with. The command defaults to the latest revision if this option is not used.	Optional.



Table 10 configuration-revision-unset-tag Options

Option	Description	Notes
-t, --tag <arg>	Specifies a comma-separated list of tags to remove from the selected configuration revision. The following options are available: <ul style="list-style-type: none"> • IVL • IFN • MPVL • DPVL • ALL - This option removes all tags from the selected configuration. 	Mandatory. Tags are not case-sensitive.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]) .

8.3 Examples

Note: You must use double quotes ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```

# configuration-revision-unset-tag -t IVL,MPVL -u admin

```

This command removes the **IVL** and **MPVL** tags from the PA5 revision of the CSCFv 15A node configuration.



9 convert-configuration

`convert-configuration` is a standalone configuration converter that can be used to convert configuration files between the supported formats. The command produces a configuration bundle that contains the converted configuration files. With the exception of configurations converted to the PVL format, all configuration bundles produced by `convert-configuration` contain the necessary logic to push the converted configuration to a target node.

To perform a conversion, `convert-configuration` takes one or more files representing a configuration (such as LDIF files) as input. In addition to the required base configuration, the command accepts an optional set of delta configurations that are applied sequentially (in order of appearance) on top of the base configuration. The delta configuration files must share the same schema as the base configuration.

The following table outlines the supported configuration formats.

Table 11 Supported Formats

Supported Input Formats	Supported Output Formats
CSV	CSV
LDIF	EAS
NETCONF	LDIF
PVL	NETCONF
	PVL

When converting a configuration to the LDIF format, the converter supports the use of Initial Value Lists (IVLs). IVLs represent the configuration of an LDAP node after maiden installation. When using an IVL, the converter considers the initial values to produce LDIF files that will not collide with the configuration data that is assumed to already exist in the real node.

9.1 Usage Syntax

```
convert-configuration <input> <output>
convert-configuration <input> <output> <delta>
convert-configuration <input> <output> <delta> <delta>
convert-configuration <input> <output> <delta> <delta> <delta>
convert-configuration <input> <output> <delta> <delta> <delta> <delta>
convert-configuration <input> <output> <delta> <delta> <delta> <delta> <delta>
```



9.2 Options

Table 12 convert-configuration Options

Option	Description	Notes
-a, --application <arg>	Species the node application that is associated with this configuration.	Mandatory only when using a variables file.
-c, --configuration-file <arg>	Specifies the path and filename of a configuration file to convert.	Mandatory. Paths can be relative or absolute. Multiple inputs are specified using additional -c options.
-C, --delta-file <arg>	Specifies the path and filename of a delta configuration file in PVL format to apply on top of the base configuration.	Optional Multiple inputs are specified using additional -C options. Delta configurations must be in PVL format. These configurations must share the same schema as the base configuration. You must specify a solution type for the delta files. If the base configuration is PVL, any delta configurations must share the same solution type.
-d, --use-delta	This option instructs the configuration converter to process values from the delta column in addition to the specified solution type.	Optional. Valid for PVL input files only.
-f, --input-format <arg>	Specifies the format of the node configuration input file. Possible formats include: <ul style="list-style-type: none"> • cpp_csv • ldif • netconf • pvl 	Mandatory.



Table 12 convert-configuration Options

Option	Description	Notes
-F, --output-format <arg>	Specifies the format of the node configuration output file. Possible formats include: <ul style="list-style-type: none">• csv• eas• ldif• netconf• pvl	Mandatory.
-i, --initial-values-file <arg>	Specifies the path and filename of a configuration file to use as an IVL for a LDIF conversion.	Optional. Paths can be relative or absolute.
-I, --initial-values-format <arg>	Specifies the format of the IVL configuration.	Optional.
-m, --mim-files <arg>	Specifies MIM files, directories, or ZIP archives that will be used as a schema for the converted configuration.	Mandatory. Paths can be relative or absolute. This option supports multiple MIM files, directories, or ZIP archives as input. Multiple inputs are specified using additional -m options.
-M, --moi-format <arg>	Specifies which MOI format to apply to the conversion of a PVL configuration. Possible formats include: <ul style="list-style-type: none">• tsp• is• cba	Mandatory for PVL input files only. For more information on the PVL MOI format, refer to the Parameter List Template Description, EAB/FTI-08:0686 Uen.
-o, --output <arg>	Specifies a path and filename for the converted configuration archive.	Mandatory. Paths can be relative or absolute.



Table 12 convert-configuration Options

Option	Description	Notes
-R, --report-file <arg>	Specifies a path and filename for the error report file. The output defaults to stdout if this option is not set.	Optional. Paths can be relative or absolute.
-s, --solution-type <arg>	Specifies which PVL parameter value set to convert. Possible solution types include: <ul style="list-style-type: none"> • FIXED • MOBILE • CONVERGED • STANDALONE • INITIAL_VALUE 	Mandatory for PVL input files only.
-t, --archive-type <arg>	Specifies the output format for the converted configuration archive. Possible formats include: <ul style="list-style-type: none"> • tar • zip 	Optional. The configuration converter defaults to the ZIP format.
-v, --variables-file <arg>	Specifies the path and filename of a variables list that can be used to resolve any parameter value variables in the reference configuration before comparing.	Optional. Paths can be relative or absolute. The variables list is a user-defined CSV file that adheres to the following format, one variable per line: <name>, <value>, <description>, <usage> Use the <code>create-variables-file</code> CLI command to generate a file that you can populate with values.

9.3 Examples

Note: You must use double quotes "> to surround entries that have spaces.



Sample Command Usage on a Linux Platform

```
##### PDB Command Usage #####  
  
1. PDB Command Usage: PDB Command Usage  
  
2. PDB Command Usage: PDB Command Usage  
  
3. PDB Command Usage: PDB Command Usage  
  
4. PDB Command Usage: PDB Command Usage  
  
5. PDB Command Usage: PDB Command Usage  
  
6. PDB Command Usage: PDB Command Usage  
  
7. PDB Command Usage: PDB Command Usage  
  
8. PDB Command Usage: PDB Command Usage  
  
9. PDB Command Usage: PDB Command Usage  
  
10. PDB Command Usage: PDB Command Usage
```

This command converts the converged column of the CSCF_11A_R4B PVL configuration file to the LDIF format using initial values provided by the CSCF_11A_R4B.ldif configuration. The ss_list.lst variables file will be used to resolve any site-specific variables.

```
##### PDB Command Usage #####  
  
1. PDB Command Usage: PDB Command Usage  
  
2. PDB Command Usage: PDB Command Usage  
  
3. PDB Command Usage: PDB Command Usage  
  
4. PDB Command Usage: PDB Command Usage  
  
5. PDB Command Usage: PDB Command Usage  
  
6. PDB Command Usage: PDB Command Usage  
  
7. PDB Command Usage: PDB Command Usage  
  
8. PDB Command Usage: PDB Command Usage  
  
9. PDB Command Usage: PDB Command Usage  
  
10. PDB Command Usage: PDB Command Usage
```

This command converts the fixed column of the cscf_mpv1_pdb.xml PVL configuration file to the LDIF format. The contents of delta-pvl2.xml delta configuration will be applied on top of the base configuration and the ssl.csv variables file will be used to resolve any site-specific variables.

10 create-variables-file

`create-variables-file` generates a template file containing all of the parameter variables required by a selected node configuration. The command retrieves the list of parameter variables for a selected configuration from PDB server and saves them as a CSV file.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

The variables file that is output by this command has the following format, one variable per line:

`<name>,<value>,<description>,<usage>`

Note: The `description` and `usage` fields are used by PDB to provide additional information and should not be modified.

Values must be added to the file before it can be used in conjunction with other CLI commands, including:



- compare-configuration-file
- convert-configuration
- export-configuration
- validate-mpvl

10.1 Usage Syntax

```

1 1. create-variables-file -f <format> -n <name> -o <output> [-p <password>]
2
3

```

10.2 Options

Table 13 create-variables-file Options

Option	Description	Notes
-f, --format <arg>	Specifies the configuration format. Possible formats include: <ul style="list-style-type: none"> • LDIF • NETCONF • EAS • PVL 	Mandatory. PDB uses the selected format to add the correct Configuration Management (CM) variables to the site-specific list. For more information on CM variables, refer to Configuration Management Variables in the PDB User Guide (Reference [2]) .
-n, --configuration-name <arg>	Specifies the name of a configuration in PDB to use as a source for site-specific variables.	Mandatory.
-o, --output <arg>	Specifies a path and filename for the new variables file.	Mandatory. Paths can be relative or absolute.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.

Table 13 create-variables-file Options

Option	Description	Notes
-r, --revision <arg>	Specifies a revision of the selected configuration.	Optional. If this option has not been set, the <code>create-variables-file</code> command will automatically export the highest revision of the selected configuration.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]) .

10.3 Examples

Note: You must use double quotes "<" ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

This command retrieves site-specific variables from the latest revision of the CSCF_11A_R4B node configuration and stores them in the CSCF_11A_R4B_ss.lst file.

11 export-configuration

export-configuration can be used as an alternative to the PDB GUI to export node configurations. The command pulls a selected configuration from the PDB database and packages it in a ZIP or TAR archive that contains the necessary logic to configure the target node.



Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

11.1 Usage Syntax

```

export-configuration [-d] [-f format] [-n configuration-name] [-o output] [-p password]

```

11.2 Options

Table 14 export-configuration Options

Option	Description	Notes
-d, --use-delta	Activates the Delta Only option. Select this option to restrict the exported configuration data to information that is local to the delta configuration.	Optional. This option is only applicable to delta configurations.
-f, --format <arg>	Specifies a format for the exported configuration. Possible formats include: <ul style="list-style-type: none"> • eas • Idif • netconf • pvl 	Mandatory. Configurations exported in the PVL format only include data from the <code>ims_fixed</code> column.
-n, --configuration-name <arg>	Specifies the name of a configuration to export from PDB.	Mandatory.
-o, --output <arg>	Specifies a destination archive for the exported configuration.	Mandatory
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.



Table 14 export-configuration Options

Option	Description	Notes
-r, --revision <arg>	Specifies a revision of the selected configuration to export.	Optional. If this option has not been set, the export-configuration command will automatically export the highest revision of the selected configuration.
-s, --solution-type <arg>	Specifies which parameter value set to export. Possible solution types include: <ul style="list-style-type: none">• FIXED• MOBILE• CONVERGED• STANDALONE The command defaults to FIXED if this option is not specified.	Optional.
-t, --archive-type <arg>	Specifies the output format for the exported configuration archive. Possible formats include: <ul style="list-style-type: none">• tar• tar_ait• zip	Optional. The configuration exporter defaults to the ZIP format. tar_ait produces an Automatic Installation Tool (AIT) compliant TAR file that contains additional information intended for use by AIT. This format is only available when exporting a CBA-based node configuration in NETCONF format.



Table 14 export-configuration Options

Option	Description	Notes
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	<p>Mandatory.</p> <p>User accounts must be provisioned in the PDB authentication realm.</p> <p>User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).</p>
-v, --variables-file <arg>	Specifies the path and filename of a variables list that is used to resolve any parameter value variables in the exported configuration.	<p>Optional. Using this option makes -V, --ssl-name unavailable.</p> <p>Paths can be relative or absolute.</p> <p>The variables list is a user-defined CSV file that adheres to the following format, one variable per line:</p> <pre><name>, <value>, <description>, <usage></pre> <p>Use the <code>create-variables-file</code> CLI command to generate a file that you can populate with values.</p>
-V, --ssl-name <arg>	Specifies the name of a site-specific list on the PDB server that is used to resolve any parameter value variables in the exported configuration.	<p>Optional. Using this option makes -v, --variables-file unavailable.</p> <p>The specified PDB user must have access to the selected site-specific list.</p> <p>This option can be specified multiple times.</p>

11.3 Examples

Note: You must use double quotes ">" to surround entries that have spaces.



Sample Command Usage on a Linux Platform

```
## Export the latest revision of the HSS 5.0 configuration in Idif format to Configurations/ldif_hss50.zip.
```

This command exports the latest revision of the HSS 5.0 configuration in Idif format to Configurations/ldif_hss50.zip.

```
## Export the latest revision of the Delta HSS 5.0 configuration in Idif format to Configurations/do_hss50.zip. This operation has been set to Delta Only, meaning that only configuration data locally defined in the delta configuration will be exported.
```

This command exports the latest revision of the Delta HSS 5.0 configuration in Idif format to Configurations/do_hss50.zip. This operation has been set to Delta Only, meaning that only configuration data locally defined in the delta configuration will be exported.

```
## Export revision PC3 of the MGW 1.3.5 configuration in netconf format to Configurations/mgw135.tar and prompts for the user password.
```

This command exports revision PC3 of the MGW 1.3.5 configuration in netconf format to Configurations/mgw135.tar and prompts for the user password.

12 import-configuration

import-configuration can be used as an alternative to the PDB GUI to import a new node configuration. The command takes configuration data from one or more files, directories, or archives and imports them to the PDB database as a new configuration.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

12.1 Usage Syntax

```
## Import configuration from a directory.
```



12.2 Options

Table 15 import-configuration Options

Option	Description	Notes
-c, --configuration-files <arg>	Specifies the configuration files, directories, or archives that constitute the node configuration.	Mandatory. Paths can be relative or absolute. This option supports multiple configuration files, directories, or archives as input. Multiple inputs are specified using additional -c options.
-C, --comment<arg>	Specifies a comment for the new revision.	Optional.
-d, --use-delta	This option instructs import-configuration to process values from the delta column in addition to the specified solution type in a PVL configuration file.	Optional. This option is only applicable to delta configurations using the PVL configuration format.
-D, --document-number <arg>	Specifies a document number for the new configuration. PDB will assign an internal number if this option is not specified.	Optional.
-e, --configuration-description <arg>	Specifies a configuration description.	Optional.
-f, --format <arg>	Specifies the format for of the incoming configuration data. Possible formats include: <ul style="list-style-type: none"> • cpp_csv • ldif • netconf • pvl 	Mandatory.
-F, --freeze	Sets the document state of the new configuration to FROZ.	Optional.
-n, --configuration-name <arg>	Specifies the name of the new configuration in PDB.	Mandatory.



Table 15 import-configuration Options

Option	Description	Notes
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-r, --revision <arg>	Specifies a revision level for the imported configuration data.	Optional. If this option has not been set, import-configuration defaults to "PA1".
-R, --report-file <arg>	Specifies a path and filename for the validation report. The output defaults to stdout if this option is not set.	Optional. Paths can be relative or absolute.
-s, --solution-type <arg>	Specifies which PVL parameter value set to import. Possible solution types include: <ul style="list-style-type: none">• FIXED• MOBILE• CONVERGED• STANDALONE• INITIAL_VALUE	Mandatory only when importing PVL configuration files. The INITIALVALUE syntax is deprecated and has been replaced by INITIAL_VALUE.
-S, --schema-revision <arg>	If required, this option specifies which a schema revision to use with the imported configuration data.	Optional. If this option has not been set, import-configuration defaults to the latest revision.
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).



12.3 Examples

Note: You must use double quotes <" "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```

import-configuration-revision -f /path/to/ldif_hss11a_r9b_pb2.zip

```

This command imports the a new configuration named HSS_11A_R9B_LDIF in LDIF format from the Configurations/ldif_hss11a_r9b_pb2.zip configuration archive.

```

import-configuration-revision -f /path/to/mtas11b.pvl

```

This command imports a new configuration named MPVL_MTAS_11B_FIXED in PVL format from the Configurations/mtas11b.pvl configuration file. PDB will use values from the FIXED solution type and revision PA4 of the associated schema. The PDB user will be prompted for their password.

13 import-configuration-revision

import-configuration-revision can be used as an alternative to the PDB GUI to import a new revision of an existing configuration. The command takes configuration data from one or more files, directories, or archives and imports them to the PDB database as a new revision of an existing configuration. The previous revision must be frozen before a new revision can be added.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

13.1 Usage Syntax

```

import-configuration-revision -f /path/to/ldif_hss11a_r9b_pb2.zip
import-configuration-revision -f /path/to/mtas11b.pvl
import-configuration-revision -f /path/to/mtas11b.pvl -s /path/to/schema
import-configuration-revision -f /path/to/mtas11b.pvl -s /path/to/schema -t /path/to/type
import-configuration-revision -f /path/to/mtas11b.pvl -s /path/to/schema -t /path/to/type -r /path/to/revision

```



13.2 Options

Table 16 import-configuration-revision Options

Option	Description	Notes
-c, --configuration-files <arg>	Specifies the configuration files, directories, or archives that constitute the node configuration.	Mandatory. Paths can be relative or absolute. This option supports multiple configuration files, directories, or archives as input. Multiple inputs are specified using additional -c options.
-C, --comment<arg>	Specifies a comment for the new revision.	Optional.
-d, --use-delta	This option instructs import-configuration-revision to process values from the delta column in addition to the specified solution type in a PVL configuration file.	Optional. This option is only applicable to delta configurations using the PVL configuration format.
-f, --format <arg>	Specifies the format for of the incoming configuration data. Possible formats include: <ul style="list-style-type: none">• cpp_csv• ldif• netconf• pvl	Mandatory.
-F, --freeze	Sets the document state of the new configuration revision to FROZ.	Optional.
-n, --configuration-name <arg>	Specifies the name of a frozen configuration in PDB to update as a new revision.	Mandatory.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.



Table 16 import-configuration-revision Options

Option	Description	Notes
-r, --revision <arg>	Specifies a revision level for the imported configuration data.	Optional. If this option has not been set, the <code>import-configuration-revision</code> command defaults to the next legal revision of the selected configuration.
-R, --report-file <arg>	Specifies a path and filename for the validation report. The output defaults to stdout if this option is not set.	Optional. Paths can be relative or absolute.
-s, --solution-type <arg>	Specifies which PVL parameter value set to import. Possible solution types include: <ul style="list-style-type: none"> • FIXED • MOBILE • CONVERGED • STANDALONE • INITIAL_VALUE 	Mandatory only when importing PVL configuration files. The <code>INITIALVALUE</code> syntax is deprecated and has been replaced by <code>INITIAL_VALUE</code> .
-S, --schema-revision <arg>	If necessary, this option can be used to specify a new schema revision to use with the imported configuration data.	Optional.
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to <i>Access Control Lists</i> in the <i>PDB User Guide</i> (Reference [3]).

13.3 Examples

Note: You must use double quotes ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

[illegible]

This command imports the next legal revision (PB2) of the HSS_11A_R9B_LDIF configuration from the Configurations/ldif_hss11A_r9b_pb2.zip configuration archive.

[illegible]

This command imports a new PA4 revision of the MPVL_MTS_11B_FIXED configuration from the Configurations/mtas11b.pv1 configuration file. PDB will use values from the FIXED solution type and prompts for the user password.

```
14 import-schema-revision
```

`import-schema-revision` can be used as an alternative to the PDB GUI to import a new revision of an existing schema. The command retrieves data from one or more MIM files, directories, or archives and imports them to the PDB database as a new revision of an existing schema.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

14.1 Usage Syntax

```
getenv("PYTHONPATH") && PYTHONPATH=$(PYTHONPATH):$(pwd)
export PYTHONPATH

if [ -f "test.py" ]; then python test.py; fi
```

14.2 Options

Table 17 import-schema-revision Options

Option	Description	Notes
<code>-C, --comment<arg></code>	Specifies a comment for the new revision.	Optional.



Table 17 import-schema-revision Options

Option	Description	Notes
-F, --freeze	Sets the document state of the new schema revision to FROZ.	Optional.
-m, --mim-files <arg>	Specifies the files, directories, or archives that constitute the configuration schema.	<p>Mandatory.</p> <p>Paths can be relative or absolute.</p> <p>This option supports multiple schema files, directories, or archives as input.</p> <p>Multiple inputs are specified using additional -m options.</p>
-n, --schema-name <arg>	Specifies the name of a schema in PDB to update as a new revision.	Mandatory.
-p, --password <arg>	The PDB user password.	<p>Optional.</p> <p>If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.</p>
-r, --revision <arg>	Specifies a revision level for the imported schema data.	<p>Optional.</p> <p>If this option has not been set, the <code>import-schema-revision</code> command defaults to the next legal revision of the selected schema.</p>
-R, --report-file <arg>	Specifies a path and filename for the validation report. The output defaults to stdout if this option is not set.	<p>Optional.</p> <p>Paths can be relative or absolute.</p>
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	<p>Mandatory.</p> <p>User accounts must be provisioned in the PDB authentication realm.</p> <p>User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).</p>



14.3 Examples

Note: You must use double quotes <" "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
#####
#####
#####
```

This command imports the next legal revision (PB1) of the HSS_CM_MIM schema from the Schemas/hss_mim.zip archive.

```
#####
#####
```

This command imports a new PA2 revision of the MTAS_CM_MIM schema from the Schemas/mtas11b/ directory and prompts for the user password.

15 node-create-revision

`node-create-revision` can be used as an alternative to the PDB GUI to create a new revision of a node defined in PDB. The command creates a new revision of the specified node with an optional description. If the node revision has already been defined, specifying a new description will overwrite the existing one.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

15.1 Usage Syntax

```
#####
#####
```



15.2 Options

Table 18 node-create-revision Options

Option	Description	Notes
-d, --description <arg>	Specifies a description for the node revision.	Optional. If the node revision has already been defined, specifying a new description will overwrite the existing one.
-n, --node-name <arg>	Specifies the name of a node in PDB.	Mandatory.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-r, --node-revision <arg>	Specifies a revision of the selected node to create.	Mandatory.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).

15.3 Examples

Note: You must use double quotes "<" ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```

node-create-revision -n CSCF13B -d "WP265A" -r R11B02

```

This command creates a new R11B02 revision for the CSCF 13B node described as "WP265A".



16 node-revision-link-configuration-revision

node-revision-link-configuration-revision can be used as an alternative to the PDB GUI to link a configuration revision to a node revision. The selected configuration must be based on a schema that is associated with the specified node.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

16.1 Usage Syntax

node-revision-link-configuration-revision

16.2 Options

Table 19 node-revision-link-configuration-revision Options

Option	Description	Notes
-c, --configuration-name <arg>	Specifies the name of a configuration in PDB.	Mandatory.
-cr, --configuration-revision <arg>	Specifies a revision of the selected configuration.	Mandatory.
-n, --node-name <arg>	Specifies the name of a node in PDB.	Mandatory.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.



Table 19 node-revision-link-configuration-revision Options

Option	Description	Notes
-r, --node-revision <arg>	Specifies a revision of the selected node.	Mandatory.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	<p>Mandatory.</p> <p>User accounts must be provisioned in the PDB authentication realm.</p> <p>User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).</p>

16.3 Examples

Note: You must use double quotes ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
node-revision-link-schema-revision -r PA2 -u admin
```

This command links the PA2 revision of the MPVL HSS 14B FIXED configuration to the R4B revision of the HSS 14B node.

17 node-revision-link-schema-revision

node-revision-link-schema-revision can be used as an alternative to the PDB GUI to link a schema revision to a node revision.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

17.1 Usage Syntax

17.2 Options

Table 20 node-revision-link-schema-revision Options

Option	Description	Notes
-n, --node-name <arg>	Specifies the name of a node in PDB.	Mandatory.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-r, --node-revision <arg>	Specifies a revision of the selected node.	Mandatory.
-s, --schema-name <arg>	Specifies the name of a configuration schema in PDB.	Mandatory.
-sr, --schema-revision <arg>	Specifies a revision of the selected configuration schema.	Mandatory.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).

17.3 Examples

Note: You must use double quotes "<" ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
schema-compare -b /path/to/base/schema/ -t /path/to/target/schema/
```

This command links the PA2 revision of the HSS 14B configuration schema to the R4B revision of the HSS 14B node.

18 schema-compare

schema-compare is a comparison tool that can be used as an alternative to the PDB GUI to examine the differences between two sets of schema files. The command outputs a comparison report in CSV format. If the schema files are identical, this report is empty and the exit status will be 0.

The selected schema files are validated before each comparison to ensure proper syntax and element relationships.

18.1 Usage Syntax

```
schema-compare -b /path/to/base/schema/ -t /path/to/target/schema/
```

18.2 Options

Table 21 schema-compare Options

Option	Description	Notes
<code>-base -base, --base-mim-files <arg></code>	Specifies the path to a directory, schema file, or archive file (ZIP/TAR) of the base schema that will serve as the starting point for the comparison.	Mandatory. Paths can be relative or absolute. This option can be specified multiple times to capture all of the necessary schema file.



Table 21 schema-compare Options

Option	Description	Notes
-c, --criteria <arg>	<p>Specifies the schema properties to include in the comparison report.</p> <p>Possible properties include:</p> <ul style="list-style-type: none">• card - Cardinality• cat - Category• cons - Value Constraints• desc - Description• fdesc - Format Description• def - Default Values• m - Missing Classes/Attributes• case - Letter Case• pk - Primary Key• ro - Read-Only• stat - Status <p>If this option is not specified, the command defaults to ALL.</p>	<p>Optional.</p> <p>For more information on the properties of a comparison report, refer to Comparing Two Schemas in the PDB User Guide (Reference [3])</p>
-R, --import-report-file <arg>	<p>Specifies a path and filename for a validation error report file. The output defaults to stdout if this option is not set.</p>	<p>Optional.</p> <p>Paths can be relative or absolute.</p>
-S, --comparison-report-file <arg>	<p>Specifies a path and filename for the comparison report.</p>	<p>Mandatory.</p> <p>Paths can be relative or absolute.</p> <p>The comparison report is generated in CSV format.</p>
-target , --target-mim-files <arg>	<p>Specifies the path to a directory, schema file, or archive file (ZIP/TAR) of the target schema for the comparison.</p>	<p>Mandatory.</p> <p>Paths can be relative or absolute.</p> <p>This option can be specified multiple times to capture all of the necessary schema files.</p>

18.3 Examples

Note: You must use double quotes <" "> to surround entries that have spaces. For sample usage, see the following example.

Sample Command Usage on a Linux Platform

```
##
##
##
```

This command compares the schema files contained in 13B.zip to the schema files contained within 13B_1.zip. Any schema validation errors will be output to errors.txt. The schema comparison report will be output to comp/13B/comp.csv.

19 schema-list

schema-list connects to the PDB server and retrieves a list of configuration schemas that match the optional filters. If required, the list can be exported to a plain-text file with one schema per line.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

By default, the schema-list command will list all of the configuration schemas that are visible to the selected user on the PDB server. This list can be filtered with the following options:

Name	Only schemas with names that match the selected substring are included.
Owner	Only schemas with owners that match the selected substring are included.
Revision	Only schemas with revisions that match the selected substring are included. This option default to the latest revision.

19.1 Usage Syntax

```
##
##
##
```



19.2 Options

Table 22 schema-list Options

Option	Description	Notes
-n, --schema-name <arg>	Specifies a substring to filter the list of schemas from PDB by name.	Optional. Partial matches are supported. If any part of the schema name matches this substring, it is included.
-o, --output <arg>	Specifies the path and filename for a new plain-text output file containing the list of schemas.	Optional. Paths can be relative or absolute. By default, the command outputs the matching schemas on screen.
-O, --schema-owner <arg>	Specifies a substring to filter the list of schemas from PDB by owner.	Optional. Partial matches are supported. If any part of the schema owner matches this substring, it is included.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.



Table 22 schema-list Options

Option	Description	Notes
-r, --schema-revision <arg>	Specifies a substring to filter the list of schemas from PDB by revision. The command defaults to the latest revision if this option is not used.	Optional. Partial matches are supported. If any part of the schema name matches this substring, it is included. Use all to display every schema revision that matches the other criteria.
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. To access schemas with this command, the selected user account must have the necessary permissions to view the schemas in PDB. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).

19.3 Examples

Note: You must use double quotes ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```

$ schema-list -u PDB_User -r CSCF

```

This command connects to PDB with the PDB_User account and lists the latest revision of schemas visible to that user in PDB with names containing "CSCF". The schema list is output to CSCF schemas.txt in the current directory.



20 schema-revision-set-ivl

`schema-revision-set-ivl` can be used as an alternative the PDB GUI to associate an Initial Value List (IVL) with a configuration schema stored in PDB. If the selected configuration schema already has an IVL, this command overwrites it.

Node configurations stored in PDB must be tagged as IVL before they can be associated with a schema. For more information on tagging configurations as IVL, refer to `configuration-revision-set-tag` or [Modifying Configuration Properties in the PDB User Guide \(Reference \[3\]\)](#)

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

20.1 Usage Syntax

```
schema-revision-set-ivl  
-c <arg>  
-cr <arg>  
-p <arg>  
-r <arg>
```

20.2 Options

Table 23 `schema-revision-set-ivl` Options

Option	Description	Notes
<code>-c, --configuration-name <arg></code>	Specifies the name of a configuration in PDB.	Mandatory.
<code>-cr, --configuration-revision <arg></code>	Specifies a revision of the selected configuration. The command defaults to the latest revision if this option is not used.	Optional.
<code>-p, --password <arg></code>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
<code>-r, --schema-revision <arg></code>	Specifies a revision of the selected configuration schema. The command defaults to the latest revision if this option is not used.	Optional.



Table 23 schema-revision-set-ivl Options

Option	Description	Notes
-s, --schema-name <arg>	Specifies the name of a configuration schema in PDB.	Mandatory.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).

20.3 Examples

Note: You must use double quotes "<" ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

2017-11-17 10:10:10
 2017-11-17 10:10:10

This command sets revision B of the HSS_Lab_1 configuration as IVL for the latest revision of the HSS 15A configuration schema.

21 schema-revision-set-status

schema-revision-set-status can be used as an alternative to the PDB GUI to set the document state of a schema revision. Document states must proceed through the following sequence:

PREL > FROZ

Note: Only PDB System Administrators can set a schema to a previous state. System Administrators cannot unfreeze a schema revision that is the basis for any frozen configurations.



Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

21.1 Usage Syntax

21.1.1
21.1.2
21.1.3

21.2 Options

Table 24 schema-revision-set-status Options

Option	Description	Notes
-n, --name <arg>	Specifies the name of a configuration schema in PDB.	Mandatory.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-r, --revision <arg>	Specifies a revision of the selected configuration schema. The command defaults to the latest revision if this option is not used.	Optional.
-s, --status <arg>	Specifies a document state for the selected schema revision. Valid states are: <ul style="list-style-type: none">• PREL• FROZ	Mandatory.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).



21.3 Examples

Note: You must use double quotes <" "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

#####

This command advances the document state of the HSS 14B configuration schema, revision PA3, from PREL to FROZ.

22 schema-revision-unset-ivl

schema-revision-set-ivl can be used as an alternative the PDB GUI to remove an IVL association from a configuration schema stored in PDB.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

22.1 Usage Syntax

#####

22.2 Options

Table 25 schema-revision-unset-ivl Options

Option	Description	Notes
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-r, --schema-revision <arg>	Specifies a revision of the selected configuration schema. The command defaults to the latest revision if this option is not used.	Optional.



Table 25 schema-revision-unset-ivl Options

Option	Description	Notes
-s, --schema-name <arg>	Specifies the name of a configuration schema in PDB.	Mandatory.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).

22.3 Examples

Note: You must use double quotes <" "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
1/1540-CXP 902 0212 Uen U | 2017-11-17
```

This command removes the IVL associated with the latest revision of the HSS 15A configuration schema.

23 ssl-export

`ssl-export` can be used as an alternative to the PDB GUI to export site-specific lists. The command pulls a selected SSL from the PDB database and exports it as a list of parameter value variables in CSV or Testing and Test Control Notation (TTCN) format.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide](#) (Reference [2]).



CSV Format

When exporting an SSL in CSV format, `ssl-export` creates a variables file with a ".csv" file extension to store the exported data. These files have the following structure, one variable per line:

```
<name>,<value>,<description>,<usage>
```

Example 2 shows a CSV variables file with sample data.

```

1,1,1,1
2,2,2,2
3,3,3,3
4,4,4,4
5,5,5,5
6,6,6,6
7,7,7,7
8,8,8,8
9,9,9,9

```

Example 2 Sample CSV Variables File

CSV variables files can be used with other CLI commands to supply values for parameter value variables. These commands include:

- `compare-configuration-file`
- `convert-configuration`
- `export-configuration`
- `validate-mpvl`

TTCN Format

The TTCN format is used to provide SSL data to Titansim test bundles. When exporting an SSL in TTCN format, `ssl-export` creates a variables file with a ".cfg" file extension to store the exported data. These files have the following format, one variable per line:

```
<name> := "<value>"
```

Example 3 shows a TTCN variables file with sample data.

```

1:= "1"
2:= "2"
3:= "3"
4:= "4"
5:= "5"
6:= "6"
7:= "7"
8:= "8"
9:= "9"

```

Example 3 Sample TTCN Variables File



23.1 Usage Syntax

23.1.1 Usage Syntax

23.2 Options

Table 26 ssl-export Options

Option	Description	Notes
-f, --format <arg>	Specifies an output format for the exported SSL. Possible formats include: <ul style="list-style-type: none">• csv• ttcn The format defaults to CSV if this option is not set.	Optional.
-n, --ssl-name <arg>	Specifies the name of a site-specific list to export from PDB.	Mandatory.
-o, --output <arg>	Specifies a path and filename for the new variables file.	Mandatory. Paths can be relative or absolute.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-u, --username<arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. User accounts must have the necessary permissions to access the selected objects. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]) .

23.3 Examples

Note: You must use double quotes <" "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```

$ ssh -X root@10.10.10.10
root@10.10.10.10:~#

```

This command exports the SSL_lab1 site-specific list from PDB to /tmp/pdb/lab1_ss.csv in CSV format.

24 ssl-list

ssl-list connects to the PDB server and retrieves a list of site-specific lists that match the optional filters. If required, the list can be exported to a plain-text file with one list per line.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

By default, the ssl-list command will list all of the SSLs that are visible to the selected user on the PDB server. This list can be filtered with the following options:

Name	Only SSLs with names that match the selected substring are included.
Owner	Only SSLs with owners that match the selected substring are included.

24.1 Usage Syntax

```

$ ssh -X root@10.10.10.10
root@10.10.10.10:~#

```



24.2 Options

Table 27 ssl-list Options

Option	Description	Notes
<code>-n, --ssl-name <arg></code>	Specifies a substring to filter the list of SSLs from PDB by name.	Optional. Partial matches are supported. If any part of the SSL name matches this substring, it will be included.
<code>-o, --output <arg></code>	Specifies the path and filename for a new plain-text output file containing the list of SSLs.	Optional. Paths can be relative or absolute. By default, the command outputs the matching SSLs on screen.
<code>-O, --ssl-owner <arg></code>	Specifies a substring to filter the list of SSLs from PDB by owner.	Optional. Partial matches are supported. If any part of the SSL owner matches this substring, it will be included.
<code>-p, --password <arg></code>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
<code>-u, --username <arg></code>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. To access SSLs with this command, the selected user account must have the necessary permissions to view the SSLs. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]) .

24.3 Examples

Note: You must use double quotes `<" ">` to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
ssh -X user@pdb-server
```

This command connects to PDB with the PDB_User account and lists SSLs visible to that user in PDB with names containing "Lab1". The SSL list is output to Lab1 SSL.txt in the current directory.

25

ssl-list-variable

ssl-list-variable connects to the PDB server and fetches variables from the specified site-specific list. Variables retrieved by the command are printed on screen.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

By default, the ssl-list-variable command lists all of the variables in the specified SSL. This list can be filtered with the following option:

Name

Only variables names that match the selected substring are included.

25.1

Usage Syntax

```
ssh -X user@pdb-server
```

25.2

Options

Table 28 ssl-list-variable Options

Option	Description	Notes
-n, --name <arg>	Specifies a substring to filter the list of SSL variables retrieved from PDB by name.	Optional. Partial matches are supported. If any part of the variable name matches this substring, it will be included.



Table 28 ssl-list-variable Options

Option	Description	Notes
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-s, --ssl <arg>	Specifies the name of a site-specific list in PDB.	Mandatory.
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. To access SSLs with this command, the selected user account must have the necessary permissions to view the SSLs. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).

25.3 Examples

Note: You must use double quotes "<" ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
ssh root@10.10.10.100  
root@10.10.10.100:~#
```

This command prints all variables in the CSCF 12C site-specific list that have a name containing "CSCF".

26 ssl-set-variable

`ssl-set-variable` can be used as an alternative to the PDB GUI to set a new value to an existing SSL variable or create a new variable with the specified value.



Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the PDB System Administration Guide (Reference [2]).

26.1 Usage Syntax

26.1.1 Usage Syntax

26.2 Options

Table 29 ssl-set-variable Options

Option	Description	Notes
-n, --name <arg>	Specifies a variable name.	Mandatory. If this variable does not exist in the specified SSL, it is created and populated with the selected value. If this variable already exists in the specified SSL, it is updated with the selected value.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.
-s, --ssl <arg>	Specifies the name of a site-specific list in PDB.	Mandatory.
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	Mandatory. User accounts must be provisioned in the PDB authentication realm. To access SSLs with this command, the selected user account must have the necessary permissions to view the SSLs. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).
-v, --value <arg>	Specifies a value for the selected variable.	Mandatory.



26.3 Examples

Note: You must use double quotes <" "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```

PDB CLI 902.0212 Uen U | 2017-11-17
PDB CLI 902.0212 Uen U | 2017-11-17

```

This command creates or updates the CSCF_IP_DNS variable in the CSCF 12C site-specific list with the value 11.82.165.241.

27 ssl-unset-variable

ssl-unset-variable can be used as an alternative to the PDB GUI to delete a variable from the specified site-specific list.

Note: This command requires connectivity with a PDB server and a valid PDB user account. For more information on configuring connectivity, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

27.1 Usage Syntax

```

PDB CLI 902.0212 Uen U | 2017-11-17
PDB CLI 902.0212 Uen U | 2017-11-17

```

27.2 Options

Table 30 ssl-unset-variable Options

Option	Description	Notes
-n, --name <arg>	Specifies a variable name to delete.	Mandatory.
-p, --password <arg>	The PDB user password.	Optional. If omitted, the PDB CLI prompts for the password interactively. Passwords input at the prompt are masked.

Table 30 ssl-unset-variable Options

Option	Description	Notes
-s, --ssl <arg>	Specifies the name of a site-specific list in PDB.	Mandatory.
-u, --username <arg>	Specifies a valid PDB user name that is used to authenticate the CLI user in PDB.	<p>Mandatory.</p> <p>User accounts must be provisioned in the PDB authentication realm.</p> <p>To access SSLs with this command, the selected user account must have the necessary permissions to view the SSLs. For more information on PDB access rights, refer to Access Control Lists in the PDB User Guide (Reference [3]).</p>

27.3 Examples

Note: You must use double quotes "<" ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

This command removes the `CSCF_IP_DNS` variable from the CSCF 12C site-specific list.

28 validate-local-configuration-files

`validate-local-configuration-files` is a standalone validation command that can be used to validate a local configuration files and generate a validation report. Validation considers standard file syntax, format rules, and the limits set by the selected MIM files, including schematron rules. CLI validation applies the same logic to verify input files as configuration validation in the PDB GUI, and generates the same type of report when problems are found.

For more information on validating node configurations through the GUI including a description of the validation report, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).



28.1 Usage Syntax

```
python pdbcli validate-local-configuration-files -c <arg>
python pdbcli validate-local-configuration-files -f <arg>
python pdbcli validate-local-configuration-files -M <arg>
```

28.2 Options

Table 31 validate-local-configuration-files Options

Option	Description	Notes
-c, --configuration-files <arg>	Specifies the path and filename of a configuration file to validate.	Mandatory. Paths can be relative or absolute. Multiple inputs are specified using additional -c options.
-f, --configuration-format <arg>	Specifies the configuration format. Possible formats include: <ul style="list-style-type: none">• cpp_csv• ldif• netconf• pvl	Mandatory.
-M, --moi-format <arg>	Specifies the MOI format of the PVL configuration. Possible formats include: <ul style="list-style-type: none">• tsp• is• cba	Mandatory for PVL configurations only. For more information on the PVL format, refer to the Parameter List Template Description, EAB/FTI-08:0686 Uen.



Table 31 validate-local-configuration-files Options

Option	Description	Notes
-s, --schema-files <arg>	Specifies a MIM file, directory, or ZIP archive that constitutes the configuration schema.	<p>Mandatory.</p> <p>Paths can be relative or absolute.</p> <p>When specifying a directory, subdirectories on the same path are automatically included.</p> <p>Multiple MIM files, directories, or ZIP archives are specified using additional -m options.</p>
-T, --solution-type <arg>	<p>Only applicable when validating PVL configuration files.</p> <p>Specifies which PVL parameter value set to validate.</p> <p>Possible solution types include:</p> <ul style="list-style-type: none"> • FIXED • MOBILE • CONVERGED • STANDALONE • INITIAL_VALUE 	<p>Optional.</p> <p>If a solution type is not specified, all solutions types will be validated, including initial values.</p> <p>The INITIALVALUE syntax is deprecated and has been replaced by INITIAL_VALUE.</p>
-V, --validation-report-file <arg>	Specifies a destination file for the validation report. The output defaults to stdout if this option is not set.	Optional.

28.3 Examples

Note: You must use double quotes ">" to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```

$ validate-local-configuration-files -s /path/to/schema -m /path/to/mim1 -m /path/to/mim2 -T CONVERGED -V /path/to/report

```



This command validates the CONVERGED column of the CSCF_11A_R4B.xml PVL configuration against the schema files contained within CSCF_11A.zip and outputs the validation report to Reports/CSCF_11A_R4B.

29 validate-mim

`validate-mim` is a standalone MIM validation command that can be used to check the syntax of schema files against one of the supported MIM formats. CLI MIM validation applies the same logic used by PDB when importing a schema without adding files to the PDB server.

In addition, this command validates any schematron rules and allows you to export those rules to a designated XML file.

Multiple schema files, directories, ZIP archives, or any combination of the three can be used as input for the tool. A validation report is generated when problems are found. This report groups errors and warnings by the affected files.

29.1 Usage Syntax

```
usage: validate-mim [-h] [-d DIR] [-m FILE] [-s SCHEMATRON] [-o OUTPUT]
validate-mim: error: the following arguments are required: -d DIR
```

29.2 Options

Table 32 validate-mim Options

Option	Description	Notes
<code>-d , --mim-dir <arg></code>	This option is deprecated and has been replaced by <code>-m, --mim-file</code> . Specifies the path of a directory containing input files to validate. Subdirectories on the same path are automatically included.	Optional. Paths can be relative or absolute. This option can be specified multiple times.



Table 32 validate-mim Options

Option	Description	Notes
-m, --mim-file <arg>	Specifies a MIM file, directory, or ZIP archive that constitutes the configuration schema.	Optional. Paths can be relative or absolute. When specifying a directory, subdirectories on the same path are automatically included. Multiple MIM files, directories, or ZIP archives are specified using additional -m options.
-R, --report-file <arg>	Specifies a destination file for the validation report. The output defaults to stdout if this option is not set.	Optional.
-s, --schematron <arg>	Exports schematron rules from the selected schema to the specified XML file.	Optional. Schematron rules can only be exported in XML format.
-t, --mim-type <arg>	Specifies a supported MIM format that is used to validate the selected input files. The possible values are: <ul style="list-style-type: none"> • IMS_CM_MIM_3_0 • IMS_CM_MIM_2_0 • IMS_CM_MIM_1_0 • IMS_CM_MIM_0_1 • IS_CM_MIM • MP_DTD • TSP_MIM 	Optional. validate-mim will automatically detect the MIM type if this option has not been set. For more information on TSP MIM files, refer to CM Management Information Models for TSP based nodes in IMS, 17/1550-HSC 113 06 Uen E in CDM. For more information on IS MIM files, refer to Management Information Modeling in IS, 25/155 19-AZE 101 01/1 Uen E in CDM.
-z, --mim-zip <arg>	This option is deprecated and has been replaced by -m, --mim-file . Specifies the path and filename of a ZIP archive containing input files to validate.	Optional. Paths can be relative or absolute. This option can be specified multiple times.



Note: `validate-mim` requires at least one input (`-m`, `-d`, or `-z`) to function.

29.3 Examples

Note: You must use double quotes `<" ">` to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
### validate-mim -m /path/to/mim -f /path/to/tsp1.0
```

This command validates the contents of the `mims` subfolder against the TSP 1.0 format.

```
### validate-mim -z /path/to/tsp_mims.zip -f /path/to/tsp1.0
```

This command validates the contents of the `tsp_mims.zip` archive against the TSP 0.1 format.

```
### validate-mim -d /path/to/mim -f /path/to/is
```

```
### validate-mim -z /path/to/is.zip -f /path/to/is
```

This command validates files contained in the `mims/new` and `mims/unsorted` folders along with the contents of the `is.zip` archive against the IS format.

30 validate-mpvl

`validate-mpvl` is a standalone MPVL validation command that can be used to validate a configuration file written in PVL format. The validation considers standard XML syntax, format rules established by the official definition of the PVL format, and the limits set by the selected MIM files, including schematron rules. CLI MPVL validation applies the same logic to verify input files as configuration validation in the PDB GUI, and generates the same type of report when problems are found.

For more information on validating node configurations through the GUI including a description of the validation report, refer to the [PDB System Administration Guide \(Reference \[2\]\)](#).

30.1 Usage Syntax

```
### validate-mpvl -f /path/to/pvl
```

```
### validate-mpvl -f /path/to/pvl -s /path/to/schematron
```

```
### validate-mpvl -h
```



30.2 Options

Table 33 validate-mpvl Options

Option	Description	Notes
-c, --configuration-files <arg>	Specifies the path and filename of a MPVL configuration file to validate.	Mandatory. Paths can be relative or absolute. Multiple inputs are specified using additional -c options.
-d, --use-delta	This option instructs the MPVL validator to process values from the delta column in addition to the specified solution type.	Optional.
-m, --mim-files <arg>	Specifies a MIM file, directory, or ZIP archive that constitutes the configuration schema.	Mandatory. Paths can be relative or absolute. When specifying a directory, subdirectories on the same path are automatically included. Multiple MIM files, directories, or ZIP archives are specified using additional -m options.
-M, --moi-format <arg>	Specifies the MOI format of the MPVL configuration. Possible formats include: <ul style="list-style-type: none"> • tsp • is • cba 	Mandatory. For more information on the PVL format, refer to the Parameter List Template Description, EAB/FTI-08:0686 Uen.
-R, --report-file <arg>	Specifies a path and filename for the validation report file. The output defaults to stdout if this option is not set.	Optional. Paths can be relative or absolute.



Table 33 validate-mpvl Options

Option	Description	Notes
-s, --solution-type <arg>	Specifies which PVL parameter value set to validate. Possible solution types include: <ul style="list-style-type: none">• FIXED• MOBILE• CONVERGED• STANDALONE• INITIAL_VALUE	Optional. If a solution type is not specified, all solutions types will be validated, including initial values. The INITIALVALUE syntax is deprecated and has been replaced by INITIAL_VALUE.
-v <variables_file>	Specifies the path and filename of a variables list that is used to resolve any parameter value variables in the MPVL configuration.	Optional. Paths can be relative or absolute. The variables list is a user-defined CSV file that adheres to the following format, one variable per line: <name>, <value>, <description>, <usage> Use the create-variables-file CLI command to generate a file that you can populate with values.

30.3 Examples

Note: You must use double quotes "> to surround entries that have spaces.

Sample Command Usage on a Linux Platform

```
...  
...  
...  
...  
...
```

This command validates the CONVERGED column of the CSCF_11A_R4B.xml PVL configuration against the schema files contained within CSCF_11A.zip and output the validation report to Reports/CSCF_11A_R4B.



Reference List

- [1] Parameter Database (PDB) Installation Instructions, 1/1531-CXP 902 0212 Uen
- [2] PDB System Administration Guide, 2/1543-CXP 902 0212 Uen
- [3] PDB User Guide, 2/1553-CXP 902 0212