Oracle® Enterprise Manager

Command Line Interface

12*c* Release 1 (12.1.0.2)

E17786-06

September 2012

ORACLE®

Oracle Enterprise Manager Command Line Interface, 12*c* Release 1 (12.1.0.2)

E17786-06

Contents

3   Error Code Reference

Index

Preface

This manual provides a complete verb reference, which duplicates and enhances the command line help, for the Enterprise Manager Command Line Interface (EM CLI). This manual also covers concepts, installation, and provides an error code reference chapter.

Audience

This guide is written for Enterprise Manager adminisrators who want to perform operations remotely or script them. You should already be familiar with Oracle Enterprise Manager.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following books in the Oracle Enterprise Manager documentation set:

- *Oracle Enterprise Manager Cloud Control Basic Installation Guide*

- *Oracle Enterprise Manager Cloud Control Advanced Installation and Configuration Guide*

- *Oracle Enterprise Manager Cloud Control Administrator's Guide*

- *Oracle Enterprise Manager Cloud Control Upgrade Guide*

- *Oracle Enterprise Manager Framework, Host, and Services Metric Reference Manual*

- *Oracle Enterprise Manager Cloud Control Extensibility Programmer's Guide*

- *Oracle Database 2 Day DBA*

The latest versions of this and other Oracle Enterprise Manager documentation can be found at:

http://www.oracle.com/technology/documentation/oem.html

Oracle Enterprise Manager also provides extensive online help. Click **Help** on any Oracle Enterprise Manager page to display the online Help system.

Printed documentation is available for sale in the Oracle Store at

http://oraclestore.oracle.com/

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

http://otn.oracle.com/membership/

If you already have a user name and password for OTN, then you can go directly to the documentation section of the OTN Web site at

http://otn.oracle.com/documentation/

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Command Line Interface Concepts and Installation

This chapter discusses the following Enterprise Manager Command Line Interface (EM CLI) topics:

- Overview
- How the EM CLI Works
- EM CLI Quick Start
- Security and Authentication
- Script Availability for Output Data Verbs

1.1 Overview

The Enterprise Manager Command Line Interface (EM CLI) enables you to access Enterprise Manager Cloud Control functionality from text-based consoles (shells and command windows) for a variety of operating systems. You can call Enterprise Manager functionality using custom scripts, such as SQL*Plus, OS shell, Perl, or Tcl, thus easily integrating Enterprise Manager functionality with a company's business process.

Using EM CLI, you can perform Enterprise Manager Cloud Control console-based operations, such as monitoring and managing targets, jobs, groups, and blackouts. EM CLI is intended for use by enterprise or system administrators writing scripts, such as shell/batch files, Perl, Tcl, or PHP, that provide workflow in the customer's business process. You can also use EM CLI commands interactively from an operating system console.

EM CLI is fully integrated with Enterprise Manager's security and user administration functions, enabling you to carry out operations using EM CLI with the same security and confidentiality as the Enterprise Manager Cloud Control console. For example, you can only see and operate on targets for which you are authorized.

Examples of EM CLI usage are as follows:

- Integrate Enterprise Manager with third-party or custom software through scripting. Actions (such as adding/deleting targets, submitting/deleting jobs, creating/deleting users) that are part of a customer's business model can be performed through scripting.
- Create a new Enterprise Manager administrator account.

1.2 How the EM CLI Works

The EM CLI Client is a Java application that accepts a command as input. The EM CLI Client uses the input command to identify a verb to execute the command. A verb is a Java plug-in extension to the EM CLI Client. A verb services the command with its specific options and posts the results to the standard output stream. Any errors are posted to the error stream. The verb also returns an integer exit value that the EM CLI Client sets as the exit value of the command in the Client's calling environment (the operating system console).

A verb can perform its operations locally, but most of the verbs included with the EM CLI are covered by the remote verb in the EM CLI Client. The remote verb contacts the EM CLI OMS Extension in the Enterprise Manager Oracle Management Service (OMS) Console through HTTP/HTTPS and sends the command line through HTTP to the OMS for processing. The EM CLI OMS Extension is essentially a standard Enterprise Manager console page, and is installed in the OMS just as any other standard console page. As with the EMCLI Client, the EM CLI OMS Extension uses the input command to identify a verb to execute the command. The verb can access the Management Repository or Management Agents through OMS as necessary in processing the command.

The remote verb logs on to the OMS and establishes a session automatically, as necessary, to access the OMS-Side Controller. The remote verb impersonates the Enterprise Manager user that invoked the command from the Client. You need to use the login verb if session timeout occurs because you have not executed any EM CLI commands for x minutes, for example.

Figure 1–1 shows the high-level architecture of EM CLI.

Figure 1–1   EM CLI Architecture
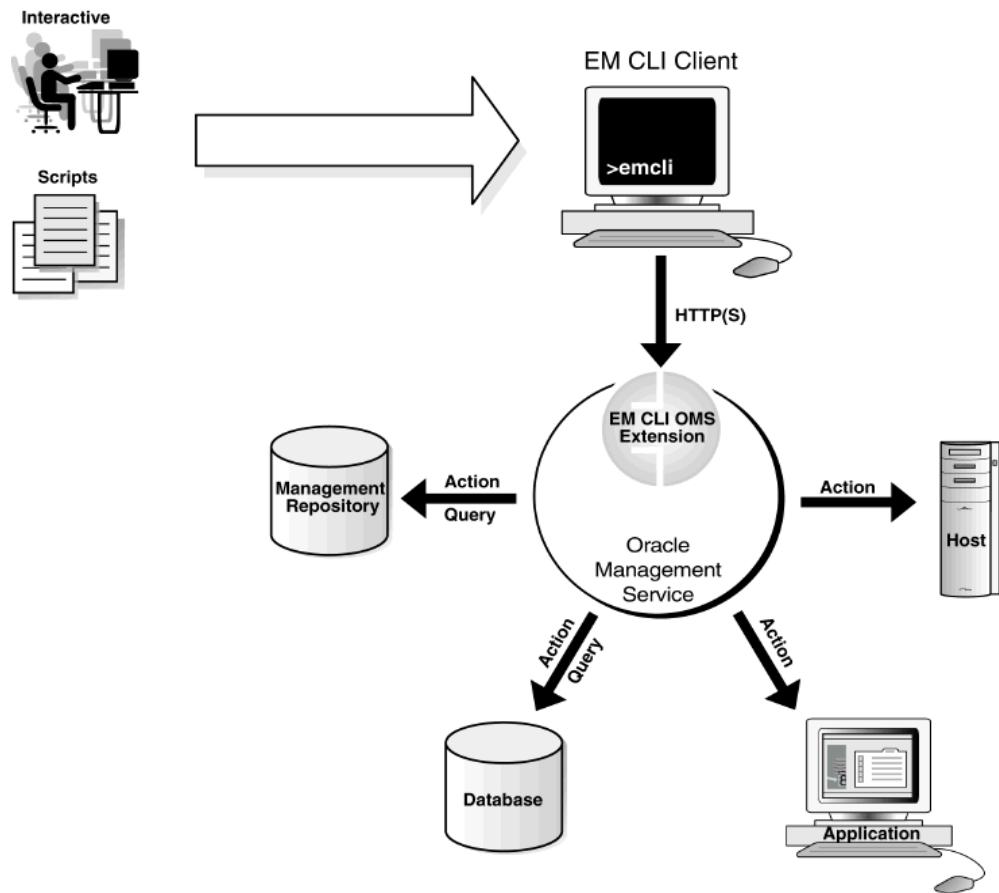
For more information about any of these functional areas, see *Oracle Enterprise Manager Concepts*.

1.3 EM CLI Quick Start

Setting up and running EM CLI is simple. EM CLI consists of two components used to access the Enterprise Manager framework functionality:

- EM CLI Client
- EM CLI Oracle Management Service Extension

You can install the EM CLI Client on any system within your managed network. The EM CLI Client is a command-line program (Java-based) that sends EM CLI verbs to a specific OMS. In some respects, the EM CLI Client functions as a command-line equivalent of an Enterprise Manager Cloud Control console. The EM CLI OMS Extension is automatically installed with the OMS and serves as the communication conduit between the EM CLI Client and the OMS.

1.3.1 Requirements

Before installing EM CLI, you will need the following items:

- Enterprise Manager 12c Cloud Control framework
- Java version 1.6.0_24 or greater

■ Workstation running Solaris, Linux, HPUX, Tru64, AIX, or Windows with NTFS (client installation)

1.3.2 Installation and Setup

As mentioned above, the EM CLI OMS Extension is automatically installed with the OMS. You must install and set up the client portion. The following instructions cover installation and setup procedures for the EM CLI Client.

1.3.2.1 Installing the EM CLI Client

1. Obtain the EM CLI Client kit (emclikit.jar).

   You can download the EM CLI client kit from any 12c Cloud Control installation at the following location:

   ```
   http(s)://oms_host:oms_port/em/console/emcli/download
   ```

   The `emclikit.jar` file is physically located in the `$ORACLE_HOME/sysman/jlib directory` of the 12c Cloud Control OMS home.

2. Set your `JAVA_HOME` environment variable and ensure that it is part of your `PATH`. Make sure that this variable is set to the home of a JDK 1.6.0_24 or greater. For example:

   ```
   setenv JAVA_HOME /usr/local/packages/j2sdk1.6.0_24

   setenv PATH $JAVA_HOME/bin:$PATH
   ```

3. Enter the following command to ensure that you have the correct Java in your PATH:

   ```
   which java
   ```

   This should show the Java in $JAVA_HOME/bin.

4. Install the EM CLI Client. You can install the client portion of EM CLI in any directory either on the same system as the OMS or on any system in your network (download the `emclikit.jar` file to that system).

   Go to the directory where you have installed emclikit.jar:

   ```
   cd $HOME/<your_emcli_installation_directory>
   ```

5. Enter the following command:

   ```
   java -jar emclikit.jar client -install_dir=<emcli_client_dir>
   ```

   The EM CLI client is installed in the <emcli_client_dir>.

After you have installed the EM CLI Client, you are ready to begin setting up the client.

1.3.2.2 Setting Up the EM CLI Client

After the EM CLI Client is installed, you are ready to begin using EM CLI. At this point, you can run the EM CLI Client out of the installation directory location, or alternatively, you can add it to your `PATH`.

Immediately after installation, only basic operational verbs are installed:

- **argfile** — Execute an EM CLI verb where the verb and any arguments are contained in a file.
- **help** — Access command-line help for EM CLI verbs.
- **login** — Log in and establish a session with the OMS.
- **logout** — Log out of EM CLI client from Enterprise Manager.
- **setup** — Configure EM CLI to function with a specific OMS.
- **status** — Show EM CLI setup details
- **sync** — Synchronize the EM CLI Client with an OMS.
- **version** — List EM CLI verb versions or the EM CLI client version.

1.3.2.3 Running Setup

You must run setup to connect the EM CLI Client to the OMS running the EM CLI Management Services. Running the `setup` verb installs all available verb-associated command-line help from the EM CLI Management Service. You can use one EM CLI client installation to function with multiple OMSes. However, at any time, EM CLI can function with a particular OMS. For either scenario, you need to set up the EM CLI client once for each OMS. You also need to subsequently set the EMCLI_STATE_DIR environment variable to the directory that was specified as client directory for the particular OMS.

1. Understand the syntax of the setup verb and its options by entering the following command or referring to the setup verb in the verb reference chapter of this guide:

   ./emcli help setup

2. Enter the setup verb with at least the minimally required parameters as shown in the following example:

   ```
   ./emcli setup -url=http://myworkstation.example.com:em_port/em
     -username=em_user
   ```

   > **Note:** Specify the URL you are using to log in to Entprise Manager through the browser.

   As you observed from step 1, the setup verb has several options, including the following important options:

   - autologin
   - noautologin

   In autologin mode, if a session times out, EM CLI automatically logs you in. In default noautologin mode, if no EM CLI command executes within the 45-minute default sesion timeout period, you need to log in using the `login` verb to be able to execute the verbs.

3. Enter your user password for Enterprise Manager when prompted after the EM CLI client connects with the EM CLI Management Services.

After running the `setup` verb, the message "Emcli Setup Successful" appears, and you are ready to begin using EM CLI.

> **Tip:** For complete information on the setup verb and its options, including autogin, noautologin, and custom_attrib_file referenced in step 2, see the setup verb on page 2-398.
>
> To configure the EM CLI Client to function with multiple Oracle Management Services by implementing multiple setups, see the Examples section for the Setup verb.

1.3.2.4 Using EM CLI Log Files

EM CLI creates log files to record informational and error messages generated during operation. Not all of the logs in the following examples are necessarily present. Logs are created as needed and are appended — they are preserved between invocations of EM CLI. You can safely delete log files any time without affecting the EM CLI operation. The logs contain stack traces, which may not be useful for the casual user, but may benefit you with a high level of system knowledge.

The following examples show possible log file locations:

```
<EM_CLI_Instance_Home>/.emcli.log
<EM_CLI_Instance_Home>/.emcli.log.1
```

`<EM_CLI_Instance_Home>` refers to the directory specified by the `-dir` option in the latest running of the `setup` verb (with an appended `.emcli` sub-directory). The current `<EM_CLI_Instance_Home>` directory can be identified by executing the `status` verb to display the setup summary.

Log files are limited to a maximum of 0.5 MB. EM CLI alternates between the two log files — as each file reaches the 0.5 MB limit, EM CLI begins writing to the other file, overwriting the oldest log file after `emcli.log.1` has been filled for the first time.

The following examples show possible log file locations:

Example 1–1   No configuration directory is specified with the setup verb (Default location)

```
user.home/.emcli/.emcli.log
user.home/.emcli/.emcli.log.1
```

If you do not specify a configuration directory when you run the `setup` verb (`-dir` option is omitted), EM CLI assumes the `.emcli` configuration directory is located within your local home directory. The log files are placed at the root level of the `.emcli` directory. The `.emcli` directory must be local (not mounted remotely).

Example 1–2   Local configuration directory is specified with the setup verb (-dir=<local directory>

```
local.dir/.emcli/.emcli.log
local.dir/.emcli/.emcli.log.1
```

In this example, the configuration directory is specified using the `-dir` option when the `setup` verb is run. This allows you to specify a local configuration directory if the user home directory is mounted remotely (through NFS, for example).

1.3.2.5 Configuring an HTTP Proxy Environment

If you are planning to use EM CLI through an HTTP proxy server, you need to set an additional environment variable (`EMCLI_OPTS`) that supplies EM CLI with the requisite proxy host and port information. The following examples illustrate setting

the `EMCLI_OPTS` environment variable for both Windows and UNIX operating systems.

Example 1–3   Setting EMCLI_OPTS in a Microsoft Windows Environment

```
>set EMCLI_OPTS=-Dhttp.proxyHost=<proxy host> -Dhttp.proxyPort=<proxy port>
```

Example 1–4   Setting EMCLI_OPTS in a UNIX Environment (TCSH)

```
>setenv EMCLI_OPTS "-Dhttp.proxyHost=<proxy host> -Dhttp.proxyPort=<proxy port>"
```

1.3.3 EM CLI Command-line Help

EM CLI incorporates a comprehensive command-line help system that provides various levels of assistance. Available from any EM CLI Client installation, the help system provides a listing of all available verbs, descriptive overviews for each verb, syntax, as well as usage examples. The command-line help is the definitive EM CLI information source.

To access command-line help, enter the following command for an overview of all available verbs ...

./emcli help

... or enter the same command followed by the verb name for a detailed verb description, verb arguments and options, and usage examples, as in:

./emcli help setup

1.4 Security and Authentication

To enable EM CLI to function with a particular OMS, configure EM CLI by executing the `setup` verb. This is a one-time operation for this particular OMS.

Example 1–5   CLI-Enterprise Manager Authentication

```
>emcli setup –url="http[s]://host:port/em" –username="<username>"  [-trustall]
[-novalidate]

>please enter password:
```

You can find out the OMS connection information from any EM CLI Client by invoking the `setup` verb without any options. For example:

```
$ emcli setup
Oracle Enterprise Manager Cloud Control 12c Release 2.
Copyright (c) 1996, 2012 Oracle Corporation and/or its affiliates. All rights
reserved.

Instance Home          : /private/emcli/setup/.emcli
Verb Jars Home         : /private/emcli/setup/.emcli
EM URL                 : https://myomshost.us.oracle.com:5416/em
EM user                : user1
Trust all certificates : true
Auto login             : false
```

You can also invoke the `status` command, which provides more information than the `setup` command:

```
$ emcli status
Oracle Enterprise Manager Cloud Control 12c Release 2.
```

```
Copyright (c) 1996, 2012 Oracle Corporation and/or its affiliates. All rights
reserved.

Instance Home          : /private/emcli/setup/.emcli
Verb Jars Home         : /private/emcli/setup/.emcli
Status                 : Configured
EMCLI Home             : /private/MWHome/oms/bin
EMCLI Version          : 12.1.0.2.0
Java Home              : /private/MWHome/jdk16/jdk
Java Version           : 1.6.0_24
Log file               : /private/emcli/setup/.emcli/.emcli.log
EM URL                 : https://myomshost.us.oracle.com:5416/em
EM user                : sysman
Auto login             : false
Trust all certificates : true
```

1.4.1 HTTPS Trusted Certificate Management

For authenticating an OMS during the SSL server authentication phase of an HTTPS connection handshake, EM CLI searches for trusted certificates in the following key stores:

```
CONFIG_DIR/.emcli/.localkeystore
user.home/.emcli/.keystore
JRE_HOME/lib/security/cacerts
```

CONFIG_DIR is the directory specified by the -dir option in the latest running of the setup verb (with an appended .emcli subdirectory). See Section 1.3.2.4, "Using EM CLI Log Files" for more information about the CONFIG_DIR parameter.

JRE_HOME in a JDK installation is typically JAVA_HOME/jre.

The JDK keytool command can manage the key stores. For more information about this tool, see the security documentation for your Java VM installation, or at the time of this writing:

```
http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/
keytool.html
```

Not all of the key stores in the list above will necessarily be present.

1.4.2 Secure Clients

You can provide credentials to EM CLI in one of two ways:

- Provide credentials at the time of use. See the login and logout verbs for information on credentials.

- Make credentials persistent on the host system where the EM CLI client is running, as might be the case when executing EM CLI verbs from a shell script.

> **Caution:** You should only persist credentials on hosts when the host is a secure client, since the only protection available for credentials is the file-system security of the OS.
>
> Oracle also recommends not using persistent credentials if the EM CLI user's home directory is mounted over NFS or any other insecure file system.

1.4.3 Secure Mode for the EM CLI Setup

The EM CLI client installs certain configuration files and a client-side implementation of verbs on the client system. The EM CLI client configuration files contain information such as the OMS URL, Enterprise Manager user names, and Enterprise Manager passwords.

By default, the EM CLI client is set up in secure mode. In this mode, EM CLI does not store any Enterprise Manager or SSO passwords on the client disk. The command `emcli setup -noautologin ...` sets up the EM CLI client in secure mode. By default, `-noautologin` is true. Therefore, you do not need to specify it if you want to set up the EM CLI client in secure mode. In secure mode, if the EM CLI session times out due to inactivity, explicit login (using the `login` verb) is required before invoking any verb.

If you want to set up EM CLI in the unsecure auto-login mode, you can use the `emcli setup -autologin ....` command. In this mode, if an EM CLI session times out due to inactivity, EM CLI automatically re-establishes the session when a verb has to be executed. However, if you explictly logged out by running `emcli logout`, you need to explicitly log in again using `emcli login`.

- For information on the noautologin option, see the setup verb on page 2-398.

- For information on logging in, see the login verb on page 2-291.

- For information on logging out, see the logout verb on page 2-292.

1.5 Script Availability for Output Data Verbs

For easy parsing of verb output by scripts, a `-script` argument is available for all verbs that generate output data. If you use the `-script` argument, all output columns become tab-separated (with non-null values), and all rows become newline-separated. You can override the default column and row separators by using the `-format` argument in place of `-script`.

```
[-script|-format="name:<format type>;column_
separator:<separator_text>;row_separator:<separator_text>"]
```

Supported `-format` arguments are shown in Table 1–1.

Table 1–1   Supported "-format" Arguments

Argument	Explanation
`-format="name:pretty"`	Pretty-print the output. This is the default when both `-script` and `-format` are not specified.
`-format="name:script"`	Identical to just specifying `-script`. Columns are tab-separated, and rows are newline-separated.
`-format="name:script;column_` `separator:<column_sep_string>"`	Causes the verb output to be column-separated by `<column_sep_string>`. Rows are separated by the newline character.
`-format="name:script;row_separator:<row_sep_` `string>"`	Causes the verb output to be row-separated by `<row_sep_string>`. Columns are separated by the tab character.

Table 1–1   (Cont.) Supported "-format" Arguments

Argument	Explanation
`-format="name:script;column_separator:<column_sep_string>;row_separator:<row_sep_string>"`	Causes the verb output to be column-separated by `<column_sep_string>` and row-separated by `<row_sep_string>`.
`-format="name:csv"`	Produces a table with the columns separated by commas and the rows by newlines.

- `-script` is equivalent to `-format="name:script;column_separator:\u0009;row_separator:\u000A"`

- The values for column and row separator are specified as one or more character strings. Any of the characters can be represented by the unicode sequence `\uXXXX` (where `X` is a hex value).

  NOTE: The ASCII character set is represented by `\u00XX`, where `XX` can range from 00 to 7F. For example, the tab character is represented by `\u0009` and the newline character is represented by `\u000A`.

- The `pretty` format type has no attributes.

- In `script` mode, any verb output cells that contain the separator strings are substituted with the unicode values for these strings so that the output does not break any scripts required to parse the output.

- `script` is the only format type for which separators can be specified.

- Separators need not be single characters, and can be specified using both regular characters interspersed with unicode sequences as shown in the following example:

Example 1–6   Complex Separator

Separator Specification: `xxx\u0009xxx\u0009`

This separator appears as `xxx` followed by a tab, followed by `xxx`, followed by another tab.

1.6 Shared EMCLI Binaries with Multiple Instances

Prior to Enterprise Manager version 12.1.0.2, if you set up 'n' EM CLI instances with a particular OMS, the client side verb jars were present in each of these 'n' EM CLI instance directories.

Enterprise Manager version 12.1.0.2 and later enables multiple EM CLI instances to share the client-side verb jars. Set up each of these instances using the following command:

```
emcli setup -url=<> -verb_jars_dir=<>
```

The client-side verb jars are downloaded only the first time, and all instances do not download again, but use them anyway.

2

Verb Reference

This chapter provides a complete listing of all EM CLI verbs in categorical as well as alphabetical order. Each verb provides complete syntax and usage information.

2.1 Verb Categories

This section lists all of the verbs for this release in the following categories:

- Basic Operational Verbs
- Agent Administration Verbs
- Agent Patch Verbs
- Agent Recovery Verbs
- Audit Settings Verbs
- Bare Metal Provisioning Verbs
- BI Publisher Reports Verbs
- Blackout Verbs
- Chargeback Verbs
- Compliance Verbs
- Connector Verbs
- Create Database Job Verbs
- Credential Verbs
- Credential Verbs - Oracle Database
- Database Machine Targets Customer Support Identifier (CSI) Assignment Verbs
- Deployment Procedure Verbs
- Diagchecks Verbs
- Discover and Push to Agents Verbs
- Execute Command Verbs
- Event and Incident Verbs
- Group Verbs
- Installation Verbs
- Job System Verbs

- Licensing Verbs

- Management Services and Repository Verbs

- Masking Verbs

- Metric Collection and Alerts Verbs

- Metric Extension Verbs

- Monitoring Template Verbs

- Notification Verbs

- OMS Configuration Properties

- OMS Plug-in Deployment Verbs

- Patch Verbs

- Ping Subsytem Verbs

- Privilege Delegation Settings Verbs

- Provisioning Verbs

- Reconfig Job Verbs

- Refresh Coherence Verbs

- Redundancy Group Verbs

- Refresh WLS Domain Verbs

- Report Import/Export Verbs

- Secure Communication Verbs

- Self Update Verbs

- Services Verbs

- SiteGuard Verbs

- Software Library Verbs

- System Verbs

- Target Data Verbs

- User-defined Metrics (UDM) Migration Verbs

- Upgrade Database Job Verbs

- User Administration Verbs

- User Session Administration Verbs

Basic Operational Verbs

Note: Only these verbs are available immediately after installation.

argfile
help
login
logout
setup
status
sync
version

Agent Administration Verbs

get_agent_properties
get_agent_property
resecure_agent
restart_agent
secure_agent
set_agent_property
start_agent
stop_agent
unsecure_agent

Agent Patch Verbs

submit_agent_patch

Agent Recovery Verbs

resync_agent

Audit Settings Verbs

disable_audit
enable_audit
show_audit_settings
show_operations_list
update_audit_settings

Bare Metal Provisioning Verbs

bareMetalProvisioning

BI Publisher Reports Verbs

deploy_bipublisher_reports
setup_bipublisher
unregister_bipublisher

Blackout Verbs

create_blackout
delete_blackout
get_blackout_details
get_blackout_reasons
get_blackout_targets
get_blackouts
stop_blackout

Chargeback Verbs

get_metering_data

Compliance Verbs

export_compliance_group
export_compliance_standard_rule
export_standard
import_compliance_object

Connector Verbs

publish_change_request_ccc

update_ticket_status

Create Database Job Verbs

create_database

Credential Verbs

clear_credential
clear_default_pref_credential
clear_monitoring_credential
clear_preferred_credential
create_credential_set
create_named_credential
delete_credential_set
delete_named_credential
get_credtype_metadata
get_duplicate_credential
get_named_credential
merge_credentials
modify_named_credential
set_credential
set_default_pref_cred
set_monitoring_credential
set_preferred_credential
show_credential_set_info
show_credential_type_info
test_named_credential
update_host_password
update_monitoring_creds_from_agent
update_password
update_target_password

Credential Verbs - Oracle Database

update_db_password

Database Machine Targets Customer Support Identifier (CSI) Assignment Verbs

assign_csi_for_dbmachine_targets

Deployment Procedure Verbs

confirm_instance
delete_instance
describe_procedure_input
get_executions
get_instance_data
get_instance_status
get_instances
get_procedure_types
get_procedure_xml
get_procedures
get_retry_arguments
ignore_instance
reschedule_instance
resume_instance
retry_instance
save_procedure_input

stop_instance
submit_procedure
suspend_instance
update_and_retry_step
update_procedure_input

Diagchecks Verbs

apply_diagcheck_exclude
define_diagcheck_exclude
diagchecks_deploy_status
diagchecks_deploy_tglist
list_diagcheck_exclusions
list_diagchecks
undeploy_diagchecks
update_diagchecks

Discover and Push to Agents Verbs

discover_coherence
discover_fa
discover_wls

Execute Command Verbs

execute_hostcmd
execute_sql

Event and Incident Verbs

create_resolution_state
delete_resolution_state
get_resolution_states
modify_resolution_state
publish_event

Group Verbs

create_group
delete_group
get_group_members
get_groups
modify_group

Installation Verbs

get_agentimage
get_agentimage_rpm
get_supported_platforms

Job System Verbs

create_job
create_library_job
delete_job
delete_library_job
describe_job
describe_job_type
describe_library_job
get_job_execution_detail

get_jobs
get_job_types
retry_job
stop_job

Licensing Verbs

grant_license_no_validation
grant_license_with_validation
revoke_license_no_validation
revoke_license_with_validation

Management Services and Repository Verbs

loader_perf

Masking Verbs

export_masking_definition
generate_masking_script
import_masking_definition
list_masking_definitions
reassoc_masking_definition
save_masking_script
submit_masking_job

Metric Collection and Alerts Verbs

clear_stateless_alerts
collect_metric
get_metrics_for_stateless_alerts
get_on _demand_metrics
get_unsync_alerts
metric_control
sync_alerts

Metric Extension Verbs

export_metric_extension
get_unused_metric_extensions
import_metric_extension
publish_metric_extension
save_metric_extension_draft

Monitoring Template Verbs

apply_template
export_template
import_template
list_templates
modify_collection_schedule

Notification Verbs

subscribeto_rule

OMS Configuration Properties

get_oms_config_property
get_oms_logging_property
list_oms_config_properties

list_oms_logging_properties
list_oms_logging_properties
list_trace
set_logging_property
set_oms_property
trace

OMS Plug-in Deployment Verbs

deploy_plugin_on_agent
deploy_plugin_on_server
get_ext_dev_kit
get_plugin_deployment_status
list_plugins_on_agent
undeploy_plugin_from_agent
undeploy_plugin_from_server

Patch Verbs

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Ping Subsytem Verbs

set_reverse_ping_interval

Privilege Delegation Settings Verbs

apply_privilege_delegation_setting
clear_privilege_delegation_setting
create_privilege_delegation_setting
delete_privilege_delegation_settings
list_privilege_delegation_settings
list_target_privilege_delegation_settings

Provisioning Verbs

provision

Reconfig Job Verbs

convert_to_cluster_database

Refresh Coherence Verbs

refresh_coherence

Redundancy Group Verbs

create_red_group
create_redundancy_group
modify_red_group
modify_redundancy_group
view_redundancy_group

Refresh WLS Domain Verbs

refresh_wls

Report Import/Export Verbs

export_report
get_reports
import_report

Secure Communication Verbs

get_ca_info
secure_agents

Self Update Verbs

export_update
import_update
import_update_catalog
verify_updates

Services Verbs

add_beacon
apply_template_tests
assign_test_to_target
change_service_system_assoc
compare_sla
create_aggregate_service
create_service
delete_metric_promotion
delete_sla
delete_test
delete_test_threshold
disable_sla
disable_test
download_ats_test_databank_file
download_ats_test_zip
enable_sla
enable_test
export_sla
extract_template_tests
get_aggregate_service_info
get_aggregate_service_members
get_test_thresholds
import_sla
list_sla
modify_aggregate_service
remove_beacon
remove_service_system_assoc

run_avail_diag
run_promoted_metric_diag
set_availability
set_key_beacons_tests
set_metric_promotion
set_properties
set_test_threshold
sync_beacon
upload_ats_test_databank_file

SiteGuard Verbs

add_siteguard_script_hosts
create_operation_plan
create_siteguard_configuration
create_siteguard_credential_association
create_siteguard_script
delete_operation_plan
delete_siteguard_configuration
delete_siteguard_credential_association
delete_siteguard_script
delete_siteguard_script_hosts
get_operation_plan_details
get_operation_plans
get_siteguard_configuration
get_siteguard_credential_association
get_siteguard_script_hosts
get_siteguard_scripts
run_prechecks
submit_operation_plan
update_operation_plan
update_siteguard_configuration
update_siteguard_credential_association
update_siteguard_script

Software Library Verbs

add_swlib_storage_location
create_swlib_entity
create_swlib_folder
list_swlib_entities
list_swlib_entity_subtypes
list_swlib_entity_types
list_swlib_folders
list_swlib_storage_locations
refer_swlib_entity_files
reimport_swlib_metadata
remove_swlib_storage_location
update_swlib_entity
upload_swlib_entity_files

System Verbs

create_system
delete_system
get_system_members
modify_system

Target Data Verbs

add_target
add_target_property
change_target_owner
delete_target
get_target_properties
get_targets
list_target_property_names
migrate_to_lifecycle_status
modify_lifecycle_stage_name
modify_target
relocate_targets
remove_target_property
rename_target
set_standby_agent
set_target_property_value

User-defined Metrics (UDM) Migration Verbs

abort_udmmig_session
analyze_unconverted_udms
create_udmmig_session
list_unconverted_udms
udmmig_list_matches
udmmig_request_udmdelete
udmmig_retry_deploys
udmmig_session_details
udmmig_submit_metricpicks
udmmig_summary
udmmig_update_incrules

Upgrade Database Job Verbs

upgrade_database

User Administration Verbs

create_role
create_user
delete_role
delete_user
get_supported_privileges
grant_privs
grant_roles
modify_role
modify_user
revoke_privs
revoke_roles

User Session Administration Verbs

list_active_sessions

EM CLI Verbs

The following sections provide descriptions, formats, and options for all EM CLI verbs. Some of the verbs also contain one or more examples.

abort_udmmig_session

Aborts the migration of user-defined metrics (UDMs) to metric extensions in a session.

Format

```
emcli abort_udmmig_session
      -session_id=<sessionId>
      [-input_file=specific_tasks:<complete_path_to_file>]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Points at a file name that contains a target UDM, one per line in the following format:

  ```
  <targetType>,<targetName>,<collection name>
  ```

  Use targetType=Template to indicate a template. Use * for the collection name to abort all UDMs for a target. The input file should be in UTF-8 format.

Examples

Example 1
The following example aborts the specified migration session. The UDM is returned to the unconverted list.

```
emcli abort_udmmig_session -session_id=<sessionId>
```

Example 2
The following example partially aborts the migration session by removing the specified UDMs from the session.

```
emcli abort_udmmig_session -session_id=<sessionId> -input_file=specific_
tasks:<complete file path>
```

add_beacon

Adds a beacon to the monitoring set of beacons. All enabled tests are pushed to the beacon.

Format

```
emcli add_beacon
      -name=target_name
      -type=target_type
      -bcnName=beacon_name
      [-dontSetKey]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **bcnName**

  Beacon name to add.

- **dontSetKey**

  Indicates the added beacon is not automatically a key beacon. Only use this option if you do not want the beacon to participate in the availability calculation of the service and tests.

Example

The following example adds MyBeacon as a key beacon to the MyTarget service target of type generic_service.

```
emcli add_beacon -name='MyTarget' -type='generic_service'
      -bcnName='MyBeacon'
```

add_siteguard_script_hosts

Adds a host to the Site Guard configuration scripts.

Format

```
emcli add_siteguard_script_hosts
    -script_id=<script_id>
    -host_name=<name1;name2;...>
```

Parameters

- **script_id**

  ID associated with the script.

- **host_name**

  Name of the host where this script will be run. You can specify more than one host name.

Examples

```
emcli add_siteguard_script_hosts
    -script_id="10"
    -host_name ="host1.domain.com"
```

See Also

create_siteguard_script

get_siteguard_script_hosts

add_swlib_storage_location

Adds a storage location in the software library.

Format

```
emcli add_swlib_storage_location
      -name="location_name"
      -path="location_path"
      [-type="OmsShared|OmsAgent|Http|Nfs|ExtAgent"]
      [-host="hostname"]
      [-credential_set_name="setname"] | [-credential_name="name" - credential_
        owner="owner"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the storage location.

- **path**

  Path of the storage location, which can be a file system path or a URL, depending on the storage type chosen.

- **type**

  Type of storage location. The default is OmsShared.

- **host**

  Target name of the host where the path for the storage location exists. This parameter is required for storage types OmsAgent, Nfs, and ExtAgent. For the Nfs storage type, the host is not required to be a target in Enterprise Manager.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target. This is a required parameter for storage types OmsAgent and ExtAgent. The set names can be one of the following:

  – HostCredsNormal: Default unprivileged credential set

  – HostCredsPriv: Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. This parameter is required for storage types OmsAgent and ExtAgent. This parameter must be specified together with the credential_owner parameter.

- **credential_owner**

  Owner of a named credential stored in the repository. This parameter is required for storage types OmsAgent and ExtAgent. This parameter must be specified together with the credential_name parameter.

Examples

Example 1

The following example adds an OMS shared file system storage location named myOMSSharedLocation for the path /u01/swlib .

```
emcli add_swlib_storage_location
      -name="myOMSSharedLocation"
      -path="/u01/swlib"
```

Example 2

The following example adds an OMS Agent File system storage location named myOMSAgtLocation for the path /u01/swlib' on host 'fs1.us.acme.com. The named credential MyAcmeCreds owned by ACME_USER is used for reading/writing files from this location.

```
emcli add_swlib_storage_location
      -name="myOMSAgtLocation"
      -path="/u01/swlib"
      -type="OmsAgent"
      -host="fs1.us.acme.com"
      -credential_name="MyAcmeCreds"
      -credential_owner="ACME_USER"
```

add_target

Adds a target to be monitored by Enterprise Manager. The target type specified is checked on the Management Agent for existence and for required properties, such as user name and password for host target types, or log-in credentials for database target types. You must specify any required properties of a target type when adding a new target of this type.

For `oracle_database` target types, you must specify Role with the monitoring credentials. If the Role is Normal, the UserName must be `dbsnmp`. Otherwise, the Role must be SYSDBA, and UserName can be any user with SYSDBA privileges.

> **Note:** You cannot use this verb for composite targets. The verb does not support adding an association between a parent target such as IAS and a child target such as OC4J.

Format

```
emcli add_target
      -name="name"
      -type="type"
      -host="hostname"
      [-properties="pname1:pval1;pname2:pval2;..."]
      [-separator=properties="sep_string"]
      [-subseparator=properties="subsep_string"]
      [-credentials="userpropname:username;pwdpropname:password;..."]
      [-input_file="parameter_tag:file_path"]
      [-display_name="display_name"]
      [-groups="groupname1:grouptype1;groupname2:grouptype2;..."]
      [-timezone_region="gmt_offset"]
      [-monitor_mode="monitor_mode"]
      [-instances="rac_database_instance_target_name1:target_type1;..."]
```

`[ ]` indicates that the parameter is optional

Parameters

- **name**

  Target name. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks.

- **type**

  Target type. Standard target types include: `host`, `oracle_database`, `oracle_apache`, `oracle_listener`, and `oracle_emd`. To see all available target types available for your environment, check the `$AGENT_HOME/sysman/admin/metadata` directory. A metadata file (XML) exists for each target type.

- **host**

  Network name of the system running the Management Agent that is collecting data for this target instance.

- **properties**

  Name-value pair (that is, `prop_name:prop_value`) list of properties for the target instance. The "name"(s) are identified in the target-type metadata definition.

They must appear exactly as they are defined in this file. Metadata files are located in `$AGENT_HOME/sysman/admin/metadata`.

> **Note:** This verb does not support setting global target properties. It is recommended that you use set_target_property_values to set target properties.

- **separator=properties**

  Specify a string delimiter to use between name-value pairs for the value of the -properties . The default separator delimiter is ";".

- **subseparator=properties**

  Specifies a string delimiter to use between the name and value in each name-value pair for the value of the -properties option. The default subseparator delimiter is ":".

- **credentials**

  Monitoring credentials (name-value pairs) for the target instance. The "name"(s) are identified in the target-type metadata definition as credential properties. The credentials must be specified exactly as they are defined in the target's metadata file. Metadata files are located in `$AGENT_HOME/sysman/admin/metadata`.

- **input_file**

  Used in conjunction with the `-credentials` option, this enables you to store specific target monitoring credential values, such as passwords, in a separate file. The `-input_file` specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific monitoring credentials of the `-credentials` option. The tag must not contain colons (:) or semi-colons (;).

- **display_name**

  Target name displayed in the Enterprise Manager Cloud Control console.

- **groups**

  Name-value pair list of the groups to which this target instance belongs. Follows the format of `groupname:grouptype;groupname2:grouptype2`.

- **timezone_region**

  GMT offset for this target instance. (-7 or -04:00 are acceptable formats.)

- **monitor_mode**

  Either 0 or 1. 0 Indicates a single Agent monitored target. 1 indicates that the target is monitored by more than one Agent, with one Agent as the master (collecting data) at any given point in time. Consequently, you should always set this option to 1 for a RAC database, since it is a multi-agent target.

- **instances**

  Name-value pair list of RAC database instances that the RAC database target has.

Examples

Example 1

The following example adds an `oracle_database` target with the name "database." Note how the credentials are specified. The "name"(s) in the name-value pairs come

from the `oracle_database` metadata file. They must appear exactly as they are named in that file. This also applies for the property "name"(s). This example uses the base minimum of required credentials and properties for the database target.

```
emcli add_target
      -name="database"
      -type="oracle_database"
      -host="myhost.us.example.com"
      -credentials="UserName:dbsnmp;password:dbsnmp;Role:Normal"
      -properties="SID:semcli;Port:15091;OracleHome:/oracle;
       MachineName:smpamp-example.com"
      -groups="Group1:group;Group2:group"
```

Example 2

The following example adds an `oracle_database` target with the name "database." This example illustrates the use of the `input_file` to camouflage the credentials. The password is actually in a file named `at_pwd_file`. The `input_file` argument is used to replace `PWD_FILE` with the contents of the `at_pwd_file` in the credentials argument.

```
emcli add_target
      -name="database"
      -type="oracle_database"
      -host="myhost.us.example.com"
      -credentials="UserName:dbsnmp;password:PWD_FILE;Role:Normal"
      -properties="SID:semcli;Port:15091;OracleHome:/oracle;
       MachineName:smpamp-example.com"
      -input_file="PWD_FILE:/emcli_dir/pwdfiles/at_pwd_file"
```

Example 3

The following example illustrates how to add a RAC database with given installed RAC database instances and clusterware. The example adds a `rac_database` target with the name `cluster_database` and the cluster name `newdb_cluster`. A RAC instance is picked up among instances on the given host. This verb should be called after database instances and clusterwares have been installed. `monitor_mode` is set to 1, because a RAC database is a multi-agent target.

```
emcli add_target
      -name="cluster_database"
      -type="rac_database"
      -host="myhost.us.example.com"
      -monitor_mode="1"
      -properties="ServiceName:service.example.com;ClusterName:
       newdb_cluster"
      -instances="database_inst1:oracle_database;database_inst2:
       oracle_database"
```

Example 4

The following example adds an `oracle_listener` target with the name `mylist`. The `LsnrName` is the name of the listener as configured in the `listener.ora` file, and `ListenerOraDir` is the directory containing the `listener.ora` file.

```
emcli add_target
      -name="mylist"
      -type="oracle_listener"
      -host="myhost.example.com"
      -properties="LsnrName:LISTENER;ListenerOraDir:/oracle/lsnr;
       Port:15091;OracleHome:/oracle;Machine:smpamp-sun1.us
```

add_target_property

Adds a new target property for a given target type. All targets of this target type will have this new target property.

Format

```
emcli add_target_property
        -target_type="target_type"
        -property="prop_name"
```

Parameters

- **target_type**

  Target type for which this property needs to be added. To add this property to all existing target types, you can specify a "*" wildcard character.

- **property**

  Name of the property to be created for this target type. Property names are case-sensitive. The property name cannot be the same as the following Oracle-provided target property names (in English):

  Comment, Deployment Type, Line of Business, Location, Contact

Examples

Example 1
The following example adds the Owner Name property for all targets of type oracle_database.

```
emcli add_target_property -target_type="oracle_database" -property="Owner Name"
```

Example 2
The following example adds the Owner property for all target types.

```
emcli add_target_property -target_type="*" -property="Owner"
```

analyze_unconverted_udms

Analyzes UDMs and lists unique UDMs, any possible matches, and templates that can apply these matching metric extensions.

Format

```
emcli analyze_unconverted_udms
      [-session_id=<sessionId>]

[ ] indicates that the parameter is optional
```

Parameters

- **session_id**

  ID of a session to be analyzed. Not specifying a session ID creates an analysis session that contains all unconverted UDMs. You can specify this session ID in future invocations to generate a fresh analysis.

Examples

Example 1
The following example lists matches for all unconverted UDMs in existing metric extensions.

```
emcli analyze_unconverted_udms
```

Example 2
The following example lists matches for all unconverted UDMs in the specified migration session.

```
emcli list_unconverted_udms -session_id=<sessionId>
```

apply_diagcheck_exclude

Applies a diagnostic check exclusion to a set of target instances. You can exclude
certain diagnostic checks by defining an exclusion name. This rule is applied when all
diagnostic checks are evaluated for the particular target type so that the checks
specified in the rule are excluded.

Format

```
emcli apply_diagcheck_exclude
      -target_type="type"
      -exclude_name="name"
      [-target_name="target_name" ]*
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target.

- **exclude_name**

  Name to use for the exclusion. To create the exclude_name, use the define_
  diagcheck_exclude verb.

- **target_name**

  Target names to apply the exclusion to.

apply_privilege_delegation_setting

Activates Sudo or PowerBroker settings for specified targets.

Format

```
emcli apply_privilege_delegation_setting
        -setting_name="setting"
        -target_type="host/composite"
        [-target_names="name1;name2;..."]
        [-input_file="FILE:file_path"]
        [-force="yes/no"]

[ ] indicates that the parameter is optional
```

Parameters

- **setting_name**

  Name of the setting you want to apply.

- **target_names**

  List of target names. The newly submitted setting applies to this list of Enterprise Manager targets.

  – All targets must be of the same type.

  – The target list must not contain more than one element if the element's target type is "group."

  – The group referenced above should have at least one host target.

- **target_type**

  Type of targets to which the setting is applied. Valid target types are "host" or "composite" (group).

- **input_file**

  Path of the file that has target names. This enables you to pass targets in a separate file. The file cannot contain any colons (:) or semi-colons (;).

- **force**

  If yes, the operation continues and ignores any invalid targets. The default is no.

Examples

Example 1

The following example applies a privilege setting named sudo_setting. This setting applies to targets of type host, and it is being applied to host1, host2, and so forth.

```
emcli apply_privilege_delegation_setting
      -setting_name=sudo_setting
      -target_type=host
      -target_names="host1;host2;....."
```

Example 2

The following example applies a privilege setting named sudo_setting. This setting applies to targets of type host, and it is being applied to host1, host2, and so forth.

The `force` flag indicates that the setting is applied to all valid targets, and invalid targets are ignored.

```
emcli apply_privilege_delegation_setting
      -setting_name=sudo_setting
      -target_type=host
      -target_names="host1;host2;....."
      -force=yes
```

Example 3

The following example applies a privilege setting named `sudo_setting`. This setting applies to targets of type `host`, and host names are selected from `/home/jdoe/file.txt` (one host per line). The `force` flag indicates that the setting is applied to all valid targets, and invalid targets are ignored.

```
emcli apply_privilege_delegation_setting
      -setting_name=sudo_setting
      -target_type=host
      -input_file="FILE:/home/jdoe/file.txt"
      -force=yes
```

apply_template

Applies a monitoring template to a list of specified targets. The parameters to the verb can be supplied in any order.

Format

```
emcli apply_template
        -name="template_name"
        -targets="tname1: ttype1;tname2: ttype2;..."
        [-copy_flags="0" or "1" or "2"]
        [-replace_metrics="0" or "1"]
        [-input_file="FILE1:file_name"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Template name as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks.

- **targets**

  The targets should be specified in the following sequence:

  TargetName1:TargetType1;TargetName2:TargetType2

  For example:

  `db1:oracle_database;my db group:composite`

  A semi-colon is the target separator. Ideally, non-composite targets should be of the target type applicable to the template. If not, the template is not applied to the indicated target. For composite targets, the template is applied only to the member targets that belong to the target type for which the template is applicable.

- **copy_flags**

  This applies only for metrics with multiple thresholds.

  '0' indicates: Apply threshold settings for key values common to the template and target.

  '1' indicates: Remove key value threshold settings in the target and replace them with key value threshold settings from the template.

  '2' indicates: Apply threshold settings for all key values defined in the template. The default is '0'.

- **replace_metrics**

  0 indicates that the thresholds of the metrics not included in the template but available in the target will not be changed. This is the default value. 1 indicates that the thresholds of the metrics present in the target, but not in the template, will be set to NULL. That is, such metrics in the target will not be monitored and therefore, no alert will be raised for them.

- **input_file**

  You can use this parameter to specify the location of a file, which contains the credentials to be used for the User Defined Metrics (UDMs) if the template contains any UDMs. file_name actually refers to the name of the file along with the

path of the location, which contains the credentials applicable for the UDMs. For example:

```
emcli apply_template -name="template1"  -targets="mydb1:oracle_database"
  -input_file= "FILE1:/usr/template/apply_udm_credentials.txt"
```

This example applies a monitoring template named "template1" to target mydb1 of type oracle_database, and the credentials needed for the UDMs are accessed from the file "/usr/template/apply_udm_credentials.txt".

The contents of the file apply_udm_credentials.txt should be in one of the following formats:

–   All UDMs use the same credentials for all targets. For example:

```
credListType:all;
usr_name:joe1;passwrd:pass1;
```

–   Each UDM uses its own credentials for all targets. For example:

```
credListType:perUDM;
udm_name:UDM1;usr_name:joe1;passwrd:pass1;
udm_name:UDM2;usr_name:joe2;passwrd:pass2;
```

–   Each UDM uses different credentials for different targets. For example:

```
credListType:perTargetperUDM;
udm_name:UDM1;tgt_name:TNAME1;usr_name:joe1;passwrd:pass1;
udm_name:UDM1;tgt_name:TNAME2;usr_name:joe2;passwrd:pass2;
udm_name:UDM2;tgt_name:TNAME1;usr_name:joe3;passwrd:pass3;
udm_name:UDM2;tgt_name:TNAME2;usr_name:joe4;passwrd:pass4;
```

It is important to specify the "credListType" in every input text file that you specify.

Examples

Example 1

The following example applies a monitoring template named `my_db_template`. This template applies to targets of type `oracle_database`, and it is being applied to `db1` , which is of type `oracle_database`, and `my_db_group`, which is of type `composite`.

For composite targets, the template is only applied to member targets that belong to the target type for which the template is applicable. Since the `copy_flags` is not specified, the default ("Apply threshold settings for monitored objects common to both template and target") is meant.

```
emcli apply_template -name="my_db_template"
        -targets="db1:oracle_database;my_db_group:composite"
```

Example 2

The following example applies a monitoring template named `my_db_template`. This template applies to targets of type `oracle_database` and it is being applied to `db1`, which is of type `oracle_database` and `my_db_group`, which is of type `composite`.

For composite targets, the template is applied only to member targets that belong to the target type for which the template is applicable. In this case, since the `copy_flags` option is specified as 1, the threshold settings on the target will be duplicated.

```
emcli apply_template -name="my_db_template"
          -targets="db1:oracle_database;my_db_group:composite"
          -copy_flags="1"
```

Example 3

The following example applies a monitoring template named my_db_template. This template is applicable to targets of type oracle_database, and it is being applied to db1 of type oracle_database and my_db_group of type composite.

For composite targets, the template is applied only to the member targets that belong to the target type for which the template is applicable. In this case, since the copy_flags option is specified as "2", the threshold settings on the target are duplicated, but the keys present only in the target and not present in the template are retained in the target, and their settings are not affected.

```
emcli apply_template -name="my_db_template"
          -targets="db1:oracle_database;my_db_group:composite"
          -copy_flags="2"
```

Example 4

The following example applies a monitoring template named `my_db_template`. This template applies to targets of type `oracle_database` and it is being applied to `db1`, which is of type `oracle_database` and `my_db_group`, which is of type `composite`.

For composite targets, the template is applied only to member targets that belong to the target type for which the template is applicable. In this case, since the `copy_flags` option is specified as "1", the threshold settings on the target will be duplicated. Furthermore, the credentials needed for the UDMs are present in the file `/usr/vmotamar/db_credentials.txt`.

```
emcli apply_template -name="my_db_template"
          -targets="db1:oracle_database;my_db_group:composite"
          -copy_flags="1" -input_file= "FILE1:/usr/vmotamar/db_credentials.txt"
```

Example 5

The following example applies the monitoring template named my_db_template. This template is applicable to targets of type oracle_database. This command applies this template to two targets: target db1 of type oracle_database and target my_db_group of type composite.

For composite targets, the template is applied only to the member targets that belong to the target type for which the template is applicable. In this case, since the copy_flags option is specified as "1", the template is superimposed on the target. All keys in the template are copied to the target, and any extra keys present in the target are deleted. The credentials needed for the UDMs are present in file /usr/user/db_credentials.txt.

The replace_metrics flag set to 1 denotes that the thresholds of the metrics present in the target, but not in the template, are set to NULL. That is, these metrics in the target are not monitored, and therefore, no alert is raised for them.

```
emcli apply_template -name="my_db_template"
          -targets="db1:oracle_database;my_db_group:composite"
          -copy_flags="1" -replace_metrics="1" -input_file=
            "FILE1:/usr/user/db_credentials.txt"
```

apply_template_tests

Applies the variables and test definitions from the file(s) into a repository target.

Format

```
emcli apply_template_tests
    -targetName=target_name
    -targetType=target_type
    -input_file=template:template_filename
    [-input_file=variables:<variable_filename>]
    [-input_file=atsBundleZip:<ats_bundle_zip_filename>]
    [-useBundleDatabankFile]
    [-useFirstRowValues]
    [-input_file=atsBundleZip:<ats_bundle_zip_filename>]
    [-overwriteExisting=all | none | <test1>:<type1>;<test2>:<type2>;...]
    [-encryption_key=key]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Target name.

- **targetType**

  Target type.

- **input_file=template**

  Name of the input file containing the test definitions.

- **input_file=variables**

  Name of the input file containing the variable definitions. If this attribute is not specified, the variables are extracted from the same file containing the test definitions.

  The variables file format is as follows:

  ```
  <variables xmlns="template">
  <variable name="<name1>" value="<value1>"/>
  <variable name="<name2>" value="<value2>"/>
  ...
  </variables>
  ```

- **input_file=atsBundleZip**

  Name of the ATS bundle zip defined in the template.

- **useBundleDatabankFile**

  If you specify this parameter, the bundle databank files are used.

- **useFirstRowValues**

  If you specify this parameter, the first row values are used.

- **overwriteExisting**

  Specifies which tests should be overwritten in case they already exist on the target. The possible values are:

1. 'none' (default): None of the existing tests on the target will be overwritten.

2. 'all': If a test with the same name exists on the target, it will be overwritten with the test definition specified in the template file.

3. <test1>:<type1>;<test2>:<type2>;...: If any of the tests with names <test1>, <test2>, and so forth exist on the target, they are overwritten with the definition in the template file.

- **encryption_key**

    Optional key to decrypt the file contents. This key should be the same as the one used to encrypt the file.

Examples

You must have the following privileges to perform these examples:

- Operator privilege on the target.

- Operator privilege on all beacons currently monitoring the target. Alternatively, you must have the "use any beacon" privilege.

Example 1

The following example applies the test definitions contained in the file `my_template.xml` into the Generic Service target `my_target`, using the key `my_password` to decrypt the file contents. If tests with names `my_website` or `my_script` exist on the target, they are overwritten by the test definitions in the file.

```
emcli apply_template_tests
    -targetName='my_target' -targetType='generic_service'
    -input_file=template:'my_template.xml' -encryption_key='my_password'
    -overwriteExisting='my_website:HTTP;my_script:OS'
```

Example 2

The following example applies the test definitions contained in file my_template.xml into the Web Application target my_target using the variable values specified in file my_variables.xml. If any tests in the target have the same name as tests specified in the template file, they are overwritten.

```
emcli apply_template_tests
        -targetName='my_target' -targetType='website'
        -input_file=template:'my_template.xml' -input_file=variables:
          'my_variables.xml'
        -overwriteExisting='all'
```

argfile

Executes one or more EM CLI verbs, where both verbs and the associated arguments are contained in an ASCII file. argfile enables you to use verbs with greater flexibility. For example, when specifying a large list of targets to be blacked out (create_ blackout verb), you can use the argfile verb to input the target list from a file.

Multiple EM CLI verb invocations are permitted in this file. You should separate each verb invocation with a new line.

Format

```
emcli argfile <file_name>
      [-delim=<delimiter_string>]
```

Parameters

- **delim**

  String used as a delimiter between two verbs in the argument file. The default delimiter is a newline character.

assign_csi_for_dbmachine_targets

Assigns or updates the Customer Support Identifier (CSI) for all of the associated Exadata, RAC, and database targets for a database machine name.

Format

```
emcli assign_csi_for_dbmachine_targets
      -target_name="database_system_name"
      -csi="customer_support_identifier_value"
      -mos_id="my_oracle_support_ID"
```

Parameters

- target_name

  Name of the database system target.

- csi

  Customer Support Identifier (CSI) to be assigned.

- mos_id

  My Oracle Support (MOS) user ID.

Example

The following example assigns the CSI 1234567 to database system abcdef.company.com.

```
emcli assign_csi_for_dbmachine_targets
      -target_name=abcdef.company.com
      -csi=1234567
      -mos_id=abc@xyz.com
```

assign_test_to_target

Assigns a test-type to a target-type. If a test-type t is assigned to target-type T, all targets of type T can be queried with tests of type t.

Format

```
emcli assign_test_to_target
    -testtype=test-type_to_be_assigned
    -type=target_type
    [-tgtVersion]=version_of_target_type
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **testtype**

  Test-type to be assigned. Should be the internal name; that is, 'HTTP' instead of 'Web Transaction'.

- **type**

  Service target type.

- **tgtVersion**

  Version of the target type. If not specified, the latest version is used.

Examples

The following example assigns test type HTTP to targets of type generic service v2.

```
emcli assign_test_to_target -testtype='HTTP' -type='generic_service'
    -tgtVersion='2.0'
```

bareMetalProvisioning

Assigns a test-type to a target-type. If a test-type t is assigned to target-type T, all targets of type T can be queried with tests of type t.

Format

```
emcli bareMetalProvisioning
        [-input_file="config_properties:input_XML"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Input XML file confirming to the XSD for BMP.

Example

```
emcli bareMetalProvisioning
        [-input_file="config_properties:input XML"]
```

change_service_system_assoc

Changes the system that hosts a given service.

Format

```
emcli change_service_system_assoc
      -name='name'
      -type='type'
      -systemname='system_name'
      -systemtype='system_type'
      -keycomponents='keycomp1name:keycomp1type[;keycomp2name:keycomp2type;...]'

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service name.

- **type**

  Service type.

- **systemname**

  System on which the service resides.

- **systemtype**

  System type.

- **keycomponents**

  Name-type pair (such as `keycomp_name:keycomp_type`) list of key
  components in the system used for the service.

Example

The following example changes the system for a generic service named `my service`
to a generic system named `my system` with specified key components.

```
emcli change_service_system_assoc
      -name='my service' -type='generic_service'
      -systemname='my system' -systemtype='generic_system'
      -keycomponents='database:oracle_database; mytestbeacon:oracle_beacon'
```

change_target_owner

Changes the owner of the target.

Format

```
emcli change_target_owner
        -target="target_name:target_type"
        [-target="target_name:target_type"]
        -owner="current_target_owner_name"
        -new_owner="new_owner_name"

[ ] indicates that the parameter is optional
```

Parameters

- **target**

  Target name and target type to change the owner.

- **owner**

  Name of the existing owner of the target. The default value for this parameter is the currently logged in user.

- **new_owner**

  New owner name of the target.

Example

The following example changes the ownership of two targets from admin to admin2.

```
emcli change_target_owner
        -target="abc.oracle.com:host"
        -target="testDBSystem:oracle_database"
        -owner="admin1"
        -new_owner="admin2"
```

clear_credential

Clears preferred or monitoring credentials for given users.

Format

```
emcli clear_credential
      -target_type="ttype"
      [-target_name="tname"]
      -credential_set="cred_set"
      [-user="user"]
      [-oracle_homes="home1;home2"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target, which must be "host" if you specify the oracle_homes parameter.

- **target_name**

  Name of the target. Omit this option to clear enterprise-preferred credentials. The target name must be the host name if you specify the oracle_homes parameter.

- **credential_set**

  Credential set affected.

- **user**

  Enterprise Manager user whose credentials are affected. If omitted, the current user's credentials are affected. This value is ignored for monitoring credentials.

- **oracle_homes**

  Name of Oracle homes on the target host. Credentials are cleared for all specified homes.

Examples

```
emcli clear_credential
      -target_type=oracle_database
      -target_name=myDB
      -credential_set=DBCredsNormal
      -user=admin1


emcli clear_credential
      -target_type=oracle_database
      -credential_set=DBCredsNormal
      -user=admin1
```

clear_default_pref_credential

Clears the named credential set as the default preferred credential for the user. The named credential is not deleted from the credential store. Only the user preference to use the named credential as the default preferred credential is cleared.

Format

```
emcli clear_default_pref_cred
      -set_name="set_name"
      -target_type="ttype"
```

Parameters

- **set_name**

  Clears the default preferred credential for this credential set.

- **target_type**

  Target type for the credential set.

Examples

The following example clears the default preferred credential set for the host target type for the HostCredsNormal credential set.

```
emcli clear_default_pref_cred
      -set_name=HostCredsNormal
      -target_type=host
```

clear_monitoring_credential

Clears the monitoring credential set for the target.

Format

```
emcli clear_monitoring_credential
      -set_name="set_name"
      -target_name="target_name"
      -target_type="ttype"
```

Parameters

- **set_name**

  Clears the monitoring credential for this credential set.

- **target_name**

  Clears the preferred credential for this target.

- **target_type**

  Target type for the target/credential set.

Examples

The following example clears the monitoring credential set for the target testdb.example.com for the DBCredsMonitoring credential set.

```
emcli clear_monitoring_credential
      -set_name=DBCredsMonitoring
      -target_name=testdb.example.com
      -target_type=oracle_database
```

clear_preferred_credential

Clears the named credential set as the target preferred credential for the user. The named credential is not deleted from the credential store. Only the user preference to use the named credential as the preferred credential is cleared.

Format

```
emcli clear_preferred_credential
      -set_name="set_name"
      -target_name="target_name"
      -target_type="ttype"
```

Parameters

- **set_name**

  Sets the preferred credential for this credential set.

- **target_name**

  Clears the preferred credential for this target.

- **target_type**

  Target type for the target/credential set.

Examples

The following example clears the preferred credential set for the host target test.example.com for the HostCredsNormal credential set.

```
emcli clear_preferred_credential
      -set_name=HostCredsNormal
      -target_name=test.example.com
      -target_type=host
```

clear_privilege_delegation_setting

Clears the privilege delegation setting from a given host or hosts.

Format

```
emcli clear_privilege_delegation_setting
      -host_names="name1;name2;..."
      [-input_file="FILE:file_path"]
      [-force="yes/no"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **host_names**

  Names of the hosts.

- **input_file**

  Path of the file that has the list of hosts. The file should have one host name per line.

- **force**

  If set to yes, invalid and unreachable targets are ignored and the setting is removed from all valid and up targets. If set to no, invalid and down targets raise an error. The default is no.

Examples

Example 1

```
emcli clear_privilege_delegation_setting
      -host_names="host1;host2;....."
```

Example 2

```
emcli clear_privilege_delegation_setting
      -host_names="host1;host2;....."
      -force=yes
```

Example 3

```
emcli clear_privilege_delegation_setting
      -input_file="FILE:/home/user/file.txt"
      -force=yes
```

clear_stateless_alerts

Clears the stateless alerts associated with the specified target. Only a user can clear these stateless alerts; the Enterprise Manager Agent does not automatically clear these alerts. To find the metric internal name associated with a stateless alert, use the get_metrics_for_stateless_alerts verb.

You cannot use this command to clear stateless alerts associated with diagnostic incidents. You can only clear these alerts in the Enterprise Manager console by clearing their associated Incident or Problem.

Format

```
emcli clear_stateless_alerts
        -older_than=number_in_days
        -target_type=target_type
        -target_name=target_name
        [-include_members]
        [-metric_internal_name=target_type_metric:metric_name:metric_column]
        [-unacknowledged_only]
        [-ignore_notifications]
        [-preview]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **older_than**

  Specify the age of the alert in days. (Specify 0 for currently open stateless alerts.)

- **target_type**

  Internal target type identifier, such as host, oracle_database, and emrep.

- **target_name**

  Name of the target.

- **include_members**

  Applicable for composite targets to examine alerts belonging to members as well.

- **metric_internal_name**

  Metric to be cleaned up. Use the get_metrics_for_stateless_alerts verb to see a complete list of supported metrics for a given target type.

- **unacknowledged_only**

  Only clear alerts if they are not acknowledged.

- **ignore_notifications**

  Use this option if you do not want to send notifications for the cleared alerts. This may reduce the notification sub-system load.

- **preview**

  Shows the number of alerts to be cleared on the target(s).

Examples

The following example clears alerts generated from the database alert log over a week old. In this example, no notifications are sent when the alerts are cleared.

```
emcli clear_stateless_alerts -older_than=7 -target_type=oracle_database -tar
get_name=database -metric_internal_name=oracle_database:alertLog:genericErrStack
-ignore_notifications
```

collect_metric

Performs an immediate collection and threshold evaluation of a set of metrics associated with the specified internal metric name. Metric data collection and threshold evaluation occur asynchronously to the EM CLI call.

You typically use this command when you believe you have resolved an open metric alert or error and would like to clear the event by immediately collecting and reevaluating the metric. This command applies to most metrics except server-generated database metrics.

Use the get_on_demand_metrics verb to see a complete list of supported metrics for a given target.

Format

```
emcli collect_metric
      -target_name=name
      -target_type=type
      -metric_name=metric_name | -collection_name=user_defined_metric_name

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Name of the target.

- **target_type**

  Internal target type identifier, such as host, oracle_database, and emrep.

- **metric_name**

  Internal name that represents a set of metrics that are collected together. Use the get_on_demand_metrics verb to see the supported list of metrics for a given target.

- **collection_name**

  Name of the user-defined metric or SQL user-defined metric. This parameter only applies to user-defined metrics and SQL user-defined metrics.

Examples

Example 1

If you want to collect the "CPU Utilization (%)" metric, look for the appropriate metric internal name (which is Load) using the get_on_demand_metrics command, then run the command as follows:

```
emcli collect_metric  -target_type=host  -target_name=hostname.example.com
-metric_name=Load
```

Example 2

The following example immediately collects and evaluates thresholds for the user-defined metric called MyUDM:

```
emcli collect_metric -target_type=host -target_name=hostname.example.com
-collection=MyUDM
```

Example 3

The following example immediately collects and evaluates thresholds for the SQL user-defined metric called MySQLUDM:

```
emcli collect_metric -target_type=oracle_database -target_name=database
-collection=MySQLUDM
```

compare_sla

Compares two SLAs as defined by two XML files. This utility outputs the difference trees as sla1_compare.dif and sla2_compare.dif in the specified directory. You can use a diff utility to diff these two files. Compare two sla.xml's to find out the difference.

Format

```
emcli compare_sla
    -input_file=sla1:'first_xml
    -input_file=sla2:'second_xml'
    [-dir='directory']

[ ] indicates that the parameter is optional
```

Parameters

- **input_file=sla1**

  File name for the first XML file.

- **input_file=sla2**

  File name for the second XML file.

- **dir**

  The default is the current directory. If you need to specify another directory, use this option for the output files sla1_compare.dif and sla2_compare.dif.

Example

The following example compares two SLAs as defined in sla1.xml and sla2.xml, and outputs sla1_compare.dif and sla2_compare.dif in the current directory.

```
emcli compare_sla
        -input_file=sla1:sla1.xml -input_file=sla2:sla2.xml
```

You can use a standard diff tool to diff the files, such as the following example for Linux:

```
diff sla1_compare.dif sla2_compare.dif
```

confirm_instance

Confirms a manual step. An instance cannot be confirmed when its status is suspended, stopped, completed, or completed with an error.

Format

```
emcli confirm_instance
      [-instance=<instance_guid>]
      [exec=<execution_guid>]
      [-name=<execution name>]
      [-owner=<execution owner>]
      -stateguid=<state_guid>
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **stateguid**

  Comma-separated list of state GUIDs.

Examples

```
emcli confirm_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168
```

```
emcli confirm_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid='51F762417C4943DEE040578C4E087168,51F762417C4944DEE040578C4E087168'
```

convert_to_cluster_database

Converts a single-instance database to a Real Application Cluster (RAC) database.

Format

```
emcli convert_to_cluster_database
      -sourceTargetName="Single instance database target to be converted to RAC"
      -sysdbaCreds="Named credentials for SYSDBA user"
      -hostCreds="Named credentials for Host"
      [-newOracleHome="RAC Oracle Home, if moving to differnt home"]
      [-racConfigType="ADMIN_MANAGED | POLICY_MA NAGED"]
      [-nodeList="Comma-separated node names for Admin Managed RAC database"]
      [-serverPoolList="Comma-separated list of server pools for Policy Managed
         database"]
      [-databaseArea="Shared storage location for database files"]
      [-recoveryArea="Shared storage location for recovery files"]
      [-listenerPort="RAC Listener port"]
```

[ ] indicates that the parameter is optional

Parameters

- **sourceTargetName**

  Enterprise Manager target name of the single-instance database to be converted to a RAC database. Database versions 10.2.0.1.0 and above are supported for conversion. The single-instance database target should exist on one of the nodes of the cluster where the RAC database will be created, and the cluster should be an Enterprise Manager target.

- **sysdbaCreds**

  Named database credentials with SYSDBA privileges on the database to be converted to a RAC database.

- **hostCreds**

  Named host credentials of the user who owns the Oracle home installation.

- **newOracleHome**

  RAC Oracle home location of the converted database. You only need to provide this if different from the Oracle home of the single-instance database to be converted.

- **racConfigType**

  RAC configuration type. Valid values are POLICY_MANAGED and ADMIN_MANAGED. POLICY_MANAGED is valid only for database versions 11.2 or higher. The default is ADMIN_MANAGED if not provided.

- **nodeList**

  List of valid node names for an ADMIN_MANAGED RAC database. It should include the node where the single instance database to be converted exists. If not provided, all the nodes in the cluster are used.

- **serverPoolList**

  Comma-separated list of server pool names for a POLICY_MANAGED RAC database. Applicable only for database versions 11.2 or higher.

- **databaseArea**

  New location for data files of the RAC database. This location should be shared across the nodes of the cluster. It can either be a Cluster File System location or an Automatic Storage Management diskgroup. If not specified, the existing database files should already be on shared storage, and files are not moved during RAC conversion.

- **recoveryArea**

  Fast recovery area location of the RAC database. This location should be shared across the nodes of the cluster. It can either be a Cluster File System location or an Automatic Storage Management diskgroup. If not specified, the existing recovery area location should already be on shared storage, and it does not change during RAC conversion.

- **listenerPort**

  Port of the new RAC listener to be created for the new RAC database. If not provided, the existing listener is used. This option is only applicable to 10.2 and 11.1 database versions. For 11.2 or higher database versions, this value is ignored and the RAC database is always registered with the existing listener in the Cloud Infrastructure home.

Examples

Example 1
```
emcli convert_to_cluster_database -sourceTargetName=sidb
 -sysdbaCreds=sysCreds  -hostCreds=hostCreds racConfigType=ADMIN_MANAGED
```

Example 2
```
emcli convert_to_cluster_database -sourceTargetName=sidb
 -sysdbaCreds=sysCreds  -hostCreds=hostCreds racConfigType=POLICY_MANAGED
 -serverPoolList=sp1,sp2 -databaseArea=+DATA -recoveryArea=+RECOVERY
```

Example 3
```
emcli convert_to_cluster_database -sourceTargetName=sidb
 -sysdbaCreds=sysCreds  -hostCreds=hostCreds -nodeList=node1,node2
 -databaseArea=/u01/share/oradata -recoveryArea=/u01/share/fra -listenerPort=1525
```

create_aggregate_service

Defines an aggregate service: name and its sub-services. After the aggregate service is created, you can edit it from the Enterprise Manager Cloud Control console to configure performance and usage metrics to be collected and displayed.

Format

```
emcli create_aggregate_service
    -name="name"
    -type="type"
    -add_sub_services="name1:type1;name2:type2;..."
    -avail_eval_func="function to evaluate availability"
    [-timezone_region="timezone region"]
```

[ ] indicates that the parameter is optionalis optional

Parameters

- **name**

  Aggregate service name.

- **type**

  Aggregate service type.

- **add_sub_services**

  Sub-services list.

- **avail_eval_func**

  PL/SQL function to evaluate the availability of the aggregate service. Use [or | and] for a predefined evaluate helper function.

- **timezone_region**

  Time zone region of the service.

Examples

```
emcli create_aggregate_service -name="My_Name"
    -type="aggregate_service"
    -add_sub_services="sub1:type1;sub2:type2"
    -avail_eval_func="my_pkg.my_eval_func"
    -timezone_region="PST"
```

create_blackout

Creates a scheduled blackout to suspend any data collection activity on one or more monitored targets.

Format

```
emcli create_blackout
      -name="name"
      -add_targets="name1:type1;name2:type2;..."...
      -reason="reason"
      [-description="description"]
      [-jobs_allowed]
      [-propagate_targets]
      -schedule=
         frequency:once|interval|weekly|monthly|yearly];
         duration:[HH...][:mm...];
         [start_time:yy-MM-dd HH:mm];
         [end_time:yy-MM-dd HH:mm];
         [repeat:#m|#h|#d|#w];
         [months:#,#,...];
         [days:#,#,...];
         [tzinfo:specified|target|repository]
         [tzoffset:#|[-][HH][:mm]]
         [tzregion:...]
```

[ ] indicates that the parameter is optional

Constraints on schedule arguments:

```
frequency:once
    requires => duration or end_time
    optional => start_time, tzinfo, tzoffset
frequency:interval
    requires => duration, repeat
    optional => start_time, end_time, tzinfo, tzoffset
frequency:weekly
    requires => duration, days
    optional => start_time, end_time, tzinfo, tzoffset
frequency:monthly
    requires => duration, days
    optional => start_time, end_time, tzinfo, tzoffset
frequency:yearly
    requires => duration, days, months
    optional => start_time, end_time, tzinfo, tzoffset
```

Parameters

- **name**

  Name of the blackout to create.

- **add_targets**

  Targets to add to the blackout, each specified as target_name:target_type. You can specify this parameter more than once.

- **reason**

  Reason for the blackout. If you have SUPER_USER privileges (you are an Enterprise Manager Super Administrator), any text string can be used for the

reason. The reason is added to the list of allowable blackout reasons if it is not already in the list. If you do not have SUPER_USER privileges, you must specify one of the text strings returned by the get_blackout_reasons verb.

- **description**

    Description or comments pertaining to the blackout. The description, limited to 2000 characters, can be any text string.

- **jobs_allowed**

    When you specify this option, jobs are allowed to run against blacked-out targets during the blackout period. If you do not specify this option, jobs scheduled to be run against these targets are not allowed to run during the blackout period. After a blackout has been created, you cannot change the "allowed jobs" from either EM CLI or the Enterprise Manager Cloud Control console.

- **propagate_targets**

    When you specify this option, a blackout for a target of type "host" applies the blackout to all targets on the host, including the Agent. This is equivalent to nodelevel in the emctl command. Regardless of whether you specify this option, a blackout for a target that is a composite or a group applies the blackout to all members of the composite or group.

- **schedule**

    Blackout schedule. Note that the "frequency" argument determines which other arguments are required or optional.

- **schedule=frequency**

    Type of blackout schedule (default is "once").

- **schedule=duration**

    Duration in hours and minutes of the blackout (-1 means indefinite). Hours and minutes each can be up to 6-digits long.

- **schedule=start_time**

    Start date/time of the blackout. The default value is the current date/time. The format of the value is "yy-MM-dd HH:mm", for example: "2003-09-25 18:34"

- **schedule=end_time**

    Last date/time of the blackout. When "frequency" is weekly, monthly, or yearly, only the date portion is used. When "frequency" is interval or once, the date and time are taken into account. The format of the value is "yy-MM-dd HH:mm"; for example: "2003-09-25 18:34"

- **schedule=repeat**

    Time between successive start times of the blackout. The letter following the number value represents the time units: "m" is minutes, "h" is hours, "d" is days, and "w" is weeks.

- **schedule=months**

    List of integer month values in the range 1-12. Each value must have a corresponding "day" value to fully specify (month, day) pairs that indicate the blackout starting days of the year.

- **schedule=days**

When "frequency" is weekly, this is a list of integer day-of-week values in the range 1-7 (1 is Sunday). When "frequency" is monthly, this is a list of integer day-of-month values in the range 1-31 or -1 (last day of the month). When "frequency" is yearly, this is a list of integer day-of-month values in the range 1-31 or -1 (last day of the month); in this case, the month is taken as the corresponding "month" value for each (month, day) pair.

- **schedule=tzinfo**

  Type of timezone. The `tzinfo` argument is used in conjunction with `tzoffset`. Available timezone types are: "specified" (offset between GMT and the target timezone), "target" (timezone of the specified target), and "repository" (repository timezone -- default setting when tzinfo is not specified). See `-schedule=tzoffset` for more information.

- **schedule=tzoffset**

  Value of the timezone. When the `tzinfo` argument is not specified or is "repository", the timezone value is the repository timezone. In this case, the `tzoffset` argument must not be specified. Otherwise, the `tzoffset` argument is required. When `tzinfo` is set to "specified", the `tzoffset` argument specifies the offset in hours and minutes between GMT and the timezone. When `tzinfo` is set to "target", the `tzoffset` argument specifies an integer index (the first is 1) into the list of targets passed as arguments. For example, for a `tzoffset` setting of 1, the timezone of the first target specified in the `-add_targets` parameter is used.

  Note that the timezone is applied to the start time and the end time of the blackout periods. The timezones associated with each target are not taken into account when scheduling the blackout periods (except that when `tzinfo` is set to "target", the specified target's timezone is used for the blackout times).

- **schedule=[tzregion:<...>]**

  Time zone region to use. When you "specify" the tzinfo parameter, this parameter determines which timezone to use for the blackout schedule. Otherwise, it is ignored. It defaults to "GMT".

Examples

Example 1
The following example creates blackout `b1` for the specified target (`database2`) to start immediately and last for 30 minutes.

```
emcli create_blackout -name=b1 -add_targets=database2:oracle_database
     -schedule="duration::30"
     -reason="good reason1"
```

Example 2
The following example creates blackout `b1` for all targets on `myhost` to start immediately and last until 2007-04-26 05:00 (in the timezone America/New_York).

```
emcli create_blackout -name=b1 -add_targets=myhost:host
     -propagate_targets -jobs_allowed
     -schedule="end_time:2007-04-26 05:00;tzinfo:specified;
       tzregion:America/New_York"
     -reason="good reason2"
```

Example 3

The following example creates blackout `b1` for all targets in group `mygroup` to start immediately and last until 2007-04-26 05:00 (in the timezone America/New_York). No jobs are allowed to run during the blackout.

```
emcli create_blackout -name=b1 -add_targets=mygroup:group
     -schedule="end_time:2007-04-26 05:00;tzinfo:specified;
       tzregion:America/New_York"
     -reason="good reason3"
```

Example 4

The following example creates blackout `b1` for the specified targets (`database2` and `database3`) to start at 2007-08-24 22:30 and last for 30 minutes. The timezone is the timezone for the `database2` target.

```
emcli create_blackout -name=b1
     -add_targets="database2:oracle_database;database3:oracle_database
     -schedule="frequency:once;start_time:07-08-24
22:30;duration::30;tzinfo:target:tzoffset:1"
     -reason="good reason4"
```

Example 5

The following example creates blackout `b1` for the specified targets (`database2` and `database3`) to start at 2007-08-24 22:30 and last for 30 minutes. The timezone is the timezone for the `database3` target.

```
emcli create_blackout -name=b1 -add_targets=database2:oracle_database
     -add_targets=database3:oracle_database
     -schedule="frequency:once;start_time:07-08-24
22:30;duration::30;tzinfo:target;tzoffset:2"
     -reason="good reason5"
```

Example 6

The following example creates blackout `b2` for the specified target (`database2`) to start at 2007-08-25 03:00 and every day thereafter, and to last 2 hours each time. The timezone is the repository timezone.

```
emcli create_blackout -name=b2 -add_targets=database2:oracle_database
     -schedule="frequency:interval;start_time:2007-08-25
03:00;duration:2;repeat=1d"
     -reason="good reason"
```

Example 7

The following example creates blackout `b2` for the specified target (`database2`) to start immediately and every 2 days thereafter (until 06-12-31 23:59), and to last 2 hours 5 minutes each time. The timezone is the repository timezone.

```
emcli create_blackout -name=b2 -add_targets=database2:oracle_database
     -schedule="frequency:interval;duration:2:5;end_time:06-12-31
23:59;repeat=2d;tzinfo:repository"
     -reason="another good reason"
```

Example 8

The following example creates blackout `b4` for all targets on `myhost` and `otherhost` to start every Sunday through Thursday at the current time. The blackout will last 1 hour each time.

```
emcli create_blackout -name=b4 -add_targets="myhost:host;otherhost:host"
```

create_blackout

```
      -propagate_targets
      -schedule="frequency:weekly;duration:1:;days:1,2,3,4,5"
      -reason="very good reason"
```

Example 9

The following example creates blackout b5 for all targets within group mygroup to start on the 15th and last day of each month at time 22:30 and last until 2011-11-24 (2011-11-15 will be the actual last blackout date). The blackout will last 1 hour 10 minutes each time. Jobs are allowed to run during the blackouts.

```
emcli create_blackout -name=b5 -add_targets=mygroup:group
      -propagate_targets -jobs_allowed
      -schedule="frequency:monthly;duration:1:10;start_time:06-10-24 22:30;
       end_  time:06-12-24 23:59:days:15,-1"
      -reason="pretty good reason"
```

Example 10

The following example creates blackout b6 for the specified target (database2) to start at 13:30 on the following dates of each year: 03-02, 04-22, 09-23. The blackout will last 2 hours each time. Jobs are not allowed to run during the blackouts.

```
emcli create_blackout -name=b6 -add_targets=database2:oracle_database
      -propagate_targets
      -schedule="frequency:yearly;duration:2;start_time:07-08-24
13:30:months=3,4,9;days:2,22,23"
      -reason="most excellent reason"
```

create_credential_set

Creates a new credential set. Only Enterprise Manager Super Administrators can create new credential sets.

Format

```
emcli create_credential_set
      -set_name="set_name"
      -target_type="ttype"
      -supported_cred_types="supported_cred_types"
      -monitoring
      [-auth_target_type = "authenticating_target_type"
      [-description ="description]"
```

[ ] indicates that the parameter is optional

Parameters

- **set_name**

  Credential set name to be created.

- **target_type**

  Target type of the new credential set.

- **supported_cred_types**

  Credential types supported by this credential set. You can list the available credential types by using the command show_credential_type_info.

- **monitoring**

  Creates a monitoring credential set.

- **auth_target_type**

  Target type for the supported cred types. The default value is target_type.

- **description**

  Description of the credential set.

Examples

The following example creates a new credential set named New_Credential_Set.

```
emcli create_credential_set
      -set_name=New_Credential_Set
      -target_type=host
      -supported_cred_types=HostCreds;HostSSHCreds
      -description="Example credential set"
```

create_database

Creates a database.

Format

```
emcli create_database
      [-dbType="type_of_database"]
      [-hostTargets="list_of_host_targets"]
      [-cluster="cluster_target_name"]
      -oracleHome="Oracle_Home_location"
      -gdbName="global_database_name"
      -templateName="fully_qualified_path_for_template"
      -hostCreds="named_credential_for_OS_user"
      -sysCreds="named_credential_for_SYS_user"
      -systemCreds="named_credential_for_SYSTEM_user"
      -dbsnmpCreds="named_credential_for_DBSNMP_user"
      [-sid="database_system_identifier"]
      [-racConfigType="RAC_configuration_type"]
      [-nodeList="comma-separated_node_names"]
      [-serverPoolList="comma-separated_list_of_server_pools"]
      [-newServerPool="new_server_pool_name_and_cardinality"]
      [-racOneServiceName="service_name_for_RAC_one-node_database"]
      [-templateInSwlib="TRUE|FALSE"]
      [-templateStageLocation="temporary_directory_on_agent_side"]
      [-storageType="FS|ASM"]
      [-dataFileLocation="Location_of_data_files "]
      [-recoveryAreaLocation="Fast_Recovery_Area_location "]
      [-enableArchiving]
      [-useOMF]
      [-listeners="comma-separated_list_of_listeners_database"]
      [-newListener="new_listener_and_port"]
```

[ ] indicates that the parameter is optional

Parameters

- **dbType**

  Type of database that needs to be created. Valid values are:

  - SINGLE_INSTANCE —To create a database on one particular host or a list of hosts.

  - RAC — To create a cluster database on multiple nodes.

  - RACONE — To create a RAC One-node database.

  RAC and RACONE require the use of the cluster parameter.

- **hostTargets**

  Comma-separated list of host targets where a single-instance database needs to be created. This is a mandatory parameter for a SINGLE_INSTANCE database.

- **cluster**

  Cluster target name for the RAC database on which a cluster needs to be created. The target name should be valid and should have at least one node attached to the target. This is a mandatory parameter for RAC and RACONE databases.

- **oracleHome**

Oracle home of the host targets or cluster target. The Oracle home should be present in all of the targets.

- **gdbName**

  Global database name of the database.

- **templateName**

  Fully-qualified path of the template if the template is located at the Oracle home. Otherwise, provide the display name if the template is present in the software library.

- **hostCreds**

  Named host credentials of the user who owns the Oracle Home installation.

- **sysCreds**

  Named database credentials to be used to create the SYS user.

- **systemCreds**

  Named database credentials to be used to create the SYSTEM user.

- **dbsnmpCreds**

  Named database credentials to be used to create the DBSNMP user.

- **sid**

  Database system identifier., which can be a maximum length of 12 for SINGLE_INSTANCE, 8 otherwise. This should be alphanumeric, with the first character being an alpha character.

- **racConfigType**

  RAC configuration type. Valid values are:

  – POLICY_MANAGED

  – ADMIN_MANAGED

  The default is ADMIN_MANAGED if not provided.

- **nodeList**

  List of valid node names for ADMIN_MANAGED RAC databases. If not provided, all the nodes for the given cluster target are used.

- **serverPoolList**

  Comma-separated list of server pool names for POLICY_MANAGED RAC databases.

- **newServerPool**

  > **Note:** You can either use serverPoolList or newServerPool, but not both. For newServerPool, cardinality is mandatory and should be a positive integer greater than 0.

- **racOneServiceName**

  Service name for the RAC One Node database.

- **templateInSwlib**

Boolean value stating whether the template is from the software library. Valid values are TRUE if the template is from the software library, otherwise FALSE. The default is FALSE if you do not provide this parameter.

- **templateStageLocation**

   Fully-qualified path to where the template should be staged on the host target.

- **storageType**

   Type of storage preferred for the database. Valid values are:

   – FS for File System. This is the default if the parameter is not provided.

   – ASM for Automatic Storage Management.

- **dataFileLocation**

   Location of the data files.

- **recoveryAreaLocation**

   Fast Recovery Area location.

- **enableArchiving**

   Indicates whether archiving of the database is required. Valid values are TRUE if archiving is required, otherwise FALSE. The default is FALSE.

- **useOMF**

   Indicates whether to use Oracle Managed Files.

- **listeners**

   Comma-separated list of listeners (name:port) to register the created database. This is for the SINGLE_INSTANCE database type only, and will be ignored for a RAC database.

- **newListener**

   New listener (name:port) creates a new listener and registers the database. This is for the SINGLE_INSTANCE database type only, and will be ignored for a RAC database.

Examples

Example 1

```
emcli create_database -oracleHome=/u01/app/oracle/product/11.2.0/dbhome_2
-gdbName=testdbee -hostCreds=host_named
                        -sysCreds=sys -systemCreds=system -dbsnmpCreds=dbsnmp
-templateName=/u01/app/oracle/product/11.2.0/
dbhome_2/assistants/dbca/templates/General_Purpose.dbc
                        -dbType=SINGLE_INSTANCE -hostTargets=host1
```

Example 2

```
emcli create_database -oracleHome=/u01/app/oracle/product/11.2.0/dbhome_2
-gdbName=testdbee -hostCreds=host_named
                        -sysCreds=sys -systemCreds=system -dbsnmpCreds=dbsnmp
-templateName=/u01/app/oracle/product/11.2.0/
dbhome_2/assistants/dbca/templates/General_Purpose.dbc
                        -dbType=SINGLE_INSTANCE -hostTargets=host1
-newListener=NEWLSNR:1527
```

Example 3

```
emcli create_database -oracleHome=/u01/app/oracle/product/11.2.0/dbhome_2
-gdbName=testRACCli -hostCreds=cluster_named -sysCreds=sys -systemCreds=system
-dbsnmpCreds=dbsnmp
                        -templateName=/u01/app/oracle/product/11.2.0/
dbhome_2/assistants/dbca/templates/General_Purpose.dbc -dbType=RAC
-cluster=cluster1
                        -dataFileLocation=/u01/share/oradata
-recoveryAreaLocation=/u01/share/fra
```

Example 4

```
emcli create_database -oracleHome=/u01/app/oracle/product/11.2.0/dbhome_2
-gdbName=testdbee -hostCreds=cluster_named
                        -sysCreds=sys -systemCreds=system -dbsnmpCreds=dbsnmp
-templateName=/u01/app/oracle/product/11.2.0/
dbhome_2/assistants/dbca/templates/General_Purpose.dbc
                        -dbType=RAC -cluster=cluster1 -racConfigType=POLICY_MANAGED
-newServerPool=sp1:2
```

create_group

Defines a group name and its members. After you create the group, you can edit it from the Enterprise Manager Cloud Control console to configure Summary Metrics to be displayed for group members.

Format

```
emcli create_group
     -name="name"
     [-type=<group>]
     [-add_targets="name1:type1;name2:type2;..."]...
     [-is_propagating="true/false"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the group.

- **type**

  Group type: group. Defaults to "group".

- **add_targets**

  Add existing targets to the group. Each target is specified as a name-value pair `target_name:target_type`. You can specify this option more than once.

- **is_propagating**

  Flag that indicates whether or not privilege on the group will be propagated to member targets. The default is false.

Examples

Example 1

The following example creates a database-only group named `db_group`. This group consists of two Oracle databases: `emp_rec` and `payroll`.

```
emcli create_group -name=db_group
     -add_targets="emp_rec:oracle_database"
     -add_targets="payroll:oracle_database"
```

Example 2

The following example creates a mixed member-type group named `my_group` that consists of an Oracle database (`database2`), listener (`dblistener`), and host (`mymachine.myco.com`).

```
emcli create_group -name=my_group
     -add_targets="database2:oracle_database;dblistener:oracle_listener
     -add_targets="mymachine.myco.com:host"
```

Example 3

The following example creates a host-only group named `my_hosts` that consists of three systems within the `example.com` domain: `smpsun`, `dlsun`, and `supersun`.

```
emcli create_group -name=my_hosts
      -add_targets="smpsun.example.com:host"
      -add_targets="dlsun.example.com:host;supersun.example.com:host"
```

create_job

Creates and schedules a job.

EM CLI has only been certified for submitting OS Script and SQL Script jobs. EM CLI does not allow OS Script jobs to be run against database targets. The Enterprise Manager Cloud Control console, however, does allow this.

Format

```
emcli create_job
    -name=<job_name>
    -job_type=<job_type>
    -input_file="property_file:<filename>"
```

Parameters

- **name**

  Name of the job.

- **job_type**

  Name of the job type. You can obtain a template property file for the job type by using the describe_job_type verb.

- **input_file**

  Provide the file name to load the properties for creating and scheduling the job. The property file must be accessible to the EM CLI client for reading. Another job of the same job type could also be used to generate the property file using the EM CLI verb describe_job.

Example

The following example creates and schedules a job with name MYJOB1 and job type MyJobType1 with the property file present at location /tmp/myjob1_prop.txt .

```
emcli create_job -name=MYJOB1 -job_type=MyJobType1 -input_file="property_
file:/tmp/myjob1_prop.txt"
```

create_library_job

Creates a library job.

Format

```
emcli create_library_job
    -name=<job_name>
    -job_type=<job_type>
    -input_file="property_file:<filename>"
```

Parameters

- **name**

  Name of the job.

- **job_type**

  Name of the job type. You can obtain a template property file for the job type by using the describe_job_type verb.

- **input_file**

  Provide the file name to load the properties for creating the library job. The property file must be accessible to the EM CLI client for reading. Another library job of the same job type could also be used to generate the property file using the EM CLI verb describe_library_job.

Example

The following example creates a library job with the name MYLIBJOB1 and job type MyJobType1 with the property file present at location /tmp/myjob1_prop.txt .

```
emcli create_library_job -name=MYLIBJOB1 -job_type=MyJobType1
-input_file="property_file:/tmp/myjob1_prop.txt"
```

create_named_credential

Creates a named credential. You can provide input parameters using command line arguments or an input properties file. It also supports the input_file tag for passwords and parameter values.

Format

```
emcli create_named_credential
        -cred_name=<name>
        -auth_target_type=<authenticating_target_type>
        -cred_type=<credential_type>
        -cred_scope=<credential_scope>
        -cred_desc=<credential_description>
        -target_name=<target_name>
        -target_type=<target_type>
        -test
        -test_target_name=<test_target_name>
        -test_target_type=<test_target_type>
        -input_file=<tag:value>
        -input_bfile=<tag:value>
        -properties_file=<filename>
        -attributes=<p1:v1;p2:v2;...>
```

Parameters

- **cred_name**

  Credential name, such as MyBackUpCreds. This is required if you do not use properties_file.

- **auth_target_type**

  Authenticating target type (e.g. host). This is required if you do not use properties_file.

- **cred_type**

  Credential type. This is required if you do not use properties_file.

- **cred_scope**

  Possible values are global | instance. The default is global.

- **cred_desc**

  Credential description.

- **target_name**

  This is required when cred_scope is instance.

- **target_type**

  This is required when cred_scope is instance.

- **test**

  Use this to test the credential before saving.

- **test_target_name**

  Use this to supply the target name to test a global credential. This is required when cred_scope is global and the test parameter is used.

■ **test_target_type**

Use this to supply the target type to test a global credential. This is required when cred_scope is global and the test parameter is used.

■ **input_file**

Use this to supply sensitive property values from the file.

■ **input_bfile**

Use this to supply binary property values from the file.

■ **properties_file**

Use this to pass all parameters from the file. Values given on the command line take precedence.

■ **attributes**

Specify credential columns as follows:

```
colname:colvalue;colname:colvalue
```

You can change the separator value using -separator=attributes=<newvalue>, and you can change the sub-separator value using -subseparator=attributes=<newvalue>.

Examples

Example 1

The following example creates a HostCreds named credential with username foo and password bar:

```
emcli create_named_credential
        -cred_name=NC1
        -auth_target_type=host
        -cred_type=HostCreds
        -attributes="HostUserName:foo;HostPassword:bar"
```

Example 2

The following example creates a privilege delegation credential with user name foo, password bar, privilege delegation type SUDO, and RUNAS user root:

```
emcli create_named_credential
        -cred_name=NC1
        -auth_target_type=host
        -cred_type=HostCreds
        -attributes="HostUserName:foo;HostPassword:bar;PDPTYPE:SUDO;RUNAS:root"
```

To use Powerbroker attributes, the string should be:

```
-attributes="HostUserName:foo;HostPassword:bar;PDPTYPE:POWERBROKER;RUNAS:root;
PROFILE:EMGC
```

Example 3

The following example reads the password from the mypasswordfile.txt file.

```
emcli create_named_credential
        -cred_name=NC1
        -auth_target_type=host
        -cred_type=HostCreds
        -attributes="HostUserName:foo;HostPassword:tag"
```

```
                           -input_file="tag:mypasswordfile.txt"
```

Example 4

The following example prompts for the password from standard input:

```
emcli create_named_credential
      -cred_name=NC1
      -auth_target_type=host
      -cred_type=HostCreds
      -attributes="HostUserName:foo;HostPassword:"
```

Example 5

The following example specifies prop1.txt as a multi-line Java properties file, in which each line contains a parameter=value format. You can provide the password in the same file or not specify it. If not specified, you are prompted for it.

```
emcli create_named_credential
      -properties_file=prop1.txt
```

Example 6

The following example shows output for various credential types.

```
emcli show_credential_type_info -target_type=host
```

```
Target Type   Cred Type Name  Cred Type Column Name   Key Column
host          HostCreds       HostPassword            No
                              HostUserName            Yes
              HostSSHCreds    SSH_PUB_KEY             No
                              SSH_PVT_KEY             No
                              USERNAME                Yes
              ProvisionCreds  InstallPassword         No
                              InstallUserName         Yes
                              OMSRegistrationPassword No
                              ProvCompPasswd          No
              WBEMCreds       WBEMPassword            No
                              WBEMUserName            Yes
```

create_operation_plan

Creates an operational plan for the Oracle Site Guard operation.

Format

```
emcli create_operation_plan
      -primary_system_name="name_of_primary_system"
      -standby_system_name="name_of_standby_system"
      -system_name="name_of_system"
      -operation="name_of_operation"
      -name="name_of_operation_plan"
      -role="role_associated_with_system"
```

Parameters

- **primary_system_name**

  Name of your system associated with the primary site. Enter this parameter for switchover or failover operations.

- **standby_system_name**

  Name of your system associated with the standby site. Enter this parameter for switch-over or fail-over operations.

- **system_name**

  Name of the system. Enter this parameter for start or stop operations.

- **operation**

  The function of the operation. Examples: switchover, failover, start, or stop.

- **name**

  Name of the operation plan.

- **role**

  Role associated with a system when you run an operation (start or stop).

Examples

Example 1

```
emcli create_operation_plan
      -primary_system_name="BISystem1"
      -standby_system_name="BISystem2"
      -operation="switchover"
      -name="BISystem1-switchover-plan"
```

Example 2

```
emcli create_operation_plan
      -system_name="austin"
      -operation="start"
      -name="BISystem1-start-plan"
      -role="Primary"
```

See Also

emcli get_operation_plans and emcli submit_operation_plan

create_patch_plan

Creates a new patch plan with the specified name and the patch-target map.

Format

```
emcli create_patch_plan
        -name="name"
        -input_file=data:"file_path"
        [-impact_other_targets="add_all | add_original_only | cancel"]
        [-problems_assoc_patches="ignore_all_warnings | cancel"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the setting.

- **input_file**

  Input data to create a new patch plan. You must provide the data in the property name-value pairs.

- **impact_other_targets**

  Action to take when other targets are impacted while adding the patches to the plan. Possible values for this parameter are:

  add_all — Add all impacted targets to the plan.
  add_original_only — Only add original targets to the plan.
  cancel — Cancel the plan creation.

- **problems_assoc_patches**

  Action to take when there are problems associating patches to targets. Possible values for this parameter are:

  ignore_all_warnings — Ignore all warnings.
  cancel — Cancel the plan creation.

See Also

delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

Examples

```
emcli create_patch_plan -name="plan_name" -input_file=data:"/tmp/patchplan.props"

emcli create_patch_plan -name="plan name" -input_file=data:"/tmp/patchplan.props"
-impact_other_targets="add_all"

emcli create_patch_plan -name="plan name"  -input_file=data:"/tmp/patchplan.props"
-impact_other_targets="add_all" -problems_assoc_patches="ignore_all_warnings"
```

You can use the following sample input file to create a patch plan with two patches:

```
patch.0.patch_id=4518443
         patch.0.release_id=80102010
         patch.0.platform_id=226
         patch.0.language_id=0
         patch.0.target_name=orclws
         patch.0.target_type=oracle_database
         patch.1.patch_id=4424952
         patch.1.release_id=80102030
         patch.1.platform_id=46
         patch.1.language_id=0
         patch.1.target_name=arac
         patch.1.target_type=rac_database
```

create_privilege_delegation_setting

Creates Sudo or PowerBroker settings to apply later. You must create at least one
setting to use the `apply_privilege_delegation_setting` verb.

Format

```
emcli create_privilege_delegation_setting
       -setting_name="setting_name"
       -setting_type="ttype"
       [-settings="setting"]
       [-separator=settings=";"]
       [-subseparator=settings=","]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **setting_name**

  Name of the setting.

- **setting_type**

  Type of setting you want to create.

- **settings**

  Parameter value. Choose one of the following parameters:

  %USERNAME% — Name of the user running the command.
  %RUNAS% — Run the command as this user.
  %COMMAND% — Sudo command.

  The %USER%, %RUNAS%,, %COMMAND% are tokens that the end-user has to
  use as-is while creating/modifying the privilege delegation settings. The system
  replaces these tokens with the actual values at run time depending on the
  command being run and for which user. Also, %command% should be upper case
  %COMMAND% for 10.2.0.5 GC.

- **separator**

  Delimiter inserted between name-value pairs for the given name. The default
  value is a semi-colon (;).

- **subseparator**

  Separator inserted between the name and value in each name-value pair for the
  given name. The default value is a semi-colon (;).

Examples

Example 1
The following example creates a setting named `sudo_setting`. The setting is of type
SUDO, and the Sudo path used is `/usr/local/bin/sudo`. Sudo arguments are:

```
-S
```
-u
%RUNAS%
%COMMAND%

```
emcli create_privilege_delegation_setting
    -setting_name=sudo_setting
    -setting_type=SUDO
    -settings="SETTINGS:/usr/local/bin/sudo -S -u %RUNAS% %COMMAND%"
```

Example 2

The following example creates a setting named `pb_setting`. The setting is of type POWERBROKER, and the PowerBroker path used is `/etc/pbrun`. Arguments are:

```
%RUNAS%
%PROFILE%
%COMMAND%
;PASSWORD_PROMPT_STRING
Password:
```

```
emcli create_privilege_delegation_setting
    -setting_name=pb_setting
    -setting_type=POWERBROKER
    -settings="SETTINGS,/etc/pbrun %RUNAS% %PROFILE% %COMMAND%
    ;PASSWORD_PROMPT_STRING,Password:"
    -separator=settings=";"
    -subseparator=settings=","
```

For more examples, see the online help.

create_red_group

Defines a redundancy group name and its members. After you create the redundancy group, you can edit it from the Enterprise Manager Cloud Control console to configure charts to be displayed for redundancy group members.

Format

```
emcli create_red_group
    -name="name"
    [-type=<generic_redundancy_group>]
    -add_targets="name1:type1;name2:type2;..."...
    [-owner=<redundancy_group_owner>]
    [-timezone_region=<actual_timezone_region>]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the redundancy group.

- **type**

  Redundancy group type. Defaults to generic_redundancy_group.

- **add_targets**

  Add existing targets to the redundancy group. Each target is specified as a name-value pair target_name:target_type. You can specify this option more than once.

- **owner**

  Owner of the redundancy group.

- **timezone_region**

  Time zone region of this redundancy group.

Example

The following example creates a redundancy group named lsnr_group. This group consists of two Oracle listeners: emp_rec and payroll.

```
emcli create_red_group -name=lsnr_group
    -add_targets="emp_rec:oracle_listener"
    -add_targets="payroll:oracle_listener"
```

create_redundancy_group

Creates a redundancy group.

Format

```
emcli create_redundancy_group
      -redundancyGroupName="redGrpName"
      -memberTargetType="tType"
      -memberTargetNames="tName1;tName2"
      [-group_status_criterion=NUMBER|PERCENTAGE]
      [-group_status_tracked=UP|DOWN]
      [-group_status_value=<group_status_value>]
      [-timezone_region=<valid_time_zone_region>]
      [is_propagating=true|false]
```

`[ ] indicates that the parameter is optional`

Parameters

- **redundancyGroupName**

  Name of the redundancy group.

- **memberTargetType**

  Target type of the constituent member targets.

- **memberTargetNames**

  Member targets for this redundancy group.

- **group_status_criterion**

  This parameter and the next two calculate the status of the Redundancy Group. Consequently, you need to specify all three options together. If this is not to be a capacity group, you need to specify the following combination:

  ```
  -group_status_criterion='NUMBER' -group_status_tracked='UP' -group_status_
  value='1']
  ```

- **group_status_tracked**

  See the parameter above.

- **group_status_value**

  See the group_status_criterion parameter.

  You can specify any value between 1 and 100 if -group_status_criterion= "PERCENTAGE", or any value between 1 and the number of targets present if -group_status_criterion="NUMBER".

- **timezone_region**

  Time zone region of this redundancy group. For a list of valid time zone regions, enter the following command at SQLPLUS:

  ```
  SELECT TZNAME FROM V$TIMEZONE_NAMES
  ```

  You may need to have the SELECT_CATALOG_ROLE role to execute this command.

- **is_propagating**

Indicates whether or not the privilege on the redundancy group will be propagated to member targets. The default value is false.

Examples

Example 1

The following example creates a redundancy group with the name 'redGrp1' and with listener, listener2, listener3 as its member targets. The status is calculated as the redundancy group being up if 55 percent of its member targets are up.

```
emcli create_redundancy_group -redundancyGroupName='redGrp1'
       -memberTargetType='oracle_listener'
       -memberTargetNames='listener;listener2;listener3'
       -group_status_criterion='PERCENTAGE'
       -group_status_tracked='UP'
       -group_status_value='55'
```

Example 2

The following example creates a 'redGrp1' redundancy group with listener, listener2, and listener3 as its member targets and time zone as PST8PDT. The status is calculated as the redundancy group being up if two of its member targets are up.

```
emcli create_redundancy_group -redundancyGroupName='redGrp1'
       -memberTargetType='oracle_listener'
       -memberTargetNames='listener;listener2;listener3'
       -timezone_region='PST8PDT'
       -group_status_criterion='NUMBER'
       -group_status_tracked='UP'
       -group_status_value='2'
```

create_resolution_state

Creates a new resolution state that describes the state of incidents or problems. Only super administrators can execute this command. The new state is always added between the New and Closed states. You need to specify the exact position of this state in the overall list of states by using the position option. The position can be between 2 and 98.

The state is applicable by default to both incidents and problems. You can use the applies_to option to indicate that the state is applicable only to incidents or problems. A success message is reported if the command is successful. An error message is reported if the create fails.

Format

```
emcli create_resolution_state
      -label="label_for_display"
      -position="display_position"
       [-applies_to="INC|PBLM"]
```

`[ ]` indicates that the parameter is optional

Parameters

- **label**

  End-user visible label of the state. The label cannot exceed 32 characters. You can change this later if needed.

- **position**

  Position of this state within the overall list of states. This is used when displaying the list of states in the user interface. The position can be between 2 and 98. You can change the position of the state later if needed.

  It is recommended that you set the position with sufficient gaps to facilitate moving states around. For example, if you set the positions to 5, 10, and 15 instead of 2, 3, and 4, it is easier to move a state from position 15 to 9, for instance, in contrast to the latter scheme, in which you would have to move all states to provide space for the reordering.

- **applies_to**

  Indicates that the state is applicable only for incidents or problems. By default, states apply to both incidents and problems. Supported values are "INC" or "PBLM".

Examples

Example 1
The following example adds a resolution state that applies to both incidents and problems at position 25.

```
emcli create_resolution_state -label="Waiting for Ticket" -position=25
```

Example 2
The following example adds a resolution state that applies to problems only at position 35.

```
emcli create_resolution_state -label="Waiting for SR" -position=35 -applies_
to=PBLM
```

create_role

Creates a new Enterprise Manager admininistrator role.

Format

```
emcli create_role
      -name="role_name"
      [-type="type_of_role"]
      [-description="description"]
      [-roles="role1;role2;..."]
      [-users="user1;user2;..."]
      [-privilege="name[;secure_resource_details]]"
      [-separator=privilege="sep_string"]
      [-subseparator=privilege="subsep_string"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Role name.

- **type**

  Type of role. The default value for this parameter is EM_ROLE. The other possible value is EXTERNAL_ROLE.

- **description**

  Description of the role.

- **roles**

  List of roles to assign to this new role. Currently, the only built-in role is PUBLIC.

- **users**

  List of users to whom this role is assigned.

- **privilege**

  Privilege to grant to this role. You can specify this option more than once. **Note:** Privileges are case-insensitive.

  secure_resource_details should be specified as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
  name2=resource_column_value2]..]"
  ```

- **separator**

  Specify a string delimiter to use between name-value pairs for the value of the privilege option. The default separator delimiter is ";" .

- **subseparator**

  Specify a string delimiter to use between name and value in each name-value pair for the value of the privilege option. The default separator delimiter is ";" .

Examples

Example 1

The following example creates a role named my_new_role with the one-sentence description - "This is a new role called my_new_role". The role combines three existing roles: role1, role2, and role3. The role also has two added privileges: to view the job with ID 923470234ABCDFE23018494753091111 and to view the target host1.example.com:host. The role is granted to johndoe and janedoe.

```
emcli create_role
      -name="my_new_role"
      -desc="This is a new role called my_new_role"
      -roles="role1;role2;role3"
      -privilege="view_job;923470234ABCDFE23018494753091111"
      -privilege="view_target;host1.example.com:host"
      -users="johndoe;janedoe"
```

Example 2

The following example creates a role named my_external_role with a role type of EXTERNAL_ROLE and one-sentence description of "This is an external role."

```
resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
name2=resource_column_value2]..]"
```

create_service

Creates a service to be monitored by Enterprise Manager.

Format

```
emcli create_service
      -name='name'
      -type='type'
      -availType=test|system
      -availOp=and|or
      [-hostName=<host_name>
      [-agentURL=<agent_url>
      [-properties='pname1|pval1;pname2|pval2;...']
      [-timezone_region=<gmt_offset>]
      [-systemname=<system_name>]
      [-systemtype=<system_type>]
      [-keycomponents='keycomp1name:keycomp1type;keycomp2name:keycomp2type;...']
      [-beacons='bcn1name:bcn1isKey;bcn2name:bcn2isKey;...']
      [-input_file='template:Template_file_name;[vars:Variables_file_name]']
      [-sysAvailType=<availability_type>]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service name. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks.

- **type**

  Service type.

- **availType**

  Sets the availability to either test-based or system-based. If availability is set to test, template file, beacons, and variable are required arguments. If availability is set to system, systemname, systemtype, and keycomponents are required.

- **availOp**

  Availability operator. If and, uses all key tests/components to decide availability. If or, uses any key tests/components to decide availability.

- **hostName**

  Network name of the system running the Management Agent that is collecting data for this target instance.

- **agentURL**

  URL of the Management Agent that is collecting data for this target instance. If you enter the host name, the Agent URL of the host is automatically entered in this field.

- **properties**

  Name-value pair (that is, prop_name|prop_value) list of properties for the service instance.

- **timezone_region**

GMT offset for this target instance (-7 or -04:00 are acceptable formats).

- **systemname**

  System name on which service resides.

- **systemtype**

  Type of system for which you want to create the service.

- **keycomponents**

  Name-type pair (that is, `keycomp_name:keycomp_type`) list of key components in the system that are used for the service.

- **beacons**

  Name-isKey pairs that describe the beacons of the service. If isKey is set to `Y`, beacon is set as a key-beacon of the service. The service should have at least one key beacon if the availability is set to test-based.

- **input_file**

  Template file name is the XML file that includes the template definition. Variable file defines the values for the template.

- **sysAvailType**

  Type of availability when the availType is system-based. Sets the availability to either system target directly or selected components of a system.

  If availability is set to 'system target directly,' the system needs to have availability[status] defined. systemname and systemtype are required parameters.

  If availability is set to 'selected components of a system,' systemname, systemtype and keycomponents are required parameters.

  If availability is set to 'system target directly,' and if the system does not have availability[status] defined, the availability set is invalid. Therefore, the only option that can be set is 'selected components of a system'.

Examples

Example 1

The following example creates a generic service named my_service with specified properties on a generic system named my_system. The availability is set as system-based, and the availability is based on system target status.

```
emcli create_service
    -name='my service' -type='generic_service'
    -availType='system' -availOp='or'
    -sysAvailType='system target directly'
    -properties='prop1:value1; prop2:value2'
    -timezone_region='PST8PDT'
    -systemname='my system' -systemtype='generic_system'
```

Example 2

The following example creates a generic service named `my_service` with specified properties on a generic system named `my system` with specified key components. The availability is set as system-based.

```
emcli create_service
    -name='my_service' -type='generic_service'
    -availType='system' -availOp='or'
```

```
-properties='prop1:value1; prop2:value2'
-timezone_region='PST8PDT'
-systemname='my system' -systemtype='generic_system'
-keycomponents='database:oracle_database; mytestbeacon:oracle_beacon'
```

Example 3

The following example creates a generic service named my_service with specified properties with tests defined in mytests.xml, and beacons MyBeacon as the key beacon and MyOtherBeacon as a non-key beacon. Availability is set as test-based.

```
emcli create_service
      -name='my_service' -type='generic_service'
      -availType='test' -availOp='or'
      -properties='prop1:value1; prop2:value2'
      -timezone_region='PST8PDT'
      -input_file='template:mytests.xml'
      -beacons='MyBeacon:Y;MyOtherBeacon:N'
```

create_siteguard_configuration

Creates a site configuration for Site Guard. It associates the systems and their roles.

Format

```
emcli create_siteguard_configuration
      -primary_system_name=<name>
      -standby_system_name=<name1;name2;...>
```

Parameters

- **primary_system_name**

  Name of the system associated with the primary site.

- **standby_system_name**

  Name of the system associated with the standby system. You can specify more than one system name.

Examples

```
emcli create_siteguard_configuration
      -primary_system_name="BISystem1"
      -standby_system_name="BISystem2"
```

See Also

update_siteguard_configuration

delete_siteguard_configuration

create_siteguard_credential_association

Associates the credentials with the targets in a site.

Format

```
emcli create_siteguard_credential_association
      -system_name=<name>
      [-target_name=<name>]
      -credential_type=<type>
      [-credential_name=<name>]
      [-use_preferred_credential=<type>]
      -credential_owner=<owner>
```

```
[ ] indicates that the parameter is optional.
```

Parameters

- **system_name**

  Name of the system.

- **target_name**

  Name of the target.

- **credential_type**

  Type of credential, which can be HostNormal, HostPrivileged, WLSAdmin, or DatabaseSysdba.

- **credential_name**

  Name of the credential. If you do not specify this parameter, you need to specify the use_preferred_credential parameter.

- **use_preferred_credential**

  Name of the credential. If you do not specify this parameter, you need to specify the credential_name parameter.

- **credential_owner**

  Owner of the credential.

Examples

Example 1

```
emcli create_siteguard_credential_association
      -system_name="BISystem1"
      -credential_type="HostNormal"
      -credential_name="HOST-SGCRED"
      -credential_owner="sysman"
```

Example 2

```
emcli create_siteguard_credential_association
      -system_name="BIsystem1"
      -target_name="database-instance"
      -credential_type="HostNormal"
      -credential_name="HOST-DBCRED"
      -credential_owner="sysman"
```

create_siteguard_script

Associates scripts (pre-script, post-script, and storage script) with the Site Guard configuration.

Format

```
emcli create_siteguard_script
      -system_name=<name>
      -operation=<name>
      -script_type=<type>
      [-host_name=[<name1;name2;...>]
      -path=<path_of_script>
      [-all_hosts=true|false]
      [-role=Primary|Standby]

[ ] indicates that the parameter is optional.
```

Parameters

- **system_name**

  Name of the system.

- **operation**

  Name of the operation. Examples: Switchover, Failover, Start, or Stop.

- **script_type**

  Type of script, which can be Mount, UnMount, Pre-Script, Post-Script, Failover, or Switchover.

- **host_name**

  Name of the host where this script will run. You can specify this option more than once.

- **path**

  Path to the script.

- **all_hosts**

  Allows the script to run on all the hosts in the system. This parameter overrides the host_name.

- **role**

  Configures the script based on the system role. By default, the script is configured for both primary and standby roles for a given system.

Examples

Example 1

```
emcli create_siteguard_script
      -system_name="BISystem1"
      -operation="Switchover"
      -script_type="Pre-Script"
      -path="/tmp/prescript"
      -all_hosts="true"
      -role="Primary"
```

Example 2

```
emcli create_siteguard_script
      -system_name="BISystem1"
      -operation="Switchover"
      -script_type="Pre-Script"
      -path="/tmp/prescript"
      -host_name="BIHOST1"
      -host_name="BIHOST2"
```

```
emcli create_siteguard_script
      -system_name="BISystem1"
      -operation="Switchover"
```

create_swlib_entity

Creates an entity in the software library. Upon successful creation, the entity revision appears under the specified folder on the software library home page.

Format

```
emcli create_swlib_entity
      -name="entity_name"
      -folder_id="folder_id"
      [-type]="type_internal_id"]
      [-subtype]="subtype_internal_id"]
      [-desc="entity_desc"]
      [-attr="<attr_name>:<attr value>"]
      [-prop="<prop_name>:<prop value>"]
      [-secret_prop="<secret_prop_name>:<secret_prop=_value>"]
      [-note="note_text"]
```

```
[ ] indicates that the parameter is optionalis optional
```

Parameters

- **name**

  Name of the entity.

- **folder_id**

  Identifier of the folder where the entity is to be created. The software library home page exposes the identifier for folders and entities as a custom column (Internal ID), and is hidden by default.

- **type**

  Use the list_swlib_entity_types verb to identify the type.

- **subtype**

  Internal identifier of the entity subtype, which defaults to the 'Generic Component' subtype for the 'Component' type. Use the list_swlib_entity_types verb to identify the subtype.

- **desc**

  Description of the entity.

- **attr**

  An attribute and its value, separated by a colon (:). To specify values for multiple attributes, repeat this option.

- **prop**

  A configuration property and its value, separated by a colon (:). To specify values for multiple properties, repeat this option.

- **secret_prop**

  A configuration property and its secret value separated by a colon (:). It is recommended to not specify the secret value on the command line. If omitted from the command line, the value is prompted for. To specify values for multiple properties, repeat this option.

- **note**

A note on the entity. For multiple notes, repeat this option.

Examples

Example 1

The following example creates an entity named 'myAcmeInstall' under the specified folder. The entity is of type 'Component' and subtype 'Generic Component, by default. The folder identifier value can be found on the software library home page. The software library home page exposes the identifier for folders and entities as a custom column (Internal ID), and is hidden by default.

```
emcli create_swlib_entity
      -name="myAcmeInstall"
      -folder_id=
"oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
```

Example 2

The following example creates an entity named 'myAcmeInstall' under the specified folder with the specified description. The entity is of type 'Component' and subtype 'Generic Component' by default. Values for the entity attributes, viz. PRODUCT, PRODUCT_VERSION and VENDOR, are specified. The value for the configuration property named DEFAULT_HOME is specified. A note on the entity is also specified. The identifier of the newly created entity revision is printed on the standard output.

```
emcli create_swlib_entity
      -name="myAcmeInstall"
      -folder_id=
"oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
       -desc="myAcmeInstall description"
       -attr="PRODUCT:Acme"
       -attr="PRODUCT_VERSION:3.0"
       -attr="VENDOR:Acme Corp"
       -prop="DEFAULT_HOME:/u01/acme3/"
       -note="myAcmeInstall for test servers"
```

create_swlib_folder

Creates a folder in the software library.

Format

```
emcli create_swlib_folder
      -name="folder_name"
      -parent_id="parent_folder_id"
      [-desc="folder_description"]
```

```
[ ] indicates that the parameter is optionalis optional
```

Parameters

- **name**

  Name of the folder.

- **parent_id**

  Identifier of the parent folder under which the folder is to be created. To create a folder under the root folder, specify the parent folder identifier as 'ROOT.' The software library home page exposes the identifier for folders and entities as a custom column (Internal ID) and is hidden by default.

- **desc**

  Description of the folder.

Example

The following example creates a folder named 'myFolder' under the specified parent folder.

```
emcli create_swlib_folder
      -name="myFolder"
      -parent_id=
"oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
      -desc="myFolder description"
```

create_system

Defines a system: name and its members. After the system is created, you can edit the system from the Enterprise Manager Cloud Control console to configure charts to be displayed for system members.

Format

```
emcli create_system
     -name="name"
     [-type=<system>]
     [-add_members="name1:type1:key_member/non_key_member;name2:type2;..."]...
     [-separator=add_members="sep_value"]
     [-subseparator=add_members="subsep_value"]
     -timezone_region="actual_timezone_region"
     [-owner="owner"]
     [-meta_ver="meta_version_of_system_type"]
     [-is_propagating="true|false"]
     [-availability_type="ALL|ANY"]

[ ] indicates that the parameter is optionalis optional
```

Parameters

- **name**

  Name of the system.

- **type**

  System type: generic_system. Defaults to "generic_system".

- **add_members**

  Add existing targets to the system. Each target is specified as a name-value pair `target_name:target_type`. You can specify this option more than once. key_member specifies that this target is a part of the systems availability calculation.

- **separator**

  Name-value pair separator for the given argument.

- **subseparator**

  Separates the name from the value for the given argument.

- **timezone_region**

  Actual time zone region.

- **owner**

  Owner of the system.

- **meta_ver**

  Meta version of the system type. Defaults to "1.0".

- **is_propagating**

  Flag to indicate if the privilege on the system will be propagated to member targets or not. The default value is false.

- **availability_type**

Availability calculation method of the system. Defining this is required if key_member is defined. ALL denotes that all key members must be up in order to mark the system as up. ANY denotes that at least one of the key members must be up in order to mark the system as up.

Examples

Example 1

The following example creates a generic system named db_system and supports backward compatibility. This system consists of two Oracle databases: emp_rec and payroll. The owner of this system is user1. The meta version of the system type is 3.0.

```
emcli create_system -name=db_system
     -add_members="emp_rec:oracle_database"
     -add_members="payroll:oracle_database"
     -timezone_region="PST8PDT"
     -owner="user1"
```

Example 2

The following example creates a generic system named my_system that consists of an oracle database (database2), listener (dblistener), and host (mymachine.myco.com). The owner of this system is the logged-in user. The meta version of the system type is 1.0. The example supports backward compatibility.

```
emcli create_system -name=my_system
     -add_members="database2:oracle_database;dblistener:oracle_listener
     -add_members="mymachine.myco.com:host"
     -timezone_region="PST8PDT"
```

Example 3

The following example creates a generic system named db_system1. This system consists of two Oracle databases: emp_rec and payroll. emp_rec is a key member for the system. The availability calculation method is if ANY of the key members is up, the system is up. The meta version of the system type is 3.0. This example shows the recommended method for creating a system.

```
emcli create_system -name=db_system1
     -add_members="emp_rec$oracle_database$key_member"
     -add_members="payroll$oracle_database"
     -subseparator=add_members="$"
     -timezone_region="PST8PDT"
     -availability_type="ANY"
```

create_udmmig_session

Creates a session to migrate user-defined metrics (UDMs) to metric extensions for targets.

Format

```
emcli create_udmmig_session
      -name=<session_name>
      -desc=<session_description>
      [-udm_choice=<specific_udm_to_convert>]*
      {-target=<type:name_of_target_to_migrate> }*
      | {-input_file=targetList:<complete_path_to_file>};
          {-template=<template_name_to_update> }*
      | {-input_file=templateList:<complete_path_to_file>}
      [-allUdms]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the migration session to be created.

- **desc**

  Description of the migration session to be created.

- **udm_choice**

  Specify if the session should migrate specific UDMs. Otherwise, all UDMs are migrated.

- **target**

  The type:name of the target to be updated. You can specify multiple values.

- **input_file**

  Specify a file name that contains a list of targets, one per line, in the following format:

  ```
  <targetType>:<targetName>
  ```

- **template**

  Name of the monitoring template to update. You can specify multiple values.

- **input_file**

  Specify a file name that contains a list of templates, one name per line.

- **allUdms**

  Forces the session to contain all UDMs from targets and templates. (The default behavior just selects those not in a session.)

Examples

Example 1

The following example creates a new session named hostsession that migrates the UDM hostudm on the target testhost.

```
emcli create_udmmig_session
      -name=hostsession -desc="Convert UDMs for Host Target"
      -udm_choice=hostudm -target=host:testhost
```

Example 2

The following example creates a new session named hostsession that migrates all the unconverted UDMs on the target testhost that are not in a session.

```
emcli create_udmmig_session
      -name=hostsession -desc="Convert UDMs for Host Target"
      -target=host:testhost -allUdms
```

create_user

Creates a new Enterprise Manager administrator.

Format

```
emcli create_user
       -name="name"
       -password="password"
       [-type="user_type"]
       [-roles="role1;role2;..."]
       [-email="email1;email2;..."]
       [-privilege="name[;secure-resource-details]]"
       [-separator=privilege="sep_string"]
       [-subseparator=privilege="subsep_string"]
       [-profile="profile_name"]
       [-desc="user_description"]
       [-expired="true|false"]
       [-prevent_change_password="true|false"]
       [-department="department_name"]
       [-cost_center="cost_center"]
       [-line_of_business="line_of_business"]
       [-contact="contact"]
       [-location="location"]
       [-input_file="arg_name:file_path"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Administrator name.

- **password**

  Administrator password.

- **type**

  Type of User. The default value of this parameter is EM_USER. Possible values for this parameter are:

  – EM_USER

  – EXTERNAL_USER

  – DB_EXTERNAL_USER

- **roles**

  List of roles to grant to this administrator. Currently, the built-in roles include PUBLIC.

- **email**

  List of e-mail addresses for this administrator.

- **privilege**

  Privilege to grant to this administrator. You can specify this option more than once. The original administrator privileges will be revoked. Specify <secure_resource_details> as:

```
resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
name2=resource_column_value2]..]"
```

To retrieve the list of system privileges that do not require resource information, execute the get_supported_privileges command.

- **separator**

  Specify a string delimiter to use between name-value pairs for the value of the privilege option. The default separator delimiter is a semi-colon (;).

- **subseparator**

  Specify a string delimiter to use between name and value in each name-value pair for the value of the privilege option. The default subseparator delimiter is a colon (:).

- **profile**

  Database profile name. It uses DEFAULT as the default profile name.

- **desc**

  User description for the user being added.

- **expired**

  Use this option to expire the password immediately. The default is false.

- **prevent_change_password**

  When set to true, you cannot change your own password. The default is false.

- **department**

  Name of the department of the administrator.

- **cost_center**

  Cost center of the administrator in the organization.

- **line_of_business**

  Line of business of the administrator.

- **contact**

  Contact information of the administrator.

- **location**

  Location of the administrator.

- **input_file**

  Allow the administrator to provide the value of any argument in a file. The format of the value will be the name_of_argument:file_path_with_file_name. You can specify this option more than once.

Examples

Example 1

The following example creates an Enterprise Manager administrator named new_admin. This administrator has two privileges: the ability to view the job with ID 923470234ABCDFE23018494753091111 and the ability to view the target host1.example.com:host. The administrator new_admin is granted the PUBLIC role.

```
emcli create_user
      -name="new_admin"
      -password="oracle"
      -email="first.last@example.com;joe.shmoe@shmoeshop.com"
      -roles="public"
      -privilege="view_job;923470234ABCDFE23018494753091111"
      -privilege="view_target;host1.example.com:host"
```

Example 2

The following example makes User1 an Enterprise Manager user, which is already created on an external user store like the SSO server. The contents of priv_file are view_target;host1.example.com:host . User1 will have view privileges on the host1.example.com:host target.

```
emcli create_user
      -name="User1"
      -type="EXTERNAL_USER"
      -input_file="privilege:/home/user1/priv_file"
```

Example 3

The following example makes User1 an Enterprise Manager user, provides a description for the user, and prevents the password from being changed. Only another super administrator can change the password. The profile is set as MGMT_ADMIN_ USER_PROFILE.

```
emcli create_user
      -name="User1"
      -desc="This is temp hire."
      -prevent_change_password="true"
      -profile="MGMT_ADMIN_USER_PROFILE"
```

Example 4

The following example makes User1 an Enterprise Manager user, provides a description for the user, and immediately expires the password. When the user logs in the first time, he/she must change the password.

```
emcli create_user
      -name="User1"
      -desc="This is temp hire."
      -expire="true"
```

Example 5

The following example makes User1 an Enterprise Manager user, and provides a description, department name, cost center, line of business, contact, and location for the administrator.

```
emcli create_user
      -name="User1"
      -password="oracle"
      -desc="This is temp hire."
      -department="dept1"
      -cost_center="testCostCenter"
      -line_of_business="testLineOfBusiness"
      -contact="contact"
      -location="location"
```

define_diagcheck_exclude

Defines a diagnostic check exclusion with regard to groups and checks to exclude.

Format

```
emcli define_diagcheck_exclude
      -target_type="type"
      -exclude_name="name"
      { [-excl_group="diag_group" ]*
        [-excl_check="diag_check" ]* |
       -input_file=excl_def:<complete_path_to_file> }
```

```
[ ] indicates that the parameter is optionalis optional
```

Parameters

- **target_type**

  Type of target.

- **exclude_name**

  Name to use for the exclusion.

- **excl_group**

  Group of diagchecks to exclude.

- **excl_check**

  Name of diagcheck to exclude.

delete_blackout

Deletes a blackout that has already ended or has been fully stopped. You cannot delete a blackout that is either in progress or currently scheduled. You need to first run `stop_blackout`.

Format

```
emcli delete_blackout
    -name="name"
    [-createdby="blackout_creator"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the blackout to delete.

- **createdby**

  Enterprise Manager user who created the blackout. The default is the current user. The SUPER_USER privilege is required to delete a blackout created by another user.

Examples

Example 1
The following example deletes blackout `backup_monthly` created by the current user.

```
emcli delete_blackout -name=backup_monthly
```

Example 2
The following example deletes blackout `db_maintenance` that was created by Enterprise Manager administrator `sysadmin2`. The current user must either be user `sysadmin2` or a user with the SUPER_USER privilege.

```
emcli delete_blackout -name=db_maintenance -createdby=sysadmin2
```

delete_credential_set

Deletes a credential set. Only Enterprise Manager Super Administrators can delete credential sets. Out-of-box credential sets cannot be deleted.

Format

```
emcli delete_credential_set
      -set_name="set_name"
      -target_type="ttype"
```

Parameters

- **set_name**

  Credential set name to be deleted.

- **target_type**

  Target type of the credential set.

Examples

The following example creates a credential set named Old_Credential_Set.

```
emcli delete_credential_set
      -set_name=Old_Credential_Set
      -target_type=host
```

delete_group

Deletes a group. Deleting a non-existent group generates the error "Group X does not exist."

Format

```
emcli delete_group
      -name="name"
      [-type=<group>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the group to delete.

- **type**

  Group type: group. Defaults to "group".

Examples

Example 1

The following example removes the group `payroll_group` that consists of database target types.

```
emcli delete_group -name=payroll_group
```

Example 2

The following example removes the group `my_hosts` that consists of host target types.

```
emcli delete_group -name=my_hosts
```

Example 3

The following example removes the group `my_group` that consists of mixed target types.

```
emcli delete_group -name=my_group
```

delete_instance

Deletes a stopped or completed deployment instance. An instance can only be deleted when its status is stopped, completed, or completed with an error.

Format

```
emcli delete_instance
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

Examples

Example 1

```
emcli delete_instance -instance=16B15CB29C3F9E6CE040578C96093F61
```

Example 2

```
emcli delete_instance -exec=2B15CB29C3F9E6CE040578C96093F16
```

delete_job

Deletes a specified job. A job cannot be deleted if any of its executions are in the EXECUTING (Running) state. Use the get_jobs verb to obtain a list of existing jobs along with their job IDs and statuses.

Format

```
emcli delete_job
      -job_id="jobID" | -name="jobName"
```

Parameters

- **job_id**

  Job ID of the job to delete.

- **name**

  Name of the job to delete. To uniquely identify the job, the current user is used.

Examples

Example 1
The following example deletes an existing job with the job ID 123456789012345678901234567890012.

```
emcli delete_job -job_id=123456789012345678901234567890012
```

Example 2
The following example deletes an existing job named my_job, which belongs to the current Enterprise Manager user.

```
emcli delete_job -name=my_job
```

delete_library_job

Deletes a library job you created using the create_library_jobs command.

Format

```
emcli delete_library_job
      -name=<library_job_name>
      [-owner=<library_job_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the library job.

- **owner**

  Owner of the library job if different from the current logged-in EM CLI administrator.

Examples

Example 1
The following example deletes the library job "libjob1" owned by the current logged-in Enterprise Manager administrator.

```
emcli delete_library_job -name=libjob1
```

Example 2
The following example deletes the library job "libjob2" owned by the Enterprise Manager administrator "emadmin1."

```
emcli delete_library_job -name=libjob2 -owner=emadmin1
```

delete_metric_promotion

Deletes a promoted metric.

Format

```
emcli delete_metric_promotion
      -name=<service_target_name>
      -type=<service_target_type>
      [-category=<usage/performance/business>]
      [-promotedMetricName=<promoted_metric]
      [-promotedMetricColumn=<promoted_metric_column>]
      -promotedMetricKey=<key_value_of_promoted_metric>

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the service target.

- **type**

  Name of the service type.

- **category**

  Defines whether the promoted metric is a usage or a performance metric of a service. This determines the promoted metric name and metric column. If you do not specify this, you must specify the promotedMetricName and promotedMetricColumn.

- **promotedMetricName**

  Promoted metric name. This is optional if you specify the category .

- **promotedMetricColumn**

  Promoted metric column. This is optional if you specify the category .

- **promotedMetricKey**

  Determines the key value of the promoted metric. It is equivalent to the displayed name of the promoted metric in the user interface.

Examples

The following example deletes the promoted performance metric with the key value mymetric1 on the service MyTarget.

```
emcli delete_metric_promotion -name='MyTarget' -type='generic_service'
      -category=Performance -promotedMetricKey=mymetric1
```

delete_named_credential

Deletes an existing named credential.

Format

```
emcli delete_named_credential
      -cred_owner=<owner>
      -cred_name=<name>
```

Parameters

- **cred_owner**

  Credential owner.

- **cred_name**

  Required credential name. This does not support wild cards.

delete_operation_plan

Deletes the specified operation plan from a Site Guard configuration.

Format

```
emcli delete_operation_plan
    -name=<plan_name>
```

Parameters

- **name**

  Name of the operation plan you want to delete.

Example

```
emcli delete_operation_plan
    -name="BISystem1-switchover"
```

delete_patches

Deletes patches from the software library.

Format

```
emcli delete_patches
      -patch_name=<patch_name>
      -release=<release_id>
      -platform=<platform_id>
```

Parameters

- **patch_name**

  Patch number.

- **release**

  Patch release ID.

- **platform**

  Patch platform ID.

Example

```
emcli delete_patches -patch_name=13741363 -release=80112310 -platform=226
```

See Also

create_patch_plan
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

delete_privilege_delegation_settings

Deletes a privilege delegation setting template.

Format

```
emcli delete_privilege_delegation_settings
      -setting_names="setting_name1;setting_name2;setting_name3;"
```

Parameters

■   **setting_names**

Name of the settings you want to delete.

Example

The following example deletes the privilege settings for the names
setting_name1, setting_name2, and setting_name3.

```
emcli delete_privilege_delegation_settings
      -setting_names="sudo_setting1;sudo_setting2;pbSetting1
```

delete_resolution_state

Deletes an existing resolution state. You typically use this command for resolution states that are no longer used. You need to also specify an alternative resolution state in case there are any references to the state. In this case, the references are changed to this alternative state. This action might require some time.

Only a super administrator can execute this command. A success message is reported if the command is successful. An error message is reported if the deletion fails.

> **Note:** No notifications are sent for any incidents or problems updated in this process.

Format

```
emcli delete_resolution_state
     -label="label of the state to be deleted"
     -alt_res_state_label="alternative resolution state"
```

Parameters

- **label**

  Label of the state to be deleted.

- **alt_res_state_label**

  Alternative state to be used.

Examples

The following example deletes the resolution state "Waiting for SR" and replaces any references to this state with the state "Work in Progress".

```
emcli delete_resolution_state  -label="Waiting for SR" -alt_res_state_label="Work
in Progress"
```

delete_role

Deletes an existing Enterprise Manager administrator role.

Format

```
emcli delete_role
      -name="role_name"
```

Parameters

- **name**

  Role name.

Examples

The following example deletes the role name existing_role.

```
emcli delete_role -name="existing_role"
```

delete_siteguard_configuration

Deletes the Site Guard configuration. The entire configuration (scripts, credential associations, site associations, operation plans) pertaining to the specified system and all the associated standby systems are deleted.

Format

```
emcli delete_siteguard_configuration
      -primary_system_name=<name> | -standby_system_name=<name>
```

Parameters

- **primary_system_name**

  Name of the primary system. Specify either primary_system_name or standby_system_name.

- **standby_system_name**

  Name of the standby system.

Examples

Example 1

```
emcli delete_siteguard_configuartion
      -primary_system_name="BISystem1"
```

Example 2

```
emcli delete_siteguard_configuration
      -standby_system_name="BISystem2"
```

See Also

create_siteguard_configuration

get_siteguard_configuration

delete_siteguard_credential_association

Deletes the credential association from the Site Guard configuration.

Format

```
emcli delete_siteguard_credential_association
      -system_name=<name>
      [-target_name=<name>]
      -credential_type=<type>
```

```
{ } indicates that the parameter is optional
```

Parameters

- **system_name**

  Name of the system.

- **target_name**

  Name of the target.

- **credential_type**

  Type of the credential, which can be HostNormal, HostPrivileged, WLSAdmin, or DatabaseSysdba.

Examples

Example 1

```
emcli create_siteguard_credential_association
      -system_name="BISystem1"
      -credential_type="HostNormal"
      -credential_name="HOST-SGCRED"
      -credential_owner="sysman"
```

Example 2

```
emcli create_siteguard_credential_association
      -system_name="BIsystem1"
      -target_name="database-instance"
      -credential_type="HostNormal"
      -credential_name="HOST-DBCRED"
      -credential_owner="sysman"
```

See Also

create_siteguard_credential_association

update_siteguard_credential_association

get_siteguard_credential_association

delete_siteguard_script

Deletes the specified script from the Site Guard configuration.

Format

```
emcli delete_siteguard_script
    -script_id=<script_id>
```

Parameters

- **script_id**

  ID associated with the script.

Examples

```
emcli delete_siteguard_script
    -script_id="10"
```

See Also

create_siteguard_script

get_siteguard_scripts

delete_siteguard_script_hosts

Deletes the host or hosts associated with a given script.

Format

```
emcli delete_siteguard_script_hosts
      -script_id=<script_id>
      -host_name=<name1;name2;...>
```

Parameters

- **script_id**

  ID associated with the script.

- **host_name**

  Name of the host where this script will be run. You can specify this parameter more than once.

Examples

```
emcli delete_siteguard_script_hosts
      -script_id="10"
      -host_name="BIHOST1"
```

Output Columns

Step Number, Operation Name, Target Name, Target Host, and Error Mode

See Also

create_siteguard_script

add_siteguard_script_hosts

delete_sla

Deletes one or more SLAs for a target.

Format

```
emcli delete_sla
    -targetName=<target_name>
    -targetType=<target_type>
    -slaName=<SLA_name>
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

Example

The following example deletes the SLA with the name 'gold_sla' from the target.

```
emcli delete_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla'
```

delete_system

Deletes a system.

Format

```
emcli delete_system
      -name="name"
      [-type=<generic_system>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the system to delete.

- **type**

  System type: generic_system. Defaults to "generic_system".

Examples

The following example deletes the system my_system.

```
emcli delete_system -name=my_system
```

delete_target

Deletes a specified target from the Enterprise Manager Cloud Control monitoring framework. Deleting a target removes it from the Management Repository and does not physically remove the target itself.

You can use the `get_targets` verb to obtain a list of available targets and their respective types.

Format

```
emcli delete_target
      -name=<name>
      -type=<type>
      [-delete_monitored_targets]
      [-async]
      [-delete_members]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name.

- **type**

  Target type.

- **delete_monitored_targets**

  Deletes the targets monitored by the specified Management Agent. This is only applicable with the oracle_emd target type.

- **async**

  Deletes the target asynchronously.

- **delete_members**

  Deletes all the members of the target as well.

Examples

Example 1

The following example deletes the `oracle_database` target with the name `database`.

```
emcli delete_target
      -name="database"
      -type="oracle_database"
```

Example 2

The following example deletes the Agent named test.example.com:1836 and all of its monitored targets. The Agent must be marked UNREACHABLE in Enterprise Manger Cloud Control to perform this operation.

```
emcli delete_target
      -name="test.example.com:1836"
      -type="oracle_emd"
```

```
-delete_monitored_targets
-async
```

Example 3

The following example deletes the example_ias_farm target with the name "farm01_base_domain" and all of its members, such as domain, clusters, servers, application deployments, and so forth.

```
emcli delete_target
      -name="farm01_base_domain"
      -type="example_ias_farm"
      -delete_members
```

delete_test

Deletes a Services test along with its constituent steps and step groups.

Format

```
emcli delete_test
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

Example

The following example deletes an HTTP test name MyTest for the generic_service target name MyTarget.

```
emcli delete_test -name='MyTarget' -type='generic_service'
                  -testname='MyTest' -testtype='HTTP'
```

delete_test_threshold

Deletes a test threshold.

Format

```
emcli delete_test_threshold
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>
      -metricName=<metric_name>
      -metricColumn=<metric_column>
      [-beaconName=<beacon_name>]
      [-stepName=<step_name>]
      [-stepGroupName=<stepgroup_name>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

- **metricName**

  Name of the metric.

- **metricColumn**

  Name of the column.

- **beaconName**

  Name of the beacon.

- **stepName**

  Name of the step.

- **stepGroupName**

  Name of the step group.

Example

```
emcli delete_test_threshold
      -name="Service Name"
      -type="generic_service"
      -testname="Test Name"
      -testtype="HTTP"
      -metricName="http_response"
      -metricColumn="timing"
```

delete_user

Deletes an existing Enterprise Manager administrator.

When a user is deleted, all jobs the user creates are stopped and deleted. Also, any blackouts the user creates are deleted. However, a user cannot be deleted if any blackouts the user creates are active at the time the call to delete the user is issued. This situation is considered an invalid state from which to delete a user. First, all of these active blackouts must be stopped, and a thwarted delete user call must be reissued.

Format

```
emcli delete_user
      -name=<user_name>
      [-new_object_owner=<user_name>]
      [-force]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Administrator name.

- **new_object_owner**

  Name of the administrator to assign the secure objects owned by the current administrator being deleted. If you do not specify this option, the secure objects are deleted that are owned by the administrator being deleted.

- **force**

  Deletes the administrator even if the administrator is currently logged in.

Examples

Example 1

The following example deletes the Enterprise Manager administrator named sysman3.

```
emcli delete_user -name=sysman3
```

Example 2

The following example deletes the Enterprise Manager administrator named user1, and assigns all the secure objects owned by user1 to user5.

deploy_bipublisher_reports

Deploys the Enterprise Manager system reports to a configured and running BI Publisher server. By default, all installed reports are deployed. Alternatively, you can restrict the deployment to certain plug-ins.

You can also use this verb to upload a reports jar file (located on the OMS(s)'s file system. The operation does not overwrite existing BI Publisher Reports in the Enterprise Manager reports folder unless you specify -force.

Format

```
emcli deploy_bipublisher_reports
    [-force]
    [-pluginid=<plugin_id>]
    [-pluginversion=<plugin_version>]
    [-reportsjarfile=<reports_jar_file>]

[ ] indicates that the parameter is optional
```

Parameters

- **force**

  Overwrites reports. If you use this option, all reports on the BI Publisher server are overwritten with the new copies.

- **pluginid**

  In addition to Enterprise Manager system reports, also deploys any subsequently loaded plug-in-based BI Publisher Reports.

- **pluginversion**

  Limits the plug-ins to a specific version.

- **reportsjarfile**

  Deploys a single Enterprise Manager reports jar file that contains one or more BI Publisher Reports. This jar file is located relative to the OMS's $ORACLE_HOME.

Examples

Example 1

```
emcli deploy_bipublisher_reports -force
```

Example 2

The following example deploys the upgraded Chargeback and Trending reports shipping with Enterpise Manager 12cR2 (12.1.0.2).

```
emcli deploy_bipublisher_reports -pluginid=oracle.sysman.emct
-pluginversion=12.1.0.3.0
```

deploy_plugin_on_agent

Deploys a plug-in on Management Agents. Agent names must be provided for plug-in deployment.

> **Note:** A plug-in can only be deployed on any Management Agent after it has been successfully deployed on the management server.

Format

```
emcli deploy_plugin_on_agent
      -agent_names=<agent1;agent2>
      -plugin=<plug-in_id[:version>]
      [-discovery_only]
```

[ ] indicates that the parameter is optional

Parameters

- **agent_names**

  Management Agents (host:port) on which the plug-in needs to be deployed.

- **plugin**

  Plug-in ID and version that needs to be deployed. Version is optional, and it defaults to the latest applicable version deployed on the management server. If a later version is available but not certified on the Agent OS platform, the latest version is not picked up.

- **discovery_only**

  To be used when only discovery content needs to be deployed.

Examples

Example 1
The following example deploys the latest version of oracle.sysman.db2 on Management Agent myhost1.example.com.

```
emcli deploy_plugin_on_agent -plugin="oracle.sysman.db2"
-agent_names="myhost1.example.com:1838"
```

Example 2
The following example deploys version 12.1.0.1.0 of plug-in oracle.sysman.db2 on management agent myhost1.us.oracle.com.

```
emcli deploy_plugin_on_agent
      -plugin="oracle.sysman.db2:12.1.0.1.0"
      -agent_names="myhost1.us.oracle.com:1838"
```

deploy_plugin_on_server

Deploys a plug-in on the Management Servers. The deployment process for some plug-ins might restart the Management Servers. If the plug-in is already deployed on one of the servers, this server is skipped. If a lower version of the plug-in is already deployed, the plug-in is upgraded. If a lower revision of the plug-in is already deployed, the new revision is applied.

Format

```
emcli deploy_plugin_on_server
        -plugin=<plug-in_id>[:<version>]
        [-sys_password=<sys_password>]
        [-prereq_check]
        [-use_last_prereq_result]

[ ] indicates that the parameter is optional
```

Parameters

- **plugin**

  ID or ID:Version of the plug-in to be deployed on the Management Servers of the form -plugin=<oracle.sysman.db:12.1.0.1.0>, where the plug-in ID (like oracle.sysman.db) is a required parameter, and the version is optional.

  If do not specify a version, the highest version of the plug-in that has been downloaded is considered for deployment. If multiple revisions of this plug-in version are downloaded, the highest revision is considered for deployment.

- **sys_password**

  Password of the repository DBA SYS. If you do not provide this , you are prompted for the password. This is not required if you use the prereq_check .

- **prereq_check**

  If you provide this option, instead of deploying the plug-in, the verb displays only a check for all the unfulfilled prerequisites for this plug-in deployment to be successful. If you do not provide this option, plug-in deployment follows a prerequisities check.

- **use_last_prereq_result**

  If prerequisites checks have been performed previously for a given set of plug-ins using the -prereq_check option and no other deployment activity occurred for these plug-ins, you can use this option to skip prerequisite checks and start the deployment immediately.

Examples

Example 1
The following example deploys the latest downloaded version of Oracle Database plug-in (plug-in ID: oracle.sysman.db) on the management server.

```
emcli deploy_plugin_on_server
    -plugin=oracle.sysman.db
    -sys_password=<sys_password>
```

Example 2

The following example deploys the latest downloaded version of a Oracle Database plug-in (plug-in ID: oracle.sysman.db) and Oracle Fusion Middleware plug-in (oracle.sysman.emas) on the management server.

```
emcli deploy_plugin_on_server
     -plugin="oracle.sysman.db;oracle.sysman.emas"
     -sys_password=<sys password>
```

Example 3

The following example deploys the Oracle Database plug-in (with version 12.1.0.2.0) and Oracle Fusion Middleware plug-in (version 12.1.0.2.0) on the management server. Since sys password has not been passed on the command line, you are prompted for it.

```
emcli deploy_plugin_on_server
     -plugin="oracle.sysman.db:12.1.0.2.0;oracle.sysman.emas:12.1.0.2.0"
```

Example 4

The folllowing example deploys the Oracle Database plug-in (with version 12.1.0.2..0) and Oracle Fusion Middleware plug-in (12.1.0.2.0) on the management server. Since sys password has not been passed on the command line, you are prompted for it. If a lower version of both plug-ins have already been deployed, they are upgraded to 12.1.0.2.0. If a lower version of only one of the plug-ins is deployed, this generates an error, and you will have to deploy them separately.

```
emcli deploy_plugin_on_server
     -plugin="oracle.sysman.db:12.1.0.2.0;oracle.sysman.emas:12.1.0.2.0"
```

Example 5

The following example only performs prerequisite checks on the Oracle Database plug-in and does not actually deploy the plug-in.

```
emcli deploy_plugin_on_server
     -plugin=oracle.sysman.db:11.2.0.1.0 -prereq_check
```

describe_job

Describes a job and gets its properties for a job you have submitted using the create_
job verb. The output can be redirected into a file and used as a template.

Format

```
emcli describe_job
      -name=<job_name>
      [-owner=<job_owner>]
      [-verbose]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the job to describe.

- **owner**

  Enterprise Manager administrator who owns this job. If not provided, the current
  EM CLI logged-in administrator is assumed as the owner. The logged-in
  Enterprise Manager administrator must have at least the view privilege to
  describe a job.

- **verbose**

  Outputs a help template along with the properties.

Examples

Example 1
The following example describes the library job "myJob" owned by the logged-in
Enterprise Manager administrator.

```
emcli describe_job -name=myJob
```

Example 2
The following example describes the library job "yourJob" owned by the Enterprise
Manager administrator "admin1". The logged-in Enterprise Manager administrator
has view privilege on this job.

```
emcli describe_job -name=yourJob -owner=admin1
```

Example 3
```
emcli describe_job -name=EMCLI_JOB_2

# Job Name : EMCLI_JOB_2

# Current status of the job is ACTIVE.

# Job Type: OSCommand.
# This job type supports the following target types only :
# host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.
```

```
job_target_list.1=adc2110610.us.oracle.com:host

# Variable: args
# Description: Parameters of the command to run on the target
variable.args=hello

# Variable: command
# Description: Command to run on the target
variable.command=echo

# Credential Usage: defaultHostCred
# Description:
cred.defaultHostCred.adc2110610.us.oracle.com:host=NAMED:SYSMAN:CRED1

schedule.frequency=REPEAT_BY_MINUTES
schedule.startTime=2012-02-01 01:01:01.0
schedule.endTime=2051-02-01 01:01:01.0
schedule.gracePeriod=-1
schedule.months=
schedule.days=
schedule.interval=1
schedule.timezone.type=TIMEZONE_TARGET
schedule.timezone.targetIndex=1
schedule.timezone.zoneOffset=0
schedule.timezone.region=
```

Example 4

```
emcli describe_job -name=EMCLI_JOB_2 -verbose

# Job Name : EMCLI_JOB_2

# Current status of the job is ACTIVE.

# Job Type: OSCommand.
# This job type supports the following target types only :
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.


# Target List.
# In a target list, each member is specified using the target name and target type
# in the fashion:
#    target_name:target_type
# To specify an element of the target list, the following notation is used:
#    job_target_list.1=target_name:target_type
# The suffix "1" after the key word "job_target_list" signifies that the entry is
# for the first element.
# The target target_name:target_type should exists in EM.
# Permissible target types are:
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.
# A sample target list could be:
# job_target_list.1=<target_name>:host
# job_target_list.2=<target_name>:host
# The target list can only contain targets of the same target type. A cluster,
```

```
# group, domain or system
# target must not be intermixed with targets of the other target types.


# Variable List.
# In a variable list, each member is specified in the following way:
# Scalar variable: A variable whose value can be represented as a single string.
#    variable.variable_name=variable_value
# Here "variable" is a keyword. Variable name is the name of the variable whose
# value is being specified.
# Value is specified on the right hand side after the equal to sign.
# Vector variable: A variable whose value is represented as an array or list of
# string values.
#    variable.variable_name.1=value1
#    variable.variable_name.2=value2
# Here the numbers suffixing the variable name signify the entry number in the
# list.
# Large variable: A variable whose value is exceptionally large. Syntax is similar
# to a scalar variable.
#    variable.large_variable_name=a_very_very_big_value


# Credential List.
# This is the list of credential usages declared by the job type.
# Each entry takes the form:
#    cred.credusage_name.target_details=cred_type:cred_details
# Here the prefix "cred" is a keyword signifying that this line represents a
# credential entry.
# "credusage_name" would be substituted with the name of the credential usage
# declared in the job type.
# This is followed by the target details, which take the following form:
#    target_name:target_type
# The value for this credential usage entry is specified using the type of the
# credential and its details.
# "cred_type" can take either "SET" or "NAMED" as its value, depending on whether
# the credential is a credential set or a named credential.
# "cred_details" can specify either the name of a credential set or the name of a
# named credential based on the "cred_type"
# A sample entry for a target target1:host for credential usage defaultHostCred
# for a credential set could look like:
#    cred.defaultHostCred.target1:host=SET:HostCredsNormal
# A sample entry for a target target1:host for credential usage defaultHostCred
# for a named credential could look like:
#    cred.defaultHostCred.target1:host=NAMED:MyNamedCredential
# A sample entry for a target target1:host for credential usage defaultHostCred
# for a named credential shared by EM Admin "admin1" could look like:
#    cred.defaultHostCred.target1:host=NAMED:admin1:MyNamedCredential

# Schedule.
# Specify a schedule for the job. Detailed instructions as per below:
# Frequency: Specifies the frequency of repeatedly submitting instances of this
# job.
#    scheule.frequency=Frequency_Type
# Frequency type could be either of IMMEDIATE, ONCE, WEEKLY, MONTHLY, YEARLY,
# REPEAT_BY_MINUTES, REPEAT_BY_HOURS, REPEAT_BY_DAYS, REPEAT_BY_WEEKS.
# If frequency is IMMEDIATE, then other schedule fields do not matter.
# Start Time: Start time for the schedule.
#    scheule.startTime=MM-DD-YYYY
# End Time: End time for the schedule.
#    scheule.endTime=MM-DD-YYYY
```

```
# Grace Period: Grace period in minutes for the schedule.
#     scheule.graceperiod=
# Months : Months for repetition. January is denoted by 0 and December by 11
#     schedule.months=0,1,2
# Days: Days of the week for repetition. Sunday is denoted by 0 and Saturday by 6.
#     schedule.days=0,1,2
# Timezone: Timezone information is further detailed into type, target index, zone
# offset and region.
#     schedule.timezone.type: either of TIMEZONE_TARGET, TIMEZONE_SPECIFIED,
# TIMEZONE_REGION_SPECIFIED.
#     schedule.timezone.targetIndex : specify the index of the target whose
# timezone is to be used.
#     schedule.timezone.zoneOffset : timezone offset.
#     schedule.timezone.region : timezone region
# Following is a complete schedule section, remove #  and populate the values for
# submission:
# scheule.frequency=ONCE
# schedule.startTime=12-21-2012
# schedule.endTime=12-21-2012
# schedule.gracePeriod=10
# schedule.months=
# schedule.days=
# schedule.timezone.type=TIMEZONE_TARGET
# schedule.timezone.targetIndex=1
# schedule.timezone.zoneOffset=
# schedule.timezone.region=

job_target_list.1=adc2110610.us.oracle.com:host

# Variable: args
# Description: Parameters of the command to run on the target
variable.args=hello

# Variable: command
# Description: Command to run on the target
variable.command=echo

# Credential Usage: defaultHostCred
# Description:
cred.defaultHostCred.adc2110610.us.oracle.com:host=NAMED:SYSMAN:CRED1

schedule.frequency=REPEAT_BY_MINUTES
schedule.startTime=2012-02-01 01:01:01.0
schedule.endTime=2051-02-01 01:01:01.0
schedule.gracePeriod=-1
schedule.months=
schedule.days=
schedule.interval=1
schedule.timezone.type=TIMEZONE_TARGET
schedule.timezone.targetIndex=1
schedule.timezone.zoneOffset=0
schedule.timezone.region=
```

describe_job_type

Describes the job type and gets its properties. The output can be redirected into a file.

This verb dumps out a properties file for a job type that supports the Job System Generic EM CLI. This file contains some documentation, a list of all required credential usages, and a list of all variables required to create a (library) job instance of the job type.

Format

```
emcli describe_job_type
    -job_type=<job_type_internal_name>
    [-verbose]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **job_type**

  Specify the name of the job type to describe. You can use the get_job_types verb to obtain the names of all job types for which a job or library jobs can be created using EM CLI.

- **verbose**

  Outputs a help template along with the properties.

Examples

Example 1
The following example describes the job type "MyJobType."

```
emcli describe_job_type -job_type=MyJobType
```

Example 2
The following example produces a property file on the console, which can be redirected to a file and used multiple times.

```
emcli describe_job_type -job_type=OSCommand
```

```
# Job Type: OSCommand.
# This job type supports the following target types only :
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.


# Variable: args
# Description: Parameters of the command to run on the target
variable.args=

# Variable: command
# Description: Command to run on the target
variable.command=

# Credential Usage: defaultHostCred
```

```
# Description:
cred.defaultHostCred.<target_name>:<target_type>=
```

Example 3

The following example with the verbose option generates a property dump with help on how to specify each individual property for the job.

```
emcli describe_job_type -job_type=OSCommand -verbose

# Job Type: OSCommand.
# This job type supports the following target types only :
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.


# Target List.
# In a target list, each member is specified using the target name and target type
# in the fashion:
#     target_name:target_type
# To specify an element of the target list, the following notation is used:
#     job_target_list.1=target_name:target_type
# The suffix "1" after the key word "job_target_list" signifies that the entry is
# for the first element.
# The target target_name:target_type should exists in EM.
# Permissible target types are:
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.
# A sample target list could be:
# job_target_list.1=<target_name>:host
# job_target_list.2=<target_name>:host
# The target list can only contain targets of the same target type. A cluster,
# group, domain or system
# target must not be intermixed with targets of the other target types.


# Variable List.
# In a variable list, each member is specified in the following way:
# Scalar variable: A variable whose value can be represented as a single string.
#     variable.variable_name=variable_value
# Here "variable" is a keyword. Variable name is the name of the variable whose
# value is being specified.
# Value is specified on the right hand side after the equal to sign.
# Vector variable: A variable whose value is represented as an array or list of
# string values.
#     variable.variable_name.1=value1
#     variable.variable_name.2=value2
# Here the numbers suffixing the variable name signify the entry number in the
# list.
# Large variable: A variable whose value is exceptionally large. Syntax is similar
# to a scalar variable.
#     variable.large_variable_name=a_very_very_big_value


# Credential List.
# This is the list of credential usages declared by the job type.
# Each entry takes the form:
#     cred.credusage_name.target_details=cred_type:cred_details
# Here the prefix "cred" is a keyword signifying that this line represents a
```

```
                # credential entry.
                # "credusage_name" would be substituted with the name of the credential usage
                # declared in the job type.
                # This is followed by the target details, which take the following form:
                #     target_name:target_type
                # The value for this credential usage entry is specified using the type of the
                # credential and its details.
                # "cred_type" can take either "SET" or "NAMED" as its value, depending on whether
                # the credential is a credential set or a named credential.
                # "cred_details" can specify either the name of a credential set or the name of a
                # named credential based on the "cred_type"
                # A sample entry for a target target1:host for credential usage defaultHostCred
                # for a credential set could look like:
                #     cred.defaultHostCred.target1:host=SET:HostCredsNormal
                # A sample entry for a target target1:host for credential usage defaultHostCred
                # for a named credential could look like:
                #     cred.defaultHostCred.target1:host=NAMED:MyNamedCredential
                # A sample entry for a target target1:host for credential usage defaultHostCred
                # for a named credential shared by EM Admin "admin1" could look like:
                #     cred.defaultHostCred.target1:host=NAMED:admin1:MyNamedCredential

                # Schedule.
                # Specify a schedule for the job. Detailed instructions as per below:
                # Frequency: Specifies the frequency of repeatedly submitting instances of this
                # job.
                #     scheule.frequency=Frequency_Type
                # Frequency type could be either of IMMEDIATE, ONCE, WEEKLY, MONTHLY, YEARLY,
                # REPEAT_BY_MINUTES, REPEAT_BY_HOURS, REPEAT_BY_DAYS, REPEAT_BY_WEEKS.
                # If frequency is IMMEDIATE, then other schedule fields do not matter.
                # Start Time: Start time for the schedule.
                #     scheule.startTime=MM-DD-YYYY
                # End Time: End time for the schedule.
                #     scheule.endTime=MM-DD-YYYY
                # Grace Period: Grace period in minutes for the schedule.
                #     scheule.graceperiod=
                # Months : Months for repetition. January is denoted by 0 and December by 11
                #     schedule.months=0,1,2
                # Days: Days of the week for repetition. Sunday is denoted by 0 and Saturday by 6.
                #     schedule.days=0,1,2
                # Timezone: Timezone information is further detailed into type, target index, zone
                # offset and region.
                #     schedule.timezone.type: either of TIMEZONE_TARGET, TIMEZONE_SPECIFIED,
                # TIMEZONE_REGION_SPECIFIED.
                #     schedule.timezone.targetIndex : specify the index of the target whose
                # timezone is to be used.
                #     schedule.timezone.zoneOffset : timezone offset.
                #     schedule.timezone.region : timezone region
                # Following is a complete schedule section, remove #  and populate the values for
                # submission:
                # scheule.frequency=ONCE
                # schedule.startTime=12-21-2012
                # schedule.endTime=12-21-2012
                # schedule.gracePeriod=10
                # schedule.months=
                # schedule.days=
                # schedule.timezone.type=TIMEZONE_TARGET
                # schedule.timezone.targetIndex=1
                # schedule.timezone.zoneOffset=
                # schedule.timezone.region=
```

```
# Variable: args
# Description: Parameters of the command to run on the target
variable.args=

# Variable: command
# Description: Command to run on the target
variable.command=

# Credential Usage: defaultHostCred
# Description:
cred.defaultHostCred.<target_name>:<target_type>=
```

describe_library_job

Describes a library job and gets its properties. The output can be redirected into a file.

Format

```
emcli describe_library_job
     -name=<job_name>
     [-owner=<job_owner>]
     [-verbose]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the library job to describe.

- **owner**

  Enterprise Manager administrator who owns this library job. If not provided, the current EM CLI logged-in administrator is assumed as the owner. The logged-in Enterprise Manager administrator must have at least the view privilege to describe a job.

- **verbose**

  Outputs a help template along with the properties.

Examples

Example 1

The following example describes the library job "myLibJob" owned by the logged-in Enterprise Manager administrator.

```
emcli describe_library_job -name=myLibJob
```

Example 2

The following example describes the library job "yourLibJob" owned by the Enterprise Manager administrator "admin1". The logged-in Enterprise Manager administrator has view privilege on this library job.

```
emcli describe_library_job -name=yourLibJob -owner=admin1
```

Example 3

```
emcli describe_library_job -name=MYJOB1

# Job Name : MYJOB1

# Current status of the job is ACTIVE.

# Job Type: OSCommand.
# This job type supports the following target types only :
host,j2ee_application,metadata_repository,oracle_apache,oracle_apm,oracle_
beacon,oracle_csa_collector,oracle_database,oracle_emd,oracle_emrep,oracle_
home,oracle_ias_farm,oracle_oms,oracle_oms_console,oracle_oms_pbs,weblogic_
domain,weblogic_j2eeserver.
```

```
job_target_list.1=adc2110610.us.oracle.com:host

# Variable: args
# Description: Parameters of the command to run on the target
variable.args=hello

# Variable: command
# Description: Command to run on the target
variable.command=echo

# Credential Usage: defaultHostCred
# Description:
cred.defaultHostCred.adc2110610.us.oracle.com:host=NAMED:SYSMAN:CRED1

schedule.frequency=REPEAT_BY_MINUTES
schedule.startTime=2012-02-01 01:01:01.0
schedule.endTime=2051-02-01 01:01:01.0
schedule.gracePeriod=-1
schedule.months=
schedule.days=
schedule.interval=1
schedule.timezone.type=TIMEZONE_TARGET
schedule.timezone.targetIndex=1
schedule.timezone.zoneOffset=0
schedule.timezone.region=
```

describe_patch_plan_input

Describes the input data of a patch plan.

Format

```
emcli describe_patch_plan_input
       -name=<name>
```

Parameters

- **name**

  Name of a given patch plan.

Example

```
emcli describe_patch_plan_input -name="plan_name
```

See Also

create_patch_plan
delete_patches
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

describe_procedure_input

Describes the input data of a deployment procedure or a procedure configuration.

Format

```
emcli describe_procedure_input
      [-procedure=<procedure_GUID>]
      [-name=<procedure_name_or_procedure_conf>]
      [-owner=<procedure_owner_or_procedure_config>]
      [-parent_proc=<procedure_of_procedure_config>]

[ ] indicates that the parameter is optional
```

Parameters

- **procedure**

  GUID of the procedure to execute.

- **name**

  Name of the procedure or procedure configuration.

- **owner**

  Owner of the procedure or procedure configuration.

- **parent_proc**

  Procedure of the procedure configuration. This applies to describe a procedure configuration when both a procedure and a procedure configuration have the same name.

Examples

```
emcli describe_procedure_input -procedure=16B15CB29C3F9E6CE040578C96093F61 >
describeDP.properties
```

diagchecks_deploy_status

Gets the status of diagnostic checks deployments against different target types.

Format

```
emcli diagchecks_deploy_status
      [-target_type=<type>]*

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target. You can specify multiple values.

diagchecks_deploy_tglist

Gets the target list for a particular deployment type for a target type.

Format

```
emcli diagchecks_deploy_tgtlist
      -target_type=<type>
      -deploy_type=<CURRENT|OLDER|MISSING|ALL>
      [-show_excludes]
```

[ ] indicates that the parameter is optional

Parameters

- **target_type**

  Type of target. You can specify multiple values.

- **deploy_type**

  Deployment type of either CURRENT, OLDER, MISSING, or ALL.

- **show_excludes**

  For targets where excludes have been set, print them.

disable_audit

Disables auditing for all user operations.

Format

```
emcli disable_audit
```

Example

The following example disables auditing for all operations.

```
emcli disable_audit
```

disable_sla

Disables an SLA for a target.

Format

```
emcli disable_sla
    -targetName=<target_name>
    -targetType=<target_type>
    -slaName=<SLA_name>
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

Examples

The following example disables an SLA named 'gold_sla' for target my_service (generic_service).

```
emcli disable_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla'1
```

disable_test

Disables monitoring of a Services test.

Format

```
emcli disable_test
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Test name.

- **testtype**

  Test type.

Examples

The following example disables the HTTP test named `MyTest` for the `generic_service` target named `MyTarget`.

```
emcli disable_test -name='MyTarget' -type='generic_service'
      -testname='MyTest' -testtype='HTTP'
```

discover_coherence

Discovers one or more Coherence clusters.

Format

```
emcli discover_coherence
     -input_file=coherence_discovery_file:file_path
     [-debug]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Fully-qualified path to a CSV-formatted file containing one line of details per Coherence cluster. The valid WebLogic version value is 10. The structure of the CSV file is as follows:

  ```
  <Management Node host machine name>,
            <Management Node listen port>,
            <Management Node username - optional>,
            <Management Node password - optional>,
            <Management Node service name - optional>,
            <Agent url>
  ```

  For example:

  ```
  host1.companyA.com,9910,,,,https://host1.companyA.com:3872/emd/main/,
  ```

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

The following example reads the my_clusters_info.csv file to determine the clusters to be added to Cloud Control.

```
emcli discover_coherence
  -input_file=coherence_discovery_file:"c:\emcli\my_clusters_info.csv
```

discover_fa

Discovers multiple Fusion Applications domains by reading the Fusion Applications domain discovery file and saving the host-wise discovered targets to the Agents provided in the Host Agent Mapping file. If the Host Agent mapping file is not provided, the local Agent (that is, the Agent on the same host as the target) is used to save/monitor the discoverd targets as well. If a local Agent is not found, the default discovery Agent is used to save/monitor the discoverd targets as well.

> **Note:** Although this verb supports discovering multiple Fusion instances at one time by adding all the details in one file, it is advisable to discover each Fusion instance separately using individual EM CLI discover_fa commands run multiple times.

Format

```
emcli discover_fa
      -input_file=fa_domain_discovery_file:file_path
      [-input_file=host_agent_mapping_file:file_path]
      [-input_file=pf_domain_cred_mapping_file:file_path]
      [-debug]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **input_file=fa_domain_discovery_file**

  Fully-qualified path to a CSV-formatted file containing one line of details per domain to be added. The valid WebLogic version value is 10. The structure of the CSV file is as follows:

  ```
  <WebLogic Server version>,
  <Administration Server host machine name>,
  <Administration Server listen port>,
  <Administration Server username>,
  <Administration Server password>,
  <External Parameters - al>,
  <JMX Protocol - required only if SSL enabled>,
  <JMX Service URL - required only if SSL enabled>,
  <Unique Domain Identifier>,
  <Agent url - al>,
  <Discover Down Servers - optional - Default if not specified is false starting
  <PS1. Before PS1 the default for this is true>,
  <Use Same Credentials for All Domains in the Fusion Instance - al - Default if
  <not specified is true>
  ```

  For example:

  ```
  10,mco01.mycompany.com,7001,weblogic,welcome1,,,,my_farm_
      01,https://mco01.mycompany.com:3872/emd/main/,,
  10,mco01.mycompany.com,7001,weblogic,welcome1,,,,my_farm_
      01,https://mco01.mycompany.com:3872/emd/main/,true,
  10,mco01.mycompany.com,7001,weblogic,welcome1,,,,my_farm_
      01,https://mco01.mycompany.com:3872/emd/main/,true,true
  10,mco01.mycompany.com,7001,weblogic,welcome1,,,,my_farm_
      01,https://mco01.mycompany.com:3872/emd/main/,false,true
  ```

- **input_file=host_agent_mapping_file**

   Fully-qualified path to a CSV-formatted file containing multiple lines of host system names where Managed Servers are to be monitored, and the Agent to be used to monitor each host's Managed Servers.

   For example:

   ```
   mycompany.com,https://mco01.mycompany.com:3872/emd/main
   ```

- **input_file=pf_domain_cred_mapping_file**

   Fully-qualified path to a CSV-formatted file containing multiple lines of WebLogic admin credentials for each domain of a fusion instance, where the credentials are different from those added in the fa_domain_discovery file.

   The same credentials are used for all the domains in a Fusion Application instance unless the credentials are overwritten in the pf_domain_cred_mapping file.

   For example:

   ```
   <UniqueKey - "<Fusion Instance
        Identifier><CommonDomainDisplayName>">,<Administration Server
        username>,<Administration Server password>,
   <UniqueKey - "<Fusion Instance
        Identifier>-<CommonDomainDisplayName>">,<Administration Server
        username>,<Administration Server password>,<Administration Server Host
        Name>

   Example:
   fi9-FS,weblogic12,welcome1,
   fi9-PRJ,faadmin,fusionfa1,
   fi9-PRC,faadmin,fusionfa1,adcdbb05.us.oracle.com
   fi9-PRC,,,adcdbb05.us.oracle.com
   ```

- **debug**

   Runs the verb in verbose mode for debugging purposes.

Examples

Example 1

The following example reads the my_domains_info.csv file to determine the Fusion Instances to be added to Cloud Control, reads the my_agent_mapping.csv file to determine which Agents should monitor which host's Managed Servers, and reads the my_domain_cred_mapping.csv file to determine which credentials are to be used to discover an individual product family.

```
emcli discover_fa
     -input_file=fa_domain_discovery_file:c:\emcli\my_domains_info.csv
     -input_file=host_agent_mapping_file:c:\emcli\my_agent_mapping.csv
     -input_file=pf_domain_cred_mapping_file:c:\emcli\my_domain_cred_mapping.csv
```

Example 2
```
emcli discover_fa -input_file=fa_domain_discovery_file:/tmp/emcli/
domain_discovery_file.txt -input_file=host_agent_mapping_file:/tmp/emcli/
host_agent_mapping_file.txt -debug
```

Example 3
```
emcli discover_fa -input_file=fa_domain_discovery_file:/tmp/emcli/
```

```
domain_discovery_file.txt -input_file=host_agent_mapping_filee:/tmp/emcli/
host_agent_mapping_file.txt -input_file=pf_domain_cred_mapping_file:/tmp/emcli/
pf_domain_cred_mapping_file.txt -debug
```

discover_wls

Purpose

Used to discover one or more version 7.x, 8.x, 9.x, and 10.x WebLogic Domains (along with Oracle Fusion Middleware 11g software deployed to it), and to specify which Management Agent should monitor which hosts' Managed Servers. Specifying which Management Agent should monitor which hosts' Managed Servers is a feature supported only with versions 9.x and 10.x of WebLogic Server. If you want to discover version 7.x or 8.x of WebLogic Server, you cannot specify which Management Agent to monitor which hosts' Managed Servers; the Management Agent used to perform discovery automatically monitors all WebLogic Servers within the version 7.x or 8.x domain.

Function

This verb discovers one or more Oracle WebLogic Server Domains. It reads a file labeled domain_discovery_file to discover WebLogic Server versions 7.x, 8.x, 9.x, and 10.x. Note that if you attempt to discover an already discovered WebLogic Server, the discovered WebLogic Server domain will be refreshed.

Requirements

To discover the WebLogic Server, the Administration Server must be up and running. After initial discovery or during refresh of domain membership, the Administration Server is not required to be up for general WebLogic Server monitoring. After initial discovery or during refresh of domain membership, the Managed Server is not required to be up for general WLS monitoring.

domain_discovery_file is required; discovery cannot occur without it. You must create the CSV (comma-separated values) formatted file before performing discovery. To save the discovered components (WebLogic Server versions 9.x and 10.x only) to a specific Management Agent for monitoring, the discover_wls verb reads a second file labeled host_agent_mapping_file. If host_agent_mapping_file does not exist, the Management Agent specified in domain_discovery_file that performs the actual discovery is used as the Agent that monitors all discovered targets.

> **Note:** For certified versions of WebLogic Server and Fusion Middleware 12c that Cloud Control 12.1 supports, refer to the Cloud Control Certification Matrix:
>
> ```
> https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=
> 412431.1
> ```

Format

```
emcli discover_wls
      -input_file=domain_discovery_file:file_path
      [-input_file=host_agent_mapping_file:file_path]
      [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file=domain_discovery_file**

  Fully-qualified path of the CSV (Comma-Separated Values) formatted file that contains one line of details per domain to be added. This is valid for WebLogic Server versions 7.x, 8.x, 9.x, and 10.x. Each line has the format shown for domain_discovery_file in the "File Structures" section below.

  Note the following points about the format of domain_discovery_file:

  Parameters —

  - The order of parameters is fixed. You must provide the parameters in the same order as shown for domain_discovery_file in the "File Structures" section below.

  - If you want to use a comma (,) in any of the parameters provided, you must escape the comma with a backslash as shown in the following example, in which a backslash precedes the comma in the password my,pwd:

    ```
    10,domain123.xyx.us,11990,weblogic,my\,pwd,,,farm_
    demo,https://myco01.mycompany.com:3872/emd/main
    ```

  Delimeters —

  - Use a comma (,) as the delimiter.

  - The total number of delimiters in each line is fixed and should be equal to 9 for WebLogic Server versions 7.x, 8.x, 9.x, and 10.x.

  - Delimiters must be present even if the corresponding parameter is not provided. See the last line for domain_discovery_file in the "File Structures" section below.

- **input_file=host_agent_mapping_file**

  Fully-qualified path of the CSV (Comma-Separated Values) formatted file that contains multiple lines of host system names where managed servers are to be monitored, and specifies the Management Agent used to monitor each host's managed servers. This is only valid for WebLogic Server versions 9.x and 10.x. Each line has the following format:

  ```
  <Discovered_target_host_machine_name>,<Agent_URL_to_save/monitor_the_host>
  ```

  For example:

  ```
  myco01.mycompany.com,https://myco01.mycompany.com:3872/emd/main
  myco02.mycompany.com,https://myco02.mycompany.com:3872/emd/main
  myco03.mycompany.com,https://myco03.mycompany.com:3872/emd/main
  ```

  Definitions for the parameters are as follows:

  - **Discovered_target_host_machine_name**

    Host machine with installed WebLogic Servers that need to be discovered. Use full host names, such as myco01.mycompany.com instead of myco01.

  - **Agent_URL_to_save/monitor_the_host**

    URL for the Management Agent to be used to monitor all discovered targets on the corresponding host.

  Note the following points about the format of host_agent_mapping_file:

  - Use a comma (,) as the delimiter.

- The total number of delimeters in each line is fixed and should be equal to 1.

- The order of parameters is fixed. You must provide the parameters in the same order as shown in the sample file structure in the "File Structures" section below. Discovered_target_host_machine_name and Agent_URL_to_save/monitor_the_host are both mandatory parameters.

■ **debug**

Runs this verb in verbose mode for debugging purposes.

File Structures

domain_discovery_file for WebLogic Server versions 7.x and 8.x

The following example shows the structure of a sample domain_discovery_file for WebLogic Server versions 7.x and 8.x. The same Management Agent is used to discover and save the targets. OPT signifies an optional parameter. The last entry shows the format when the optional parameters, Administration Server Home Directory and Trusted Keystore Filename, are not provided.

```
<WebLogic Server version>,<Administration Server Host>,<port>,<Administration
Server Username>,<password>,<Trusted Keystore Filename - OPT>,<Administration
Server Home Directory - OPT>,<Agent Host>,<Agent Host username>,<Agent Host
password>
```

Definitions for the parameters are as follows for WebLogic Server versions 7 and 8:

■ **WebLogic Server version**

Valid values are 7 or 8. The following example shows a sample entry in domain_discovery_file to discover WebLogic Server version 8:

```
8,myhost.us.mycompany.com,7001,weblogic,welcome1,,,myhost.us.mycompany.com,
oracle,welcome1
```

■ **Administration Server Host**

Full host name of the WebLogic Administration Server that needs to be discovered; for example, myhost.us.mycompany.com. This is a mandatory parameter.

■ **port**

Listen port of the WebLogic Administration Server.

■ **Administration Server Username**

Login user name for the WebLogic Administration Server.

■ **password**

Login password for the WebLogic Administration Server.

■ **Trusted Keystore Filename**

Absolute path of the Trusted Keystore Filename. This is required if the Administration Server's port is SSL enabled. If the Management Agent is on a different system than the WebLogic Server to be managed, you must manually copy the Trusted Keystore file to an accessible directory on the Management Agent system prior to discovery, and then use this path.

■ **Administration Server Home Directory**

Absolute path of the directory where the weblogic.jar file is located. If the Management Agent is on a different system than the Administration Server, you must manually copy the weblogic.jar file (located in the <WEBLOGIC_ HOME>/server/lib/ directory) to an accessible directory on the Management Agent system prior to discovery, and then use this path.

- **Agent Host**

  Host name of the Management Agent used to discover and monitor the targets.

- **Agent Host Username | Password**

  Credentials of the operating system user of the Management Agent host. These credentials are used to discover any Oracle WebLogic Server domains.

domain_discovery_file for WebLogic Server versions 9.x and 10.x

The following example shows the structure of a sample domain_discovery_file for WebLogic Server versions 9.x and 10.x. OPT signifies an optional parameter. The last entry shows the format when optional parameters External Parameters, JMX Protocol, JMX Service URL, and Management Agent URL are not provided.

```
<WebLogic Server version>,<Administration Server Host>,<port>,<username>,
<password>,<External Parameters - OPT>,<JMX Protocol - OPT>,
<JMX Service URL - OPT>,<Unique Domain Identifier>,<Agent URL -OPT>,<Discover Down
Servers - optional - Default if not specified is false (Starting PS1). Before PS1
default was true>,<Use Credential Store - optional - Default if not specified is
false>
```

Definitions for the parameters are as follows:

- **WebLogic Server version**

  Valid values are 9 or 10. The following example shows a sample entry in domain_discovery_file to discover WebLogic Server version 10:

  ```
  10,myco01.mycompany.com,7001,weblogic,welcome1,,,,soa_farm,
  https://myco02.mycompany.com:8723/emd/main
  ```

- **Administration Server Host**

  Full host name of the WebLogic Administration Server that needs to be discovered; for example, myco01.mycompany.com. This is a mandatory parameter.

- **port**

  Listen port of the WebLogic Administration Server.

- **username**

  Login user name for the WebLogic Administration Server.

- **password**

  Login password for the WebLogic Administration Server.

- **External Parameters**

  These parameters are passed to the Java process, which connects to the Administration Server. All of these parameters must begin with -D.

- **JMX Protocol**

  The Management Agent makes a JMX connection to the Administration Server to discover the domain's members. Valid values are t3, t3s, iiop, and iiops. If you do not provide a protocol, the t3 default is used.

- **JML Server URL**

  Makes a JMX connection to the Administration Server. If you do not specify this parameter, it is created based on the input parameters.

- **Unique Domain Identifier**

  Whatever you specify is used as the name for the Oracle Fusion Middleware Farm target type within Cloud Control. This must be a unique name across all farms that Oracle Enterprise Manager Cloud Control manages. The characters ' / ' , '' are not allowed.

- **Agent URL**

  URL for the Management Agent used to discover the targets. If you do not provide a value, the local Management Agent present on the target WebLogic Server is used. If a Management Agent is not found on the target WebLogic Server, an error is displayed.

- **Discover Down Servers**

  If this value is true, the servers that are down are discovered. If false, the servers that are down are not discovered.

- **Use Credential Store**

  If this value is set to true, the verb retrieves the WebLogic credentials from the credential store.

Examples

The following example reads the my_domains_info.csv file to determine the domains to be added to Cloud Control, and reads the my_agent_mapping.csv file to determine which Management Agents should monitor which host's managed servers.

```
emcli discover_wls
    -input_file=domain_discovery_file:\emcli\my_domains_info.csv
    -input_file=host_agent_mapping_file:\emcli\my_agent_mapping.csv
    -debug
```

The following example manually redirects the output of discover_wls to a file using standard output redirect.

```
emcli discover_wls input_file=domain_discovery_file:"<fully_qualified_path_of_
domain_discovery_file/domain_discovery_file.csv>" > /tmp/emcli/output_file.out
```

download_ats_test_databank_file

Downloads the specified databank file corresponding to the given ATS test. If no databank alias is specified, the command downloads all databanks for the test.

Format

```
emcli download_ats_test_databank_file
        -name=<target_name>
        -type=<target_type>
        -testname=<test_name>
        -testtype=<test_type>
        [-databankAlias=<databank_alias>]
        [-output_dir=<output_directory>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the target.

- **type**

  Name of the target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

- **databankAlias**

  Databank alias.

- **output_dir**

  Output directory. If the directory does not exist, it is created.

Examples

Example 1
The following example downloads the databank corresponding to alias1 for the specified test.

```
emcli download_ats_test_databank_file -name="Service Name"
                                      -type="generic_service"
                                      -testname="Test Name"
                                      -testtype="OATS"
                                      -databankAlias="alias1"
```

Example 2
The following example downloads all databanks corresponding to the specified test.

```
emcli download_ats_test_databank_file -name="Service Name"
                                      -type="generic_service"
                                      -testname="Test Name"
                                      -testtype="OATS"
```

download_ats_test_zip

Downloads the zip bundle corresponding to the specified ATS test.

Format

```
emcli download_ats_test_zip
        -name=<target_name>
        -type=<target_type>
        -testname=<test_name>
        -testtype=<test_type>
        [-output_dir=<output_directory>]
```

[ ] indicates that the parameter is optional

Parameters

- **name**

  Name of the target.

- **type**

  Name of the target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

- **output_dir**

  Output directory. If the directory does not exist, it is created.

Examples

```
emcli download_ats_test_zip -name="Service_Name"
                            -type="Generic_Service"
                            -testname="Test_Name"
                            -testtype="OATS"
                            -output_dir="outputDirectory"
```

enable_audit

Enables auditing for ALL and BASIC user operations. For other operations, see the update_audit_settings verb.

Format

```
emcli enable_audit
      [-level=basic]

[ ] indicates that the parameter is optional
```

Parameters

- **level=basic**

  Enables auditing for BASIC user operations.

Examples

Example 1

The following example enables auditing for all operations.

```
emcli enable_audit
```

Example 2

The following example enables auditing for LOGIN, LOGOUT, DB_LOGIN, and DB_LOGOUT.

```
emcli enable_audit -level=basic
```

enable_sla

Enables an SLA for a target.

Format

```
emcli enable_sla
    -targetName=<target_name>
    -targetType=<target_type>
    -slaName=<SLA_name>
    [-now]
    [-versionStart=<MM/dd/yyyy hh:mm a>]

[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

- **now**

  Enables the SLA now, or uses versionStart for a specific time.

- **versionStart**

  Specifies when the computation of the SLA should start.

Examples

Example 1

The following example immediately enables an SLA named 'gold_sla' for target my_service (generic_service).

```
emcli enable_sla
    -targetName='my_service' -targetType='generic_service'
    -slaName='gold_sla' -versionNum=2 -now
```

Example 2

The following example enables a SLA named 'gold_sla' for target my_service (generic_service). It becomes active and starts computing at '09/23/2012 3:30 PM'.

```
emcli enable_sla
    -targetName='my_service' -targetType='generic_service'
    -slaName='gold_sla' -versionNum=2 -versionStart='09/23/2012 3:30 PM'
```

enable_test

Enables monitoring of a Services test. It pushes the Service test collection to all the beacons.

Format

```
emcli enable_test
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Test name.

- **testtype**

  Test type.

Examples

The following example enables the HTTP test named `MyTest` for the `generic_service` target named `MyTarget`.

```
emcli enable_test -name='MyTarget' -type='generic_service'
      -testname='MyTest' -testtype='HTTP'
```

execute_hostcmd

Executes a host command across a set of targets.

Format

```
emcli execute_hostcmd
    -cmd=<host_command"
    -osscript=<script_to_be_executed>
    -targets=<name1:type1;name2:type2;...>
    -credential_set_name=<name>
    [-input_file=<parameter_tag:script_file>]

[ ] indicates that the parameter is optional
```

Parameters

- **cmd**

  Host_command can be any valid host command or group of host commands.

- **osscript**

  OS script to be executed with the cmd parameter.

- **targets**

  List of target-name, target-type pairs. The host command is executed across this list of Enterprise Manager targets. All targets must be of the type host or composite, which represents a group of targets. If it is a group, the group is expanded to extract all the host targets, and the host command is executed across these host targets.

- **credential_set_name**

  The credential_set_name parameter refers to the set name of the preferred credentials stored in the Enterprise Manager repository. If this parameter is not present, HostCredsNormal is used for executing host commands. For the host target type, two credential sets exist:

  – HostCredsNormal — Default unprivileged credential set for a host target

  – HostCredsPriv — Privileged credential set for a host target

    The credential set parameter can only be specified when the override credential parameters such as username and password are not present.

    If provided, the you must fully specify the override credential parameters. For host command, username and password must be specified together.

- **input_file**

  Used in conjunction with -osscript , this enables you to load the contents of an OS script. The -input_file specifies a mapping between a tag and a local file path. The tag is specified in lieu of actual osscript contents of the -osscript . The tag must not contain colons (:) or semi-colons (;).

Examples

Example 1

The following example executes the host command `ls -l;` against the target `stach.example.com:host` and host targets contained in the group `grp`. The stored `HostCredsPriv` preferred credentials are used for all the targets.

```
emcli execute_hostcmd
      -cmd="ls -l;"
      -credential_set_name="HostCredsPriv"
      -targets="stach.example.com:host;grp:composite"
```

Example 2

The following example loads the contents of the script `/scratch/dba_scripts/shellscript.sh` into the value of `-osscript` and executes it against target `reference.example.com:host` and host targets contained in the group `grp`. The stored `HostCredsNormal` preferred credentials are used for all the targets.

```
emcli execute_hostcmd
      -cmd="/bin/sh -s"
      -osscript="FILE"
      -input_file="FILE:/scratch/dba_scripts/shellscript.sh"
      -credential_set_name="HostCredsNormal"
      -targets="reference.example.com:host;grp:composite"
```

execute_sql

Executes a SQL command across a set of targets.

Format

```
emcli execute_sql
      -sql=<sql_command>
      -targets=<name1:type1;name2:type2;...>
      -credential_set_name=<name>
      [-input_file=<parameter_tag:script_file>]

[ ] indicates that the parameter is optional
```

Parameters

- **sql**

    "sql command" is a single SQL statement.

- **targets**

    List of target-name, target-type pairs. The SQL command executes across this list of Enterprise Manager targets. All targets must be of the type `oracle_database` or `composite`, which represents a group of targets. If it is a group, the group expands to extract all the database targets, and the SQL command is executed across these database targets.

- **credential_set_name**

    Refers to the set name of the preferred credentials stored in the Enterprise Manager repository. If this parameter is not present, the `DBCredsNormal` and `DBHostCreds` credential set is used for executing SQL commands. For each target type, several credential sets exist:

    - HostCredsNormal — Default unprivileged credential set for a host target

    - HostCredsPriv — Privileged credential set for a host target

    - DBHostCreds — Host credential set for an `oracle_database` target

    - DBCredsNormal — Default normal credential set for an oracle_database target

    - DBCredsSYSDBA — sysdba credential set for an `oracle_database` target

    You can only specify the `credential_set_name` parameter when the override credential parameters such as `[db_|host_]username` and `[db_|host_]password` are not present. If provided, the override credential parameters must be specified fully. For the SQL commands, `db_username`, `db_password`, `db_role`, `host_username`, and `host_password` must be present.

- **input_file**

    Used in conjunction with the `-sql` option, this option enables you to load the contents of a SQL script. The `-input_file` option specifies a mapping between a tag and a local file path. The tag is specified in lieu of an actual SQL command for the `-sql`. The tag must not contain colons (:) or semi-colons (;).

Examples

Example 1

The following example executes the SQL command `select * from sysman.mgmt_targets;` against the target `database:oracle_database` and database targets contained in the group `grp`. The stored SYSDBA preferred credentials are used for all the targets.

```
emcli execute_sql
     -sql="select * from sysman.mgmt_targets;"
     -credential_set_name="DBCredsSYSDBA"
     -targets="database:oracle_database;grp:composite"
```

Example 2

The following example loads the contents of the script `/scratch/dba_scripts/enterprise_schema.sql` into the value of `-sql`, and executes it against target `database:oracle_database` and database targets contained in the group `grp`. The stored SYSDBA preferred credentials are used for all the targets.

```
emcli execute_sql
     -sql="FILE"
     -input_file="FILE:/scratch/dba_scripts/enterprise_schema.sql"
     -credential_set_name="DBCredsSYSDBA"
     -targets="database:oracle_database;grp:composite"
```

Example 3

The following example executes the SQL command against "asm:osm_instance" and ASM targets contained in the group 'grp'. The SYSASM preferred credentials are used for all the targets.

```
emcli execute_sql
     -sql="select * from sysman.mgmt_targets;"
     -credential_set_name="ASMCredsSYSASM"
     -targets="asm:osm_instance;grp:composite"
```

export_compliance_group

Exports a compliance group definition and all of its element definitions given the name, author, and version.

Format

```
emcli export_compliance_group
      -name=<name>
      -author=<author>
      -version=<name>
      -output_file=<file>
```

Parameters

- **name**

  Name of the group to be exported.

- **author**

  Author of the group to be exported.

- **version**

  Version of the group to be exported.

- **output_file**

  Name of the exported file.

Examples

Example 1

```
emcli export_compliance_group \
      -name="foo" \
      -author="Jonas" \
      -version="99" \
      -output_file="$HOME/reports/group.xml"
```

export_compliance_standard_rule

Exports a rule to the specified files.

Format

```
export_compliance_standard_rule
  -name=<name>
  -target_type=<target_type>
  -output_file=<file>
```

Parameters

- **name**

  Name of the rule to be exported.

- **target_type**

  Target type of the rule to be exported.

- **output_file**

  Name of the exported file.

Examples

Example 1

```
emcli export_compliance_standard_rule \
     -name="foo" \
     -target_type="weblogic_j2eeserver" \
     -output_file="$HOME/reports/rule.xml"
```

export_masking_definition

Exports a masking definition in XML format.

Format

```
emcli export_masking_definition
    -definition_name=<masking_definition_name>
    [-path=file_path]
    [-file=file_name]

[ ] indicates that the parameter is optional
```

Parameters

- **definition name**

  Masking definition name.

- **path**

  Path for the file name to save the masking script. The file name is auto-generated.
  -path and -file are mutually exclusive. Only an absolute path is allowed.

- **file**

  File name to save the masking script. The file name must include the absolute
  path. -path and -file are mutually exclusive.

Output Columns

Success/Error messages.

Examples

Example 1

The following example exports the masking definition mask_hr_data to an XML file at
the specified path:

```
emcli export_masking_definition
    -definition_name=mask_hr_data
    -path=/tmp/
```

Example 2

The following example exports the masking definition mask_hr_data to an XML file
named abc.xml:

```
emcli export_masking_definition
    -definition_name=mask_hr_data
    -file=/tmp/abc.xml
```

export_metric_extension

Exports a metric extension archive file.

Format

```
emcli export_metric_extension
      -file_name=<metric_extension_archive_name>
      -target_type=<metric_extension_target_type>
      -name=<metric_extension_name>
      -version=<metric_extension_version>
```

Parameters

- **file_name**

  Name of the metric extension archive file to export into.

- **target_type**

  Target type of the metric extension.

- **name**

  Name of the metric extension.

- **version**

  Version of the metric extension to be exported.

Example

The following example creates an archive of a metric extension of a given target type, name, and version.

```
emcli export_metric_extension -file_name=<name of the metric extension archive>
-target_type=<target type of the metric extension> -name=<name of the metric
extension -version=<version of the metric extension>
```

export_report

Exports an Information Publisher report definition and all of its element definitions given its title and owner.

Format

```
emcli export_report
      -title=<report_title>
      -owner=<report_owner>
      -output_file=<file>
```

Parameters

- **title**

  Title of the report to export.

- **owner**

  The owner of the report to export. The logged-in emcli user must have view privilege for the report. Target names are not exported. The report is uniquely defined using title and owner, so both must be supplied.

- **output_file**

  Name of the exported file.

Examples

```
 emcli export_report
      -title=Maintenance_Report
      -owner=SHIFT1_OPERATOR
      -output_file=$HOME/reports/maint_report.xml
```

export_sla

Extracts the configuration details of an SLA into a local file. If you do not specify slaName and/or version, multiple SLA are exported to the same output file.

Format

```
emcli export_sla
    -targetName=<target_name>
    -targetType=<target_type>
    [-slaName=<SLA_name>]
    -output_file=<output_filename>

[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

- **output_file**

  Output file name of the template. If the file does not exist, it is created; if it already exists, it is overwritten. (This assumes that the extract operation was successful. If the operation fails, no files are created, and any existing files remain unchanged.)

Example

The following example creates an output file named 'service_sla.xml' that contains configuration details of the 'gold_sla' SLA for the target 'my_service'.

```
emcli export_sla
        -targetName='my_service'
        -targetType='generic_service'
        -slaName='gold_sla'
        -output_file='service_sla.xml'
```

export_standard

Exports a standard from the repository to an XML file.

Format

```
emcli export_standard
      -name=<name>
      -author=<author>
      -version=<name>
      -output_file=<file>
```

Parameters

- **name**

  Name of the standard to be exported.

- **author**

  Author of the standard to be exported.

- **version**

  Author of the standard to be exported.

- **output_file**

  Name of the exported file.

Example

```
emcli export_standard \
     -name=foo \
     -author=Curly \
     -version=99 \
     -output_file=$HOME/reports/standard.xml
```

export_template

Exports a monitoring template and also exports UDMs in the template. You can export a template to the file system in the form of an XML file, or you can print it on standard output in XML form.

Format

```
emcli export_template
    -name=<name>
    -target_type=<target_type>
    [-output_file=<file_for_exported_template>]
    [-archive]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the template. The name and target type uniquely identify a template.

- **target_type**

  Target type of the template.

- **output_file**

  Specifies the file to output the template. If not specified, the template prints to stdout.

- **archive**

  Indicates that the template must be exported as a zip file. When a Metric Extension is included in the template, this option is required to export the template as a zip file.

Examples

Example 1
The following example shows that template XML specified by name HOST_TEMP1 and target type host will be output to the screen.

```
emcli export_template -name=HOST_TEMP1 -target_type=host
```

Example 2
The following example shows that template XML specified by name HOST_TEMP1 and target type host will be created in the test.xml file.

```
emcli export_template -name=HOST_TEMP1 -target_type=host -output_file=test.xml
```

Example 3
The following example shows that the template archive specified by name HOST_TEMP1 and target type host will be created in the test.zip file.

```
emcli export_template -name=HOST_TEMP1  -target_type=host -output_file=test.zip
-archive
```

export_update

Exports a Self Update archive file from Enterprise Manager to the specified location.

Format

```
emcli export_update
      -id="internal id"
      -dir="dir"
      -omslocal
emcli export_update
      -id="internal id"
      -dir="dir"
      -host="hostname"
      [-credential_set_name="setname"] | -credential_name="name"
        -credential_owner="owner"

[ ] indicates that the parameter is optional
```

Parameters

- **id**

  Internal identification for the update to be exported.

- **dir**

  Complete path of the directory where the update is to be exported.

- **omslocal**

  Flag specifying that the directory is accessible from the OMS.

- **host**

  Target name for a host target where the update is to be exported.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target. Can be one of the following:

  HostCredsNormal — Default unprivileged credential set

  HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. You must specify this option along with the credential_owner option.

- **credential_owner**

  Owner of a named credential stored in the repository. You must specify this option along with the credential_name option.

Examples

Example 1

The following example exports the update archive file to /u01/common/. The directory must exist on the OMS host. In a multiple OMS setup, the request can be processed by any OMS, so the directory should be accessible from the OMS processing

the request. This usually means that the directory must be on a shared location accessible from all OMSes.

```
emcli export_update
        -id="914E3E0F9DB98DECE040E80A2C5233EB"
        -dir="/u01/common/"
        -omslocal
```

Example 2

The following example exports the update archive file to /u01/common/ on host host1.example.com. The host must be the managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The preferred unprivileged credentials for host host1.example.com are used to push the remote file.

```
emcli export_update
        -id="914E3E0F9DB98DECE040E80A2C5233EB"
        -dir="/u01/common/"
        -host="host1.example.com"
        -credential_set_name="HostCredsNormal"
```

Example 3

The following example exports the update archive file to /u01/common/ on host host1.example.com. The host must be the managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The named credentials "host1_creds" owned by user "admin1" are used to push the remote file.

```
emcli export_update
        -id="914E3E0F9DB98DECE040E80A2C5233EB"
        -dir="/u01/common/"
        -host="host1.example.com"
        -credential_name="host1_creds"
        -credential_owner="admin1"
```

extract_template_tests

Extracts variables and test definitions from a repository template into a local file.

Format

```
emcli extract_template_tests
      -templateName=<template_name>
      -templateType=<template_type>
      -output_file=<output_filename>
      [-encryption_key=<key>]

[ ] indicates that the parameter is optional
```

Parameters

- **templateName**

  Name of the template.

- **templateType**

  Type of template.

- **output_file**

  Name of the output file. If the file does not exist, it will be created; if it already exists, it will be overwritten. (This is assuming the extract operation was successful; if the operation fails, no files are created, and any existing files are left unchanged.)

- **encryption_key**

  Key to encrypt the file contents. The same key should be used to decrypt the file.

Example

The following example creates a file named `my_template.xml` containing the variable values and test definitions of the Web Application template `my_template`. The file contents are encrypted using the key `my_password`.

```
emcli extract_template_tests
      -templateName=my_template -templateType=website
      -output_file=my_template.xml -encryption_key=my_password
```

> **Note:**
>
> - The emcli user must have operator privilege on the repository template to perform this operation.
>
> - Beacon-related information is not exported to the file. In particular, the list of monitoring beacons, as well as any beacon-specific properties or thresholds, are not exported.
>
> - The values of password variables are not exported.

generate_masking_script

Generates a masking script for the given masking definition.

Format

```
emcli generate_masking_script
      -definition_name=masking_definition_name
      [-parameters=<name1:value1;name2:value2;...>]
      [-credential_name=cred_name]
      [-input_file=<parameter_tag:file_path>]
      [-script | -format=[name:<pretty|script|csv>];
                         [column_separator:column_sep_string];
                         [row_separator:row_sep_string];
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **definition_name**

  Name of the masking definition.

- **parameters**

  List of name-value pairs that represent the credentials required for connecting to the database instance. The supported parameters are db_username, db_password, and db_role.

- **credential_name**

  Name of the database credential. This parameter is mandatory when the db_username and db_password parameters are not specified.

- **input_file**

  Used in conjunction with the 'parameters' option, this enables you to store parameter values, such as username and password, in a separate file. This specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific parameter values of the 'parameters' . The tag must not contain colons (:) or semi-colons (;).

- **script**

  This is equivalent to -format='name: script'.

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

         –   format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

Output

Success or error messages as well as the impact report (if generated).

Examples

Example 1

The following example generates a script for the masking definition named mask_hr_data:

```
emcli generate_masking_script
     -definition_name=mask_hr_data
     -parameters=db_username:system;db_password:password;db_role:NORMAL
```

Example 2

The following example generates a script for the masking definition named mask_hr_data. The database password is read from the pwd.txt file.

```
emcli generate_masking_script
     -definition_name=mask_hr_data
     -parameters=PWD_FILE
     -input_file=PWD_FILE:pwd.txt
```

Example 3

The following example reads the database credentials from the named credential DB_NC and generates the masking script.

```
emcli generate_masking_script
     -definition_name=mask_hr_data
     -credential_name=DB_NC
```

get_agentimage

Gets the Management Agent image for the particular platform and version provided as inputs.

Format

```
emcli get_agentimage
   -destination=<download_directory>
   -platform="<platform>"
   [-version=<version>]
```

```
[ ] indicates that the parameter is optional.
```

Parameters

- **destination**

  Directory where you want to download the Management Agent software. Ensure that you have write permission on this location.

  If the destination directory is titled with two or more words separated by a space, enclose the directory name with double-quotes. For instance, if the destination directory is titled /tmp/linuxagentimage, enter the value as -destination="/tmp/linuxagentimage"

- **platform**

  Platform for which you want to download the software; this must match one of the platforms for which the software is available on the OMS host. Use the emcli get_supported_platforms command to determine this.

- **version**

  Version of the Management Agent software that you want to download. If you do not specify this , the version defaults to the OMS version.

Examples

```
emcli get_agentimage -destination=/tmp -platform=Linux_x86 -version=12.1.0.1.0
```

get_agentimage_rpm

Gets the Management Agent image for the Linux platform and version provided as inputs, then converts the image as rpm.

Format

```
emcli get_agentimage_rpm
  -destination=<download_directory>
  -platform=<platform>
  [-version=<version>]
```

```
[ ] indicates that the parameter is optional.
```

Parameters

- **destination**

  Directory where you want to download the .rpm file. Ensure that you have write permission on this location.

  If the destination directory is titled with two or more words separated by a space, enclose the directory name with double-quotes. For instance, if the destination directory is titled /tmp/linuxagentimage, enter the value as -destination="/tmp/linuxagentimage"

- **platform**

  Platform for which you want to download the .rpm file; this must match one of the platforms for which the software is available on the OMS host. Use the emcli get_supported_platforms command to determine this.

- **version**

  Version of the Management Agent for which you want to download the .rpm file. If you do not specify this , the version defaults to the OMS version.

Examples

```
emcli get_agentimage_rpm -destination=/tmp -platform=Linuxx86 -version=12.1.0.1.0
```

get_agent_properties

Displays Management Agent properties. You can use this command if you have view privilege for the Management Agent.

Format

```
emcli get_agent_properties
      -agent_name="<agent_target_name>"
      [-all]
      [-format="<format_name>"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **all**

  Shows all Management Agent properties. By default, only basic properties appear.

- **format**

  Format to display Management Agent properties. Valid values are pretty, script, and csv. By default, values are displayed in pretty format.

Examples

The following example shows all of the Management Agent properties in CSV format:

```
 emcli get_agent_properties -agent_name=agent.example.com:11850
      -all
      -format=csv
```

get_agent_property

Displays the value of a specific Management Agent property. You can use this command if you have view privilege for the Management Agent.

Format

```
emcli get_agent_property
     -agent_name=<agent_target_name>
     -name=<agent_property_name>
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **name**

  Name of the Management Agent property.

Examples

The following example shows the current value of the UploadInterval property in emd.properties.

```
emcli get_agent_property -agent_name=agent.example.com:11850
        -name=UploadInterval
```

get_aggregate_service_info

Gets time zone and availability evaluation function information of an aggregate's service instance.

Format

```
emcli get_aggregate_service_info
      -name=<name>
      -type=<type>
      [-noheader]
      [-script|-format=
              [name:<pretty|script|csv>];
              [column_separator:<sep_string>];
              [row_separator:<row_sep_string>]
      ]
```

[ ] indicates that the parameter is optional

Parameters

- **name**

  Aggregate service name.

- **type**

  Aggregate service type.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  - `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - `row_separator:"row_sep_string"` row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli get_aggregate_service_info -name=My_Name
    -type=aggregate_service
```

get_aggregate_service_members

Gets sub-services of an aggregate service instance.

Format

```
emcli get_aggregate_service_members
      -name=<name>
      -type=<type>
      [-noheader]
      [-script|-format=
                      [name:<pretty|script|csv>];
                      [column_separator:<sep_string>];
                      [row_separator:<row_sep_string>]
      ]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Aggregate service name.

- **type**

  Aggregate service type.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  – format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  – format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – format="name:csv" sets the column separator to a comma and the row separator to a newline.

  – format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli get_aggregate_service_members -name=My_Name
      -type=aggregate_service
```

get_blackout_details

Gets detailed information for a specified blackout.

Format

```
emcli get_blackout_details
      -name=<name>
      [-createdby=<blackout_creator>]
      [-noheader]
      [-script | -format=
                  [name:<pretty|script|csv>];
                  [column_separator:<column_sep_string>];
                  [row_separator:<row_sep_string>];
      ]
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the blackout.

- **createdby**

  Enterprise Manager user who created the blackout. The default is the current user.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

Output Columns

Status, Status ID, Run Jobs, Next Start, Duration, Reason, Frequency, Repeat, Days, Months, Start Time, End Time, TZ Region, TZ Offset

Examples

Example 1
The following example shows detailed information for blackout `blackout1` that the current user created.

```
emcli get_blackout_details -name=blackout1
```

Example 2
The following example shows detailed information for blackout `blackout1` that user `joe` created.

```
emcli get_blackout_details -name=blackout1 -createdby=joe
```

get_blackout_reasons

Lists all blackout reasons, one per line.

Format

```
emcli get_blackout_reasons
```

Examples

The following example lists all blackout reasons, one per line.

```
emcli get_blackout_reasons
```

get_blackout_targets

Lists targets for a specified blackout.

Format

```
emcli get_blackout_targets
      -name=<name>
      [-createdby=<blackout_creator>]
      [-noheader]
      [-script | -format=
                  [name:<pretty|script|csv>];
                  [column_separator:<column_sep_string>];
                  [row_separator:<row_sep_string>];
      ]
```

[ ] indicates that the parameter is optional

Parameters

- **name**

  Name of the blackout.

- **createdby**

  Enterprise Manager user who created the blackout. The default is the current user.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

Output Columns

Target Name, Target Type, Status, Status ID

Examples

Example 1
The following example lists targets in the blackout `blackout1` the current user created.

```
emcli get_blackout_targets -name=blackout1
```

Example 2
The following example lists targets in the blackout `blackout1` that user `joe` created.

```
emcli get_blackout_targets -name=blackout1 -createdby=joe
```

get_blackouts

Lists all blackouts or just those for a specified target or one or more hosts. Only the blackouts the user has privilege to view are listed.

Format

```
emcli get_blackouts
      [-target=<name1:type1> | -hostnames=<host1;host2;...>]
      [-noheader]
      [-script | -format=
                      [name:<pretty|script|csv>];
                      [column_separator:<column_sep_string>];
                      [row_separator:<row_sep_string>];
      ]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **target**

  Lists blackouts for this target. When neither this nor the -hostnames option is specified, all blackouts the user has privilege to view are listed.

- **hostnames**

  Lists blackouts that have a target on one of the specified hosts. The host name is just the target name part of the host target. For example, specify host.example.com, rather than host.example.com:host. When neither this nor the -target option is specified, all blackouts the user has privilege to view are listed.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format="name:script;column_separator:<column_sep_string>" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - format="name:script;row_separator:<row_sep_string>" row-separates the verb output by <row_sep_string>. Columns are separated by the tab character.

Output Columns

Name, Created By, Status, Status ID, Next Start, Duration, Reason, Frequency, Repeat, Start Time, End Time, Previous End, TZ Region, TZ Offset

Examples

Example 1

The following example shows all blackouts with some details.

```
emcli get_blackouts
```

Example 2

The following example shows all blackouts that cover the target `database2:oracle_database`.

```
emcli get_blackouts -target=database2:oracle_database
```

Example 3

The following example shows all blackouts that cover some target on host `myhost.example.com`.

```
emcli get_blackouts -hostnames=myhost.example.com
```

Example 4

The following example shows all blackouts that cover some target on host `myhost.example.com` or on host `yourhost.example.com`.

```
emcli get_blackouts -hostnames=myhost.example.com
-hostnames=yourhost.example.com
```

get_ca_info

Displays information about all of the Certificate Authorities (CA) created since the Cloud Control installation. It also displays the Management Agent names whose certificates are issued by the CA(s) when you specify the -details option. The following information is retrieved from the Cloud Control repository:

- Unique identifier of the Certificate Authority (CA) in the Cloud Control repository

- CA description

- CA creation date

- CA expiration date

- Number of Management Agents registered to this CA

- Number of secured Management Agents not registered to any CA

Format

```
emcli get_ca_info
        [-ca_id=<id1;id2;...>]
        [-details]

[ ] indicates that the parameter is optional
```

Parameters

- **ca_id**

  Specifies the Certificate Authority ID.

- **details**

  For each Certificate Authority, displays the list of Management Agent names whose certificates are issued by it.

Examples

The following example shows output for the CA with the ID of 2 specified.

```
emcli get_ca_info -ca_id=2

 Info about CA with ID: 2
 CA is configured
 DN: EMAILADDRESS=Enterprise.Manager@myomshost.mycompany.com,
 CN=myomshost.mycompany.com, OU=EnterpriseManager on myomshost.mycompany.com,
 O=EnterpriseManager on myomshost.mycompany.com, L=EnterpriseManager on
 myomshost.mycompany.com1, ST=CA, C=US, DC=com
 Serial# : 87539237298512593900
 Valid From: Mon Oct 25 17:01:15 UTC 2011
 Valid Till: Thu Oct 22 17:01:12 UTC 2020
 Number of Agents registered with CA ID 2 is 1

 Number of Agents to be re-secured, as OMS is secured using force_newca
    : 1
```

Regarding the force_newca option in the last line, the output shows that a new certificate was created with the ID of 2. Two Management Agents have been re-secured to be registered with this new certificate. The OMS running on myomshost.mycompany.com has been re-secured to be registered with the new

certificate created. There is still a Management Agent that needs to be secured to be registered to the new certificate. To retrieve the Management Agent name, you need to run the command "emcli get_ca_info -ca_id=2 -details," which is shown in the next example.

The following example displays the Management Agent names registered with the CA(s) for ID 2.

```
emcli get_ca_info -ca_id=2 -details

    Info about CA with ID: 2
    CA is configured
    DN: EMAILADDRESS=Enterprise.Manager@myomshost.mycompany.com,
     CN=myomshost.mycompany.com, OU=EnterpriseManager on myomshost.mycompany.com,
     O=EnterpriseManager on myomshost.mycompany.com, L=EnterpriseManager on
     myomshost.mycompany.com2, ST=CA, C=US, DC=com
    Serial# : 87539237298512593900
    Valid From: Mon Oct 25 17:01:15 UTC 2011
    Valid Till: Thu Oct 22 17:01:12 UTC 2020
    Number of Agents registered with CA ID 2 is 1
    usagent1.mycompany.com:20872

    Following Agents needs to be re-secured, as OMS is secured using force_newca
     :

    ukagent1.mycompany.com:1830
```

get_connection_mode

Gets the My Oracle Supporrt (MOS) connection mode. The two MOS connection modes are online and offline.

Format

```
emcli get_connection_mode
```

Parameters

None.

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

get_credtype_metadata

Prints credential-type information for a credential type. The verb prints credential column names. These column names should be used as parameter names for the create_named_credential and modify_named_credential verbs.

Format

```
emcli get_credtype_metadata
        -auth_target_type=<ttype>
        -cred_type=<name>
```

Parameters

- **auth_target_type**

  Authenticating target type.

- **cred_type**

  Credential type.

Examples

```
emcli get_credtype_metadata
        -auth_target_type=host
        -cred_type=HostCreds
```

get_duplicate_credential

Gets all the target-scoped named credentials that are the same as the given target-scoped named credential. Duplicate credentials are redundant. Named credentials can be managed better if reused. The same named credential can be reused for all of the usages.

Format

```
emcli get_duplicate_credential
      -cred_name=<cred_name>
      [-cred_owner=<cred_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **cred_name**

  Searches duplicates of this credential.

- **cred_owner**

  Owner of the credential, which defaults to the current user.

Example

The following example gets all of the credentials that are the same as the named credential MyOracleCredential and credential owner Joe.

```
emcli get_duplicate_credential
      -cred_name=MyOracleCredential
      -cred_owner=Joe
```

get_executions

Gets a list of executions of a submission using a submission GUID.

Format

```
emcli get_executions
   -instance=<Instance_GUID>
```

Parameters

- **instance**

  Displays all executions of a submission.

Output Columns

ExecutionGUID, Name, Status

Examples

```
emcli get_executions instance=16B15CB29C3F9E6CE040578C96093F61
```

get_ext_dev_kit

Downloads the Extensibility Development Kit to your local system. This verb has no parameters and only downloads a kit called edk.zip to the directory where you execute the command. After extracting the contents, you can use this kit to develop extensible components (plug-ins) of Enterprise Manager.

Format

```
emcli get_ext_dev_kit
```

Parameters

None.

get_group_members

Lists the members of the specified group.

Note that targets are only listed once, even though they can be in more than one sub-group of the group.

Format

```
emcli get_group_members
      -name=<name>
      [-type=<group>]
      [-depth=#]
      [-noheader]
      [-expand_non_groups]
      [-script | -format=
                 [name:<pretty|script|csv>];
                 [column_separator:<column_sep_string>];
                 [row_separator:<row_sep_string>];
      ]
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the group.

- **type**

  Group type: group. Defaults to `group`.

- **depth**

  Lists target members in sub-groups to the depth specified. The default is 1. When the depth is set to 0, no group target members are listed, and only the group's existence is verified. When the depth is set to -1, all group and sub-group target members are listed; in this case no groups appear in the output. Note that a target is listed at most once, even though it can be a member of several sub-groups.

- **noheader**

  Displays tabular information without column headers.

- **expand_non_groups**

  Lists members of aggregates and the aggregate target. By default, only sub-group target members are listed.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

- format="name:csv" sets the column separator to a comma and the row separator to a newline.

- format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

- row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Target Name, Target Type

Examples

Example 1

The following example lists the databases in group db2_group.

```
emcli get_group_members -name=db2_group
```

Example 2

The following example verifies that group my_hosts:group exists.

```
emcli get_group_members -name=my_hosts -depth=0
```

Example 3

The following example lists the unique targets in group my_group:group and its sub-groups.

```
emcli get_group_members -name=my_group -depth=-1
```

Example 4

The following example lists the unique targets in group my_group:group and its sub-groups/aggregates. The aggregate targets are also listed.

```
emcli get_group_members -name=my_group -depth=-1 -expand_non_groups
```

get_groups

Lists all groups.

Format

```
emcli get_groups
       [-noheader]
       [-script | -format=
                   [name:<pretty|script|csv>];
                   [column_separator:<column_sep_string>];
                   [row_separator:<row_sep_string>];
       ]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Target Name, Target Type

Example

The following example lists all groups.

```
emcli get_groups
```

get_instance_data

Downloads instance submission data.

Format

```
emcli get_instance_data
        [-instance=<instance_guid>]
        [-exec=<execution_guid>]
        [-name=<execution name>]
        [-owner=<execution owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

Output

Instance properties data.

Examples

```
emcli get_instance_data -instance=16B15CB29C3F9E6CE040578C96093F61 > data.xml
```

get_instance_status

Displays the procedure instance status identified by the GUID on the command line.

> **Tip:** See also get_instances on page 2-199 and get_job_execution_detail on page 2-200.

Format

```
emcli get_instance_status
        -instance=<instance_guid>
        [-exec=<execution_guid>]
        [-name=<execution_name>]
        [-owner=<execution_owner>]
        [-xml [-details] [-showJobOutput [-tailLength=<last_n_characters>]]]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Display the details of a procedure instance identified by the GUID number. You can find the GUID number by using the emcli get_instances command.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **xml**

  Shows the complete status of each of the steps in XML format.

- **details**

  Displays more details for the command output. This option also requires the -xml option.

- **showJobOutput**

  Shows the output or errors for the job execution steps. This option also requires the -xml option.

- **tailLength**

  Limits the number of characters in the job step output or error. This option also requires the -showJobOutput option.

  <Last N Characters> is a positive non-zero number until which the characters are chosen from the end of the job step output. The system sets the maximum permissible characters to dump. If you do not provide this option, the maximum permissible characters are dumped.

Output Columns

GUID, Procedure Type, Instance Name, Status

Examples

Example 1
The following example shows procedure details in CSV format:

```
emcli get_instance_status -guid=123456789012345678901234567890112
```

Example 2
The following example shows details in XML format:

```
emcli get_instance_status -guid=16B15CB29C3F9E6CE040578C96093F61 -xml -details
```

Example 3
The following example shows details in XML format with complete output:

```
emcli get_instance_status -guid=16B15CB29C3F9E6CE040578C96093F61 -xml -details
-showJobOutput
```

Example 4
The following example shows details in XML format with the last 1024 characters of
output:

```
 emcli get_instance_status -guid=16B15CB29C3F9E6CE040578C96093F61 -xml
-showJobOutput -tailLength=1024
```

See Also

get_instances

get_job_execution_detail

get_instances

Displays a list of procedure instances.

> **Tip:** See also get_procedure_types on page 2-216.

Format

```
emcli get_instances
      [-type=<procedure_type>]

[  ] indicates that the parameter is optional
```

Parameters

- **type**

  Displays all the procedure instances of type procedure_type.

Output Columns

Instance GUID, Execution GUID, Procedure Type, Instance Name, Status

Examples

Example 1
The following example lists all procedure instances:

```
emcli get_instances
```

Example 2
The following example lists all procedure instances of type 'PatchOracleSoftware':

```
emcli get_instances -type=PatchOracleSoftware
```

See Also

get_procedure_types

get_job_execution_detail

Displays details of a job execution.

Format

```
 emcli get_job_execution_detail
         -execution=<execution_id>
         [-xml [-showOutput [-tailLength=<length>]]]

[ ] indicates that the parameter is optional
```

Parameters

- **execution**

   Specifies that the ID of the job execution (execution_id) is the job execution ID.

- **xml**

   Shows the execution details as XML.

- **showOutput**

   Shows the output of the steps inside the job execution. You can only use this option in conjunction with the -xml option.

- **tailLength**

   Limits the display of the output to the number of characters from the end of the output. (length) is in characters. You can only use this option in conjunction with the -showOutput option. If you do not specify this option, a system-generated hard limit is enforced.

Examples

Example 1

The following example shows the details in CSV format:

```
emcli get_job_execution_detail -execution=123456789012345678901234567890901
```

Example 2

The following example shows the details in XML format:

```
emcli get_job_execution_detail -execution=1234567890123456789012345678901012 -xml
```

Example 3

The following example shows the details in XML format with complete output:

```
emcli get_job_execution_detail -execution=1234567890123456789012345678901012 -xml
-showOutput
```

Example 4

The following example shows the details in XML format with last N chars output:

```
emcli get_job_execution_detail -execution=1234567890123456789012345678901012 -xml
-showOutput -tailLength=1024
```

get_jobs

Lists existing jobs.

Format

```
emcli get_jobs
        [-job_ids=<ID1;ID2;...>]
        [-targets=<type1:name1;type2:name2;...>]
        [-status_ids=<status1;status2;...>]
        [-noheader]
        [-script | -format=
                    [name:<pretty|script|csv>];
                    [column_separator:<column_sep_string>];
                    [row_separator:<row_sep_string>];
        ]

[ ] indicates that the parameter is optional
```

Parameters

- **job_ids**

  Lists job IDs to use as the output filters.

- **targets**

  Lists targets (as name-type pairs) to use as the output filters.

- **status_ids**

  Lists numeric status IDs to use as the output filters.

  The numeric codes for all possible job statuses are as follows:

  - `SCHEDULED=1`

  - `EXECUTING(Running)=2`

  - `ABORTED(Failed Initialization)=3`

  - `FAILED=4`

  - `COMPLETED(Successful)=5`

  - `SUSPENDED_USER=6`

  - `SUSPENDED_AGENT_DOWN=7`

  - `STOPPED=8`

  - `SUSPENDED_LOCK=9`

  - `SUSPENDED_EVENT=10`

  - `SUSPENDED_BLACKOUT=11`

  - `STOP_PENDING=12`

  - `SUSPEND_PENDING=13`

  - `INACTIVE=14`

  - `QUEUED=15`

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Name, Type, ID, Execution ID, Scheduled, Completed, Status, Status ID, Owner, Target Type, Target Name

Examples

Example 1
The following example shows the jobs with the specified job IDs
12345678901234567890123456789012 and 09876543210987654321098765432100:

```
emcli get_jobs
     -job_ids=12345678901234567890123456789012, 09876543210987654321098765432100
```

Example 2
The following example shows all jobs run against a host target named
mainhost.example.com that are scheduled or have completed.

```
emcli get_jobs
     -status_ids=1,5
     -targets=mainhost.example.com:host
```

Example 3
The following example shows all jobs run against an Oracle database target named
payroll that have failed. Tabular output is generated using tabs as column
separators and newlines as row separators.

```
emcli get_jobs
     -status_ids=4
     -targets=payroll:oracle_database
     -script
```

get_job_types

Lists all the job types that can be used to create jobs and library jobs from EM CLI.

Format

```
emcli get_job_types
```

Parameters

None.

get_metering_data

Displays named credential details.

Format

```
emcli get_metering_data
    [-start_date=<start_date_in_mmddyyyy>]
    [-end_date=<end_date_in_mmddyyyy>]
    [-charge]
    [-cost_center=<cost_center_name>]
    [-target_type=<target_type>
    [-target_name=<target_name>]

[ ] indicates that the parameter is optional
```

Parameters

- **start_date**

  Report cycle start date in mmddyyyy. If you do not specify the report cycle start date, the latest report cycle is used.

- **end_date**

  Report cycle end date in mmddyyyy. If you do not specify the report cycle end date, the latest report cycle is used.

- **charge**

  Prints charge relation information.

- **cost_center**

  Cost center name. If you do not specify the cost center name, the logged in user is used as the cost center name.

- **target_type**

  If you do not specify the target type, all targets are used. Supported target types for this release are oracle_database, oracle_vm_guest, host, and weblogic_j2eeserver. This parameter is not valid without the target_name parameter.

- **target_name**

  If you do not specify the target name, all targets of a given target type are used. This parameter is not valid without the target_type parameter.

Examples

Example 1
The following example shows the latest cycle usage data for the logged in user.

```
emcli get_metering_data
```

Example 2
The following example shows usage data for the cost center cost_center_internal_name for the report cycle with a starting date of 10012011.

```
emcli get_metering_data
    -start_date=10012011
    -cost_center=cost_center_internal_name
```

Example 3

The following example shows charge data for the my_target Oracle Guest VM target
for cost center cost_center_internal_name for a report cycle with a starting date of
10012011.

```
emcli get_metering_data
    -start_date=10012011
    -cost_center=cost_center_internal_name
    -target_type=oracle_vm_guest
    -target_name=my_target
    -charge
```

get_metrics_for_stateless_alerts

For the specified target type, lists the metrics whose alerts are stateless and thus can be manually cleared. Both the metric name and metric internal name are provided in the output of this command. To clear the stateless alerts associated with the specified metric, use the clear_stateless_alerts verb.

Format

```
emcli get_metrics_for_stateless_alerts
      -target_type=type
```

Parameters

- **target_type**

  Internal target type identifier, such as host, oracle_database, oc4j, oracle_emrep, and oracle_emd.

Examples

The following example provides a list of all metrics for which stateless alerts can be manually cleared for any Oracle database (internal name for the target type is oracle_database).

```
emcli get_metrics_for_stateless_alerts -target_type=oracle_database
```

get_named_credential

Displays named credential details.

Format

```
emcli get_named_credential
        -cred_owner=<owner>
        -cred_name=<name>
        -out=<filename>
```

Parameters

- **cred_owner**

  Owner of the credential.

- **cred_name**

  Required credential name.

- **out**

  Output file name. The same file can be used as the input properties file for create_named_credential and modify_named_credential.

Examples

Example 1

The following example displays the details of the named credential NC1 owned by the current logged in user.

```
emcli get_named_credential -cred_name=NC1
```

Example 2

The following example displays the details of the named credential NC2 owned by the Administrator CREDS_MGR.

```
emcli get_named_credential -cred_name=NC2 -cred_owner=CREDS_MGR
```

get_oms_config_property

Gets the property value corresponding to the specified property name.

Format

```
emcli get_oms_config_property
      -property_name="propertyName"
      [-oms_name="omsName"]
      [-details]

[ ] indicates that the parameter is optional
```

Parameters

- **property_name**

  Name of the property whose value must be retrieved.

- **oms_name**

  Name of the mangaement server for which the property must be retrieved.

- **details**

  Specifies details about from where the property value has been derived, and also the global and default values for the property.

Examples

Example 1

The following example retrieves the property value set for the property name "propName" from the management server myhost:1159_Management_Service.

```
get_oms_config_property -property_name=propName -oms_name="myhost:1159_Management_
Service"
```

Example 2

The following example retrieves the property value set for the property name "propName" from all the management servers.

```
get_oms_config_property -property_name=propName
```

Example 3

The following example retrieves the property value set for the property name "propName" from all the management servers with details.

```
get_oms_config_property -property_name=propName -details
```

get_oms_logging_property

Gets the property value corresponding to the specified logging property name.

Format

```
emcli get_oms_logging_property
        -property_name="propertyName"
        [-oms_name="omsName"]
        [-details]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **property_name**

  Name of the logging property whose value must be retrieved.

- **oms_name**

  Name of the mangaement server for which the property must be retrieved.

- **details**

  Specifies details about from where the property value has been derived, and also the global and default values for the logging property.

Examples

Example 1

The following example retrieves the property value set for the property name "propName" from the management server myhost:1159_Management_Service.

```
get_oms_logging_property -property_name=propName -oms_name="myhost:1159_
Management_Service"
```

Example 2

The following example retrieves the property value set for the property name "propName" from all the management servers.

```
get_oms_logging_property -property_name=propName
```

get_on _demand_metrics

Gets a list of metrics that can be immediately collected with the collect_metric EMCLI verb. From this list, identify the metric you are interested in under the Metric Name column, then use its corresponding Metric Internal name in the collect_metric verb.

Format

```
emcli get_on_demand_metrics
     -target_type=type
     -target_name=name
```

Parameters

- **target_type**

  Internal target type identifier, such as host, oracle_database, oc4j, oracle_emrep, and oracle_emd.

- **target_name**

  Name of the target.

Examples

The following example shows a list of collectible metrics for the host target called hostname.example.com.

```
emcli get_on_demand_metrics -target_type=host -target_name=hostname.example.com
```

get_operation_plan_details

Provides detailed step-by-step information about the specified operation plan.

Format

```
emcli get_operation_plan_details
      -name="plan name"
```

Parameters

- **name**

  Name of the operation plan.

Examples

```
emcli get_operation_plan_details
      -name="BISystem1-switchover"
```

See Also

create_operation_plan

get_operation_plans

get_operation_plans

Lists all configured operation plans.

Format

```
emcli get_operation_plans
      -name=<operation plan_name>
      -operation=<operation_name>
```

Parameters

- **name**

  Name of the operation plan.

- **operation**

  Name of the operation, such as switchover, failover, start, or stop.

Output Columns

Plan Name, Operation Name, Configuration GUID

Examples

```
emcli get_operation_plans
      -name="austin-switchover"
      -operation="switchover"
```

See Also

create_operation_plan

submit_operation_plan

get_patch_plan_data

Gets patch plan user-editable data.

Format

```
emcli get_patch_plan_data
        -name="name"
```

Parameters

- **name**

  Name of a given patch plan.

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

Example

```
emcli get_patch_plan_data -name="plan_name"
```

get_plugin_deployment_status

Displays the status of a specific plug-in deployment or undeployment activity as well as the list of steps.

Format

```
emcli get_plugin_deployment_status
      [-plugin_id="plugin_id"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **plugin_id**

  ID of the plug-in for which you need to view the deployment/undeployment status. If not provided, the command shows the status of the latest plug-in being deployed, or the last one that was deployed or undeployed.

Examples

Example 1

Displays the status of the last plug-in deployment/undeployment activity.

```
emcli get_plugin_deployment_status
```

Example 2

The following example displays the status of the last deployment/undeployment activity of a specific plug-in.

```
emcli get_plugin_deployment_status -plugin_id=oracle.sysman.db
```

get_procedures

Gets a list of deployment procedures and pre-saved procedure configurations.

> **Tip:** See also get_procedure_types on page 2-216.

Format

```
emcli get_procedures [-type=<procedure_type>]
      [-parent_proc=<procedure_associate>]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Displays all the deployment procedures of type procedure_type.

- **parent_proc**

  Procedure associated with procedure configurations.

Output Columns

GUID, Procedure Type, Name, Display Type, Version, Created By, Procedure Name

See Also

get_procedure_types

get_procedure_xml

get_procedure_types

Gets the list of all deployment procedure types.

Format

```
emcli get_procedure_types
```

Output Column

Procedure Type

Example

The following example lists all procedure types:

```
emcli get_procedure_types
```

get_procedure_xml

Gets the deployment procedure XML file. XML is printed on standard output.

Format

```
emcli get_procedure_xml
       -procedure=[procedure_guid]
       [-name=<procedure_name>]
       [-owner=<procedure_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **procedure**

  Procedure GUID.

- **name**

  Procedure name.

- **owner**

  Procedure owner.

Output

Deployment procedure XML.

Examples

```
emcli get_procedure_xml -procedure=16B15CB29C3F9E6CE040578C96093F61 > proc.xml
```

get_reports

Returns a list of Information Publisher reports owned by or viewable by all users or a specified user. The output of this report is space-separated, quoted strings for the report title and owner, with each report on its own line.

Format

```
emcli get_reports
    [-owner="<report_owner>"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **owner**

  Enables listing of viewable reports that a specific Enterprise Manager owns.

Output

Space-separated quoted strings for the report title and owner, with each report on its own line.

Examples

```
emcli get_reports -owner=username
"report 1","username"
"example report 2","username"
```

```
emcli get_reports
"report A","username1"
"report 1","username2"
"example report 2","username2
```

get_resolution_states

Gets the list of existing resolution states used in managing incidents and problems. It also prints the display position of states. It does not list the fixed "New" and "Closed" resolution states.

Format

```
emcli get_resolution_states
```

Parameters

None.

Examples

The following example shows sample output for Incident defined states of OnHold, Waiting, and Processed, and Problem defined states of OnHold and Processed.

```
Incident resolution states
    5      OnHold
   10      Waiting
   25      Processed

Problem resolution states
    5      OnHold
   25      Processed
```

get_retry_arguments

Get arguments of failed steps that can be retried.

Format

```
emcli get_retry_arguments
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      [-stateguid=<state_guid>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **stateguid**

  State GUID.

Examples

```
emcli get_retry_arguments -instance=16B15CB29C3F9E6CE040578C96093F61


emcli get_retry_arguments -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168
```

get_siteguard_configuration

Provides the Site Guard configuration.

Format

```
emcli get_siteguard_configuration
      [-primary_system_name=<name_of_primary_system>]
      [-standby_system_name=<name_of_standby_system}]

[ ] indicates that the parameter is optional
```

Parameters

- **primary_system_name**

  Name of the primary system.

- **standby_system_name**

  Name of the standby system.

Output Columns

Primary System, Standby System(s)

Examples

```
emcli get_siteguard_configuartion
      -primary_system_name="BISystem1"
      -standby_system_name="BISystem2"
```

get_siteguard_credential_association

Lists the credential associations configured for a system.

Format

```
emcli get_siteguard_credential_association
      [-system_name=<name_of_system>]
      [-target_name=<name_of_target>]
      [-credential_type=<type_of_credential>]

[ ] indicates that the parameter is optional
```

Parameters

- **system_name**

  Name of the system.

- **target_name**

  Name of the target.

- **credential_type**

  Type of the credential, which can be HostNormal, HostPrivileged, WLSAdmin, or DatabaseSysdba.

Output Columns

Target Name, Credential Name, Credential Type

Examples

Example 1

```
emcli get_siteguard_credential_association
      -system_name="austin-system"
      -credential_type="HostNormal"
```

Example 2

```
emcli create_siteguard_credential_association
      -system_name="austin-system"
      -target_name="austin-database-instance"
      -credential_type="HostNormal"
```

See Also

create_siteguard_credential_association

update_siteguard_credential_association

get_siteguard_script_hosts

Lists the host or hosts associated with any script where the script is designated to run.

Format

```
emcli get_siteguard_script_hosts
     [-script_id=<script_id>]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **script_id**

  ID associated with the script.

Output Columns

Host Name

Examples

```
emcli get_siteguard_script_hosts
     -script_id="10"
```

See Also

create_siteguard_script

add_siteguard_script_hosts

get_siteguard_scripts

Obtains the Site Guard scripts associated with the specified system.

Format

```
emcli get_siteguard_scripts
      -system_name=<system_name>
      -operation=<operation_name>
      [-script_type=<type_of_script>]
      [-role=<role_of_system>]
```

Parameters

- **system_name**

    Name of the system.

- **operation**

    Name of the operation, such as switchover, failover, start, or stop.

- **script_type**

    Type of the script. For example: mount, unmount, pre-script, post-script, failover, or switchover.

- **role**

    Filters the scripts based on the role associated with the system. For example: Primary or Standby.

Output Columns

Script, ID, Type, Operation, Path, Role

Examples

Example 1

```
emcli get_siteguard_scripts
      -system_name="BISystem1"
      -operation="Switchover"
      -script_type="Pre-Script"
```

Example 2

```
emcli get_siteguard_scripts
      -system_name="austin-system"
      -operation="Switchover"
      -script_type="Pre-Script"
      -role="Primary"
```

See Also

create_siteguard_script

delete_siteguard_scripts

get_supported_platforms

Lists the platforms for which the Management Agent software is available on the OMS host.

Format

```
emcli get_supported_platforms
```

Output

The output of the command appears like the following example:

```
-------------------------
Platform Name : Linux x86
-------------------------
```

get_supported_privileges

Gets the list of available privileges in Enterprise Manager based on the type specified.

Format

```
emcli get_supported_privileges
        -type="ResourceType"
        [-noheader]
        [-script | -format=
                        [name:<pretty|script|csv>];
                        [column_separator:"column_sep_string"];
                        [row_separator:"row_sep_string"];
        ]
```

[ ] indicates that the parameter is optional

Parameters

- **type**

  Type of privileges to retrieve from Enterprise Manager. Possible values are:

  – ALL (default value)

  – SYSTEM

  – TARGET

  – JOB

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  – `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  – `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  – `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  – `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  – `row_separator:"row_sep_string"` row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Privilege Name, Privilege Type, Resource Class, Resource GUID Column, Resource ID Columns

get_system_members

Lists the members of the specified system.

Format

```
emcli get_system_members
      -name="name"
      [-type=<generic_system>]
      [-depth=#  (default 1)]
      [-noheader]
      [-script | -format=
            [name:<pretty|script|csv>];
            [column_separator:"column_sep_string"];
            [row_separator:"row_sep_string"];
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the system.

- **type**

  System type: `generic_system`. Defaults to `generic_system`.

- **depth**

  Lists target members in sub-systems to the specified depth. When the depth is set to 0, no system target members are listed, and only the system's existence is verified. When the depth is set to -1, all system and sub-system target members are listed.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  - `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - `row_separator:"row_sep_string"` row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Source Target Name, Member Target Name, Member Target Type, Level

Examples

Example 1

The following example lists the databases in system db2_system.

```
emcli get_system_members -name=db2_system
```

Example 2

The following example verifies that system my_system:generic_system exists.

```
emcli get_system_members -name=my_system -depth=0
```

Example 3

The following example lists the unique targets in system my_system:generic_system and its sub-systems.

```
emcli get_system_members -name=my_system -depth=-1
```

get_target_properties

Lists all the property names for the target type provided.

Format

```
emcli get_target_properties
        -target_type="target_type"
```

Parameters

- **target_type**

  Target type for which you want to list user-defined property names.

Examples

```
emcli get_target_properties -target_type="host"

Comment
Contact
Deployment Type
Line of Business
Location
Target properties fetched successfully
```

get_targets

Gets status and alert information for targets.

Format

```
emcli get_targets
       [-targets="[name1:]type1;[name2:]type2;..."]
       [-alerts]
       [-noheader]
       [-script | -format=
                  [name:<pretty|script|csv>];
                  [column_separator:"column_sep_string"];
                  [row_separator:"row_sep_string"];
       ]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **targets=name:type**

  Name or type can be either a full value or a pattern match using %. Also, name is optional, so the type can be specified alone.

- **alerts**

  Shows the count of critical and warning alerts for each target.

- **noheader**

  Display tabular output without column headers.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns

Status ID, Status, Target Type, Target Name, Critical, Warning

Examples

Example 1
The following example shows all targets. Critical and Warning columns are not included.

```
emcli get_targets
```

Example 2
The following example shows all targets. Critical and Warning columns are shown.

```
emcli get_targets
    -alerts
```

Example 3
The following example shows all `oracle_database` targets.

```
emcli get_targets
    -targets="oracle_database"
```

Example 4
The following example shows all targets whose type contains the string `oracle`.

```
emcli get_targets
    -targets="%oracle%"
```

Example 5
The following example shows all targets whose name starts with `databa` and type contains `oracle`.

```
emcli get_targets
    -targets="databa%:%oracle%"
```

Example 6
The following example shows status and alert information on the Oracle database named `database3`.

```
emcli get_targets
    -targets="database3:oracle_database"
    -alerts
```

get_test_thresholds

Shows test thresholds.

Format

```
emcli get_test_thresholds
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>
      [-script|-format=
                [name:"pretty|script|csv"];
                [column_separator:"sep_string"];
                [row_separator:"row_sep_string"]
      ]
```

```
[ ] indicates that the parameter is optional
```

Parameters

■ **name**

Target name.

■ **type**

Target type.

■ **testname**

Test name.

■ **testtype**

Test type.

■ **script**

This is equivalent to -format="name:script".

■ **format**

Format specification (default is `-format="name:pretty"`).

– `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

– `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

– `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

– format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

– row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli get_test_thresholds -name="Service Name"
                   -type="generic_service"
                   -testname="Test Name"
                   -testtype="HTTP"
```

get_unsync_alerts

Gets a list of alerts that are out-of-sync between the Management Agent and the repository for the specified target. You would typically use this command when you think that the Management Agent has not uploaded the latest alert to the repository. Under these circumstances, the repository would be out-of-sync with the Management Agent state.

Format

```
emcli get_unsync_alerts
      -target_type="type"
      -target_name="name"
```

Parameters

- **target_type**

  Internal target type identifier, such as host, oracle_database, emrep, and so forth.

- **target_name**

  Name of the target.

Output Column

Status

Examples

The following example shows the out-of-sync alert states for the host target type and abc.example.com target name:

```
emcli get_unsync_alerts -target_type=host -target_name=abc.example.com
```

get_unused_metric_extensions

Gets a list of metric extensions deployed to Agents, but not attached to any targets.

Format

```
emcli get_unused_metric_extensions
```

Parameters

None.

grant_license_no_validation

Grants licenses on a set of user-specified packs, or all packs to a set of user-specified targets, or all targets belonging to the input licensable target type.

For 11g database targets, you cannot enable or disable the Database Diagnostic and Tuning Packs through the user interface. You need to set the control_management_pack_access initialization parameter to manage your licenses. For information about this parameter, see the Enterprise Database Management chapter of *Oracle Enterprise Manager Licensing Information*.

> **Tip:** You can use this verb to grant licenses for standalone target types, such as hosts and databases, but you cannot use this verb to grant licenses for the parent Application Server (oracle_ias) target type, which has dependent target types of OC4J, Jserv, Web Cache, and so forth. To do this, use the grant_license_with_validation verb instead.
>
> For example, for pack ias_config and an Application Server target of AS1 with an associated dependent target of OC4J1, this verb grants a license to AS1, but this does not propagate to OC4J1.

Format

```
emcli grant_license_no_validation
      -type="target_type"
      [-targets="tname1;tname2;..."]
      [-packs="pack1;pack2;..."]
      [-file="file_name"]
      [-displayAllMessages]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Target type as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks. You can specify only one target type at a time; for example, -type="oracle_database".

- **targets**

  Targets should be specified in the following sequence:

  ```
  TargetName1;TargetName2;
  ```

  For example:

  ```
  -targets="database1;database2;database3;"
  ```

  The semi-colon (;) is the target separator.

  See the "Examples" section below for information about providing arguments for the targets .

- **packs**

  License packs should be specified in the following sequence:

  ```
  pack1;pack2;
  ```

For example:

```
-packs="db_diag;db_config;"
```

The semi-colon (;) is the pack separator.

See the "Examples" section below for information about providing arguments for the packs .

- **file**

  Specify the file name, including the complete path. For example:

  ```
  -file="/usr/admin1/db_license.txt"
  ```

  The file should contain the list of targets and packs according to the following cases:

  – If you only need to provide a list of targets, use the following format:

  ```
  targets=database1;database2;database3;
  ```

  – If you only need to provide a list of packs, use the following format:

  ```
  packs=db_diag;db_config;
  ```

  – If you need to provide a list of both targets and packs, use the following format:

  ```
  targets=database1;database2;database3;
  packs=db_diag;db_config;
  ```

- **displayAllMessages**

  Displays all messages. Only error messages are displayed by default. "=value" is not allowed on the command line.

Examples

Example 1 and Example 2 below grant licenses to specific packs for specific targets. In order to know which target types and pack names you can pass as arguments, you can use the view named mgmt_license_view to see a list of licensable targets, their target types, and the list of packs licensed on them.

To obtain this information, do the following:

1. Access SQL*Plus with your username and password, using sysman or other user that has access to sysman.mgmt_license_view.

2. Select a distinct pack name from sysman.mgmt_license_view, where:

```
target_type=<oracle_database>
```

The following example shows pack names for an Oracle database you specify as the target type.

```
PACK_NAME
------------------
db_config
provisioning
db_sadm
db_tuning
db_diag
provisioning_db
db_chgmgt
```

```
7 rows selected.
```

Based on this information, to grant a license to the database1 target for the db_chgmgt pack, you would enter the following command:

```
emcli grant_license_no_validation -type="oracle_database" -targets="database1"
-packs="db_chgmgt"
```

The only limitation of mgmt_license_view is that it only lists the packs for a target type where the pack is granted to at least one target of that type. That is, if the pack is not granted to any target of that type, mgmt_license_view cannot provide any information.

Example 1

The following example grants the license to the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli grant_license_no_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 2

The following example grants the license to the db_diag and db_config packs to all database targets in the setup:

```
emcli grant_license_no_validation -type="oracle_database"
        -packs="db_diag;db_config;"
```

Example 3

The following example grants the license to all packs (applicable to database targets) to database1, database2, and database3 targets in the setup:

```
emcli grant_license_no_validation -type="oracle_database"
          -targets="database1;database2;database3;"
```

Example 4

The following example grants the license to all packs (applicable to database targets) to all database targets in the setup:

```
emcli grant_license_no_validation -type="oracle_database"
```

Example 5

The following example uses a text file to pass targets and pack names as the argument. It grants the license to the db_diag and db_config packs to the database1, database2, and database3 targets (oracle_database target type):

```
emcli grant_license_no_validation -type="oracle_database"
          -file="/usr/admin1/db_license.txt"
             targets=database1;database2;database3;
                  packs=db_diag;db_config;
```

... where the content of the "/usr/admin1/license/db_license.txt" file is as follows:

```
targets=database1;database2;database3;
packs=db_diag;db_config;
```

grant_license_with_validation

Grants licenses on a set of user-specified packs, or all packs to a set of user-specified targets, or all targets belonging to the input licensable target type as per business rules.

For 11g database targets, you cannot enable or disable the Database Diagnostic and Tuning Packs through the user interface. You need to set the control_management_pack_access initialization parameter to manage your licenses. For information about this parameter, see the Enterprise Database Management chapter of *Oracle Enterprise Manager Licensing Information*.

> **Tip:** You can use this verb to grant licenses for standalone target types, such as hosts and databases, and you also use this verb to grant licenses for the parent Application Server (oracle_ias) target type, which has dependent target types of OC4J, Jserv, Web Cache, and so forth.
>
> For example, for pack ias_config and an Application Server target of AS1 with an associated dependent target of OC4J1, this verb grants a license to AS1 and also propagates to OC4J1 (and all other dependent targets associated with AS1).
>
> To grant licenses for only standalone target types, use the grant_license_no_validation verb.

Format

```
emcli grant_license_with_validation
      -type="target_type"
      [-targets="tname1;tname2;..."]
      [-packs="pack1;pack2;..."]
      [-file="file_name"]
      [-displayAllMessages]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Target type as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks. You can specify only one target type at a time; for example, -type="oracle_database".

- **targets**

  Targets should be specified in the following sequence:

  ```
  TargetName1;TargetName2;
  ```

  For example:

  ```
  -targets="database1;database2;database3;"
  ```

  The semi-colon (;) is the target separator.

  See the "Examples" section below for information about providing arguments for the targets .

- **packs**

  License packs should be specified in the following sequence:

```
pack1;pack2;
```

For example:

```
-packs="db_diag;db_config;"
```

The semi-colon (;) is the pack separator.

See the "Examples" section below for information about providing arguments for the packs .

- **file**

  Specify the file name, including the complete path. For example:

  ```
  -file="/usr/admin1/db_license.txt"
  ```

  The file should contain the list of targets and packs according to the following cases:

  - If you only need to provide a list of targets, use the following format:

    ```
    targets=database1;database2;database3;
    ```

  - If you only need to provide a list of packs, use the following format:

    ```
    packs=db_diag;db_config;
    ```

  - If you need to provide a list of both targets and packs, use the following format:

    ```
     targets=database1;database2;database3;
     packs=db_diag;db_config;
    ```

- **displayAllMessages**

  Displays all messages. Only error messages are displayed by default. "=value" is not allowed on the cmd line.

Examples

Example 1 and Example 2 below grant licenses to specific packs for specific targets. In order to know which target types and pack names you can pass as arguments, you can use the view named mgmt_license_view to see a list of licensable targets, their target types, and the list of packs licensed on them.

To obtain this information, do the following:

1. Access SQL*Plus with your username and password, using sysman or other user that has access to sysman.mgmt_license_view.

2. Select a distinct pack name from sysman.mgmt_license_view, where:

   ```
   target_type=<oracle_database>
   ```

The following example shows pack names for an Oracle database you specify as the target type.

```
PACK_NAME
-------------------
db_config
provisioning
db_sadm
db_tuning
db_diag
```

```
provisioning_db
db_chgmgt

7 rows selected.
```

Based on this information, to grant a license to the database1 target for the db_chgmgt pack, you would enter the following command:

```
emcli grant_license_with_validation -type="oracle_database" -targets="database1"
-packs="db_chgmgt"
```

The only limitation of mgmt_license_view is that it only lists the packs for a target type where the pack is granted to at least one target of that type. That is, if the pack is not granted to any target of that type, mgmt_license_view cannot provide any information.

Example 1

The following example grants a license to the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli grant_license_with_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 2

The following example grants a license to the db_diag and db_config packs to all database targets in the setup:

```
emcli grant_license_with_validation -type="oracle_database"
        -packs="db_diag;db_config;"
```

Example 3

The following example grants a license to all packs (applicable to database targets) to database1, database2, and database3 targets in the setup:

```
emcli grant_license_with_validation -type="oracle_database"
          -targets="database1;database2;database3;"
```

Example 4

The following example grants a license to all packs (applicable to database targets) to all database targets in the setup:

```
emcli grant_license_with_validation -type="oracle_database"
```

Example 5

The following example uses a text file to pass targets and pack names as the argument. It grants a license to the db_diag and db_config packs to the database1, database2, and database3 targets (oracle_database target type):

```
emcli grant_license_with_validation -type="oracle_database"
          -file="/usr/admin1/db_license.txt"
            targets=database1;database2;database3;
                  packs=db_diag;db_config;
```

where the content of the "/usr/admin1/license/db_license.txt" file is as follows:

```
targets=database1;database2;database3;
packs=db_diag;db_config;
```

grant_privs

Grants the privileges to the existing Enterprise Manager user or Enterprise Manager Role.

Note: To replace an existing Enterprise Manager administrator role, use the modify_role verb.

Format

```
emcli grant_privs
      -name="username|rolename"
      -privilege="name[;secure_resource_details]"
      [-grant_all_targets_on_host="yes|no"]
      [-separator=privilege="sep_string"]
      [-subseparator=privilege="subsep_string"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  User name or role name to which privileges will be assigned.

- **privilege**

  Privilege to be granted to the Enterprise Manager user or role. You can specify this parameter more than once.

  Specify secure_resource_details as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1
  [:resource_column_name2=resource_column_value2]..]"
  ```

  Optionally, you can drop resource column names from this parameter if you provide resource information in the order described by emcli get_supported_ privileges. See the "See Also" section below for more information.

- **grant_all_targets_on_host**

  Indicates if the privilege needs to be granted on all targets of the host specified as part of the privilege parameter. The default value is no.

- **separator=privilege**

  Specify a string delimiter to use between name-value pairs for the value of the -privilege option. The default separator delimiter is a semi-colon (;).

- **subseparator=privilege**

  Specify a string delimiter to use between the name and value in each name-value pair for the value of the -privilege option. The default subseparator delimiter is a colon (:).

Examples

Example 1

The following example grants these privileges to user1:

- Privilege to use any beacon

- Full control of the jobs with ID 923470234ABCDFE23018494753091111

- Full control on the target host1.example.com:host

- Full control on the credential cred1:user2

- View Privilege on target with ID 123451234ABCDFE23018494753092222

```
emcli grant_privs
      -name="user1"
      -privilege="USE_ANY_BEACON"
      -privilege="FULL_JOB;923470234ABCDFE23018494753091111"
      -privilege="FULL_TARGET;TARGET_NAME=host1.example.com:TARGET_TYPE=host"
      -privilege="FULL_CREDENTIAL;CRED_NAME=cred1:CRED_OWNER=user2"
      -privilege="FULL_CREDENTIAL;CRED_GUID=123451234ABCDFE23018494753092222"
```

Example 2

The following example grants target privileges to EM Role : Role1:

```
emcli grant_privs
      -name="Role1"
      -privilege="FULL_TARGET;TARGET_NAME=host1.example.com:TARGET_TYPE=host"
```

Example 3

The following example grants FULL_TARGET privilege on all targets on host host1.example.com to user1.

```
emcli grant_privs
      -name="user1"
      -privilege="FULL_TARGET;TARGET_NAME=host1.exemple.com:TARGET_TYPE=host"
      -grant_all_targets_on_host="yes"
```

Example 4

The following example uses the separator and subseparator parameters to grant FULL_TARGET privilege on host1.us.oracle.com to user1.

```
emcli grant_privs
      -name="user1"
      -privilege="FULL_TARGET->TARGET_NAME=host1.us.oracle.com@@TARGET_TYPE=host"
      -separator=privilege="->"
      -subseparator=privilege="@@"
```

See Also

To see the complete list of privileges and resource column names, execute the following command:

```
emcli get_supported_privileges
```

To see the list of SYSTEM privileges, which do require resource information:

```
emcli get_supported_privileges -type=SYSTEM
```

To see the list of TARGET privileges:

```
emcli get_supported_privileges -type=TARGET
```

To see the list of JOB privileges:

```
emcli get_supported_privileges -type=JOB
```

grant_roles

Grants roles to an existing Enterprise Manager user or Enterprise Manager role.

Format

```
emcli grant_roles
    -name="username|rolename"
    [-roles="role1;role2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  User name or role name to which roles will be assigned.

- **roles**

  Roles that will be granted to an Enterprise Manager user or role. You can specify this option more than once.

Examples

```
emcli grant_roles
    -name="user1"
    -roles="SUPER_USER"

emcli grant_roles
    -name="Role1"
    -roles="BLACKOUT_ADMIN;MAINTAIN_TARGET"
```

help

Shows a summary of all verbs or command-line help for individual EM CLI verbs.

> **Note:** EM CLI must be set up and configured before command line help is available for all verbs.

Format

```
emcli help [verbname]

[ ] indicates that the parameter is optional
```

Parameters

None.

Examples

Example 1

The following example provides an overview for all available verbs:

```
emcli help
```

Example 2

The following example provides the description, syntax, and usage examples for the add_target verb:

```
emcli help add_target
```

ignore_instance

Ignores a failed step. An instance cannot be ignored when it completes, completes with an error, is suspended, or is stopped.

Format

```
emcli ignore_instance
      -instance=<instance_guid>
      [exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      [-stateguid=<state_guid>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  Instance GUID.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **stateguid**

  Comma-separated list of state GUIDs.

Example

```
emcli ignore_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168
```

import_appreplay_workload

Imports a workload metadata XML file and creates a new application replay workload object. A Workload metadata XML file, which is stored in the workload root directory, is automatically generated as part of the workload capture process. The XML file contains a pointer to the actual raw captured workload data files. If you are importing a workload captured by one Enterprise Manager system to another, make sure the workload storage location specified in the XML file is reachable and contains the workload data files.

Format

```
emcli  import_appreplay_workload
       -input_file=template:<input_filename>
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Fully-qualified path to a workload metadata XML file. The workload XML file is automatically created during capture. However, you may need to make necessary changes to the XML file before you import. For example, you may want to change the workload name in the exported file and rename the XML file to match the workload name. You may also need to modify the storage locations to point to where the workload data files are located if you have moved the captured data files.

import_compliance_object

Imports a compliance object into the repository.

Format

```
 import_compliance_object
       -files=file1;file2;...  [-overwrite] [-deep]

[ ] indicates that the parameter is optional
```

Parameters

- **files**

  Files to be imported.

- **overwrite**

- **deep**

Examples

```
emcli import_compliance_object
      -files=file1.xml;file2.xml -overwrite
```

import_masking_definition

Imports a masking definition from the specified XML file.

Format

```
emcli import_masking_definition
    -file=/tmp/file_name.xml
```

Parameters

- **file**

  Path of the file containing the masking definition in XML format.

Output

Success or error messages.

Examples

The following example imports the masking definition from the hr_mask.xml file.

```
emcli import_masking_definition
      -file=/tmp/hr_mask.xml
```

import_metric_extension

Imports a metric extension archive file.

Format

```
emcli import_metric_extension
      -file_name=<metric_extension_archive>
      -rename_as=<metric_extension_to_import_as>
```

Parameters

- **file_name**

  Name of the metric extension archive file to be imported.

- **rename_as**

  Imports the metric extension using the specified name, replacing the name given in the archive.

Examples

The following example imports the masking definition from the hr_mask.xml file.

```
emcli import_metric_extension
    -file_name=<file name>
    -rename_as=<metric extension name>
```

import_report

Imports one or more Information Publisher report definitions from an XML file(s) using the title in the XML file and the currently logged-in CLI user as the owner of the report. If the report/owner already exists, the operation fails for this report with an accompanying error message. (You can override this with the -force option.) The report will be changed to a just-in-time report with the target type from the exported report.

You will need to edit schedules and access privileges using the Enterprise Manager user interface. The system enforces title/owner uniqueness, so an error occurs if a report with the same title and owner already exists.

Format

```
emcli import_report
      -files="file1;file2;..."
      [-force]
```

[ ] indicates that the parameter is optional

Parameters

- **files**

  List of path/file name(s) of XML file(s) that contain valid report definition(s).

- **force**

  First delete the report (and all jobs and saved copies) if a report with the same title/owner exists.

Examples

```
emcli import_report
     -files="$HOME/reports/maint_report1.xml;$HOME/reports/file2.xml"
```

import_sla

Imports an SLA configuration XML file for a target. This verb provides the functionality of creating a new SLA, creating a new version, and creating a new copy.

> **Note:** The XML file can only contain one SLA to be imported; that is, when export_sla has successfully exported a file when slaName and version are specified.

> **Note:** The target must have the metrics required by the SLA template's SLI. If the template's SLI calls for a metric not found in the target, the SLI cannot be created.

Format

```
emcli import_sla
      -targetName=<target name>
      -targetType=<target type>
      -input_file=slaTemplate:<input filename>
      [-slaName=<SLA name>]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **input_file**

  Name of the input file. There can only be one SLA root node in the XML document.

- **slaName**

  Specifying this name overrides the name contained in the SLA template XML file. This effectively creates a new SLA version series starting with version 1.

Examples

The following example creates an SLA named 'gold_sla' for the target my_service (generic_service).

```
emcli import_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla' -input_file=slaTemplate:'service_sla.xml'
```

import_template

Imports a monitoring template from an XML or zip file. The resulting definition is saved in the repository.

Format

```
emcli import_template
      -files="file1;file2;..."
```

Parameters

- **files**

  Path/file name of an XML file, which contains a valid template definition. You can specify multiple files with this option by separating each file with a semi-colon (;).

Examples

Example 1

The following example imports a template from `template.xml`.

```
emcli import_template -files="template.xml"
```

Example 2

The following example imports three templates — one from each of the files specified.

```
emcli import_template -files="e1.xml;e2.xml;e3.xml"
```

Example 3

The following example imports a template from the template.zip file along with any metric extensions.

```
emcli import_template -files="template.zip"
```

import_update

Imports a Self Update archive file into Enterprise Manager. Upon successful import, the update is displayed on the Self Update Home in downloaded status for further action.

Format

```
emcli import_update
        -file="file"
        -omslocal
emcli import_update
        -file="file"
        -host="hostname"
        [-credential_set_name="setname"] | -credential_name="name"
        -credential_owner="owner"

[ ] indicates that the parameter is optional
```

Parameters

- **file**

  Complete path name of the update archive file.

- **omslocal**

  Flag specifying that the file is accessible from the OMS.

- **host**

  Target name for a host target where the file is available.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target. Can be one of the following:

  HostCredsNormal — Default unprivileged credential set

  HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. You must specify this along with the credential_owner .

- **credential_owner**

  Owner of a named credential stored in the repository. You must specify this option along with the credential_name option.

Examples

Example 1

The following example imports the file update1.zip. The file must be present on the OMS host. In a multiple OMS setup, any OMS can process the request, so the file should be accessible from the OMS processing the request. This usually means that the file must be kept on a shared location accessible from all OMSes.

```
emcli import_update
        -file="/u01/common/update1.zip"
        -omslocal
```

Example 2

The following example imports the file update1.zip that is present on the host host1.example.com. The host must be a managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The preferred unprivileged credentials for host host1.example.com are used to retrieve the remote file.

```
emcli import_update
        -file="/u01/common/update1.zip"
        -host="host1.example.com"
        -credential_set_name="HostCredsNormal"
```

Example 3

The following example imports the file update1.zip that is present on the host host1.example.com. The host must be a managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The named credentials "host1_creds" owned by user "admin1" are used to retrieve the remote file.

```
emcli import_update
        -file="/u01/common/update1.zip"
        -host="host1.example.com"
        -credential_name="host1_creds"
        -credential_owner="admin1"
```

import_update_catalog

Imports a Self Update master catalog file when Enterprise Manager is configured in offline mode. All updates present in the catalog are processed, and the applicable updates are displayed on the Self Update Home for further action.

Format

```
emcli import_update_catalog
        -file="file"
        -omslocal
  emcli import_update_catalog
        -file="file"
        -host="hostname"
        [-credential_set_name="setname"] | -credential_name="name"
          -credential_owner="owner"
```

[ ] indicates that the parameter is optional

Parameters

- **file**

  Complete path name of the self update catalog file.

- **omslocal**

  Flag specifying that the file is accessible from the OMS.

- **host**

  Target name for a host target where the file is available.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target. Can be one of the following:

  HostCredsNormal — Default unprivileged credential set

  HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. You must specify this along with the credential_owner option.

- **credential_owner**

  Owner of a named credential stored in the repository. You must specify this option along with the credential_name option.

Examples

Example 1
The following example imports the master catalog file p9984818_121000_Generic.zip. The file must be present on the OMS host. In a multiple OMS setup, the request can be processed by any OMS, so the file should be accessible from the OMS processing the request. This usually means that the file must be kept on a shared location accessible from all OMSes.

```
emcli import_update_catalog
        -file="/u01/common/p9984818_121000_Generic.zip"
        -omslocal
```

Example 2

The following example imports the master catalog file p9984818_121000_Generic.zip that is present on the host host1.example.com. The host must be a managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The preferred unprivileged credentials for host host1.example.com are used to retrieve the remote file.

```
emcli import_update_catalog
        -file="/u01/common/p9984818_121000_Generic.zip"
        -host="host1.example.com"
        -credential_set_name="HostCredsNormal"
```

Example 3

The following example imports the master catalog file p9984818_121000_Generic.zip that is present on the host host1.example.com. The host must be a managed host target in Enterprise Manager, and the Management Agent on this host must be up and running. The named credentials "host1_creds" owned by user "admin1" are used to retrieve the remote file.

```
emcli import_update_catalog
        -file="/u01/common/p9984818_121000_Generic.zip"
        -host="host1.example.com"
        -credential_name="host1_creds"
        -credential_owner="admin1"
```

list_active_sessions

Lists active sessions on all OMSes in the environment. By default, the verb prints a summary for each OMS.

Format

```
emcli list_active_sessions
        [-details
        [-table]
        [-script]
        [-format=name:value;name:value]
        [-noheader]]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **details**

  Displays active user sessions on each OMS. The output format is non-tabular.

- **table**

  Prints details in table format.

- **script**

  Prints output that can be processed by script.

- **format**

  Supports the following name/value pairs:

  csv — Output will be comma-separated

  script — Output will be in a format that can be processed by script. You can also specify row_separator and column_separator.

- **noheader**

  Skips the header.

Examples

```
emcli list_active_sessions
    emcli list_active_sessions -details
    emcli list_active_sessions -details -table
    emcli list_active_sessions -details -table -script
    emcli list_active_sessions -details -table -script -noheader
    emcli list_active_sessions -details -table -format="name:csv"
    emcli list_active_sessions -details -table -format="name:script;row_
separator:@@;column_separator:!"
```

list_aru_languages

Lists ARU language information.

Format

```
emcli list_aru_languages
        [-name="language_name" | -id="language_id"]
        [-noheader]
        [-script | -format=
                   [name:<pretty|script|csv>];
                   [column_separator:"column_sep_string"];
                   [row_separator:"row_sep_string"];
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Language name.

- **id**

  Language ID.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  - `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_aru_languages
    emcli list_aru_languages -noheader
    emcli list_aru_languages -name="language name" -format="name:pretty"
    emcli list_aru_languages -id="language id" -format="name:script"
```

See Also

create_patch_plan

delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_aru_platforms

Lists ARU platform information.

Format

```
emcli list_aru_platforms
        [-name="platform_name" | -id="platform_id"]
        [-noheader]
        [-script | -format=
                        [name:<pretty|script|csv>];
                        [column_separator:"column_sep_string"];
                        [row_separator:"row_sep_string"];
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Platform name.

- **id**

  Platform ID.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  - `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_aru_platforms
    emcli list_aru_platforms -noheader
    emcli list_aru_platforms -name="platform_name" -format="name:pretty"
    emcli list_aru_platforms -id="platform id" -noheader -format="name:script"
```

See Also

create_patch_plan

delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_aru_products

Lists ARU product information.

Format

```
emcli list_aru_products
        [-name="product_name" | -id="product_id"]
        [-noheader]
        [-script | -format=
                        [name:<pretty|script|csv>];
                        [column_separator:"column_sep_string"];
                        [row_separator:"row_sep_string"];
        ]
```

`[ ]` indicates that the parameter is optional

Parameters

- **name**

  Product name.

- **id**

  Product ID.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  - `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - `row_separator:"row_sep_string"` row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_aru_products
    emcli list_aru_products -id="product id"
    emcli list_aru_products -name="product name"
    emcli list_aru_products -id="product id" -noheader
    emcli list_aru_products -id="product id" -noheader -script
    emcli list_aru_products -id="product id" -noheader  -format="name:pretty"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_aru_releases

Lists ARU release information.

Format

```
emcli list_aru_releases
      [-name="release_name" | -id="release_id" | -productId="product_id"]
      [-noheader]
      [-script | -format=
                        [name:<pretty|script|csv>];
                        [column_separator:"column_sep_string"];
                        [row_separator:"row_sep_string"];
      ]
```

[ ] indicates that the parameter is optional

Parameters

- **name**

    Release name.

- **id**

    Release ID.

- **productId**

    Product ID.

- **noheader**

    Displays tabular information without column headers.

- **script**

    This option is equivalent to -format="name:script".

- **format**

    Format specification (default is -format="name:pretty").

    - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

    - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

    - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

    - `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

    - `row_separator:"row_sep_string"` row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_aru_releases
      emcli list_aru_releases -noheader
      emcli list_aru_releases -name="release_name" -format="name:pretty"
      emcli list_aru_releases -id="release id" -format="name:script"
```

```
emcli list_aru_releases -productId="product id" -noheader -format="name:csv"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_diagcheck_exclusions

Gets the list of diagnostic check exclusions defined for a target type.

Format

```
emcli list_diagcheck_exclusions
       -target_type="type"
```

Parameters

- **target_type**

  Type of target.

list_diagchecks

Gets the list of diagnostic check exclusions defined for a target type.

Format

```
emcli list_diagchecks
      -target_type="type"
      [-version="<diag_version>" ]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target.

- **version**

  Diagnostic version. Defaults to the latest version.

list_masking_definitions

Gets the list of masking definitions for an associated target and its script status.

Format

```
emcli list_masking_definitions
     [-definition_name=<masking_defn_name_filter>]
     [-adm_name=<application_data_model_filter>]
     [-target_type=<target_type_filter>]
     [-target_name=<target_name_filter>]
     [-string_match]
     [-script | -format=[name:<pretty|script|csv>];
                        [column_separator:"column_sep_string"];
                        [row_separator:"row_sep_string"];
     ]
     [-noheader]

[ ] indicates that the parameter is optional
```

Parameters

- **definition_name**

  Masking definition name filter. This can be either a full value or a pattern match (%).

- **adm_name**

  Application Data Model (ADM) name. This can be either a full value or a pattern match (%).

- **target_type**

  Database target type. This can be either 'oracle_database' or 'rac_database'.

- **target_name**

  Database target name. This can be either a full value or a pattern match (%).

- **string_match**

  Uses an exact string match for a target_name and definition_name match.

- **script**

  This option is equivalent to -format='name: script' .

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

    – row_separator:"row_sep_string" row-separates the verb output by
       <row_sep_string>. Rows are separated by the tab character.

■ **noheader**

Suppresses printing of column headers.

Output Columns

Masking Definition, Database, Status

Examples

Example 1
The following example lists all masking definitions.

```
emcli list_masking_definitions
```

Example 2
The following example lists the masking definition named mask_hr_data.

```
emcli list_masking_definitions -definition_name=mask_hr_data
```

Example 3
The following example lists all masking definitions with names starting with credit_
card.

```
emcli list_masking_definitions  -definition_name=credit_card%
```

Example 4
The following example lists all masking definitions created on a database named
testdb.

```
emcli list_masking_definitions -target_name=testdb
```

Example 5
The following example lists all masking definitions created on databases with names
starting with test.

```
emcli list_masking_definitions -target_name=test%
```

Example 6
The following example lists the masking definition named mask_hr_data created on a
database named testdb.

```
emcli list_masking_definitions -definition_name=mask_hr_data -target_name=testdb
```

Example 7
The following example lists all masking definitions with names starting with credit
and created on databases with names starting with test.

```
emcli list_masking_definitions -definition_name=credit% -target_name=test%
```

Example 8
The following example lists all masking definitions without printing the column
headers.

```
emcli list_masking_definitions -noheader
```

list_oms_config_properties

Lists the OMS configuration properties.

Format

```
emcli list_oms_config_properties
        [-oms_name="omsName"]
        [-details]

[ ] indicates that the parameter is optional
```

Parameters

- **oms_name**

  Name of the OMS from where the properties have to be retrieved.

- **details**

  Displays the details about from where the property value has been derived, and also the global and default values for the property.

Examples

Example 1

The following example lists the entire set of properties.

```
list_oms_config_properties
```

Example 2

The following example lists all the properties set on the management server myhost:1159_Management_Service.

```
list_oms_config_properties -oms_name="myhost:1159_Management_Service"
```

list_oms_logging_properties

Lists the logging configuration properties.

Format

```
emcli list_oms_logging_properties
        [-oms_name="omsName"]
        [-details]

[ ] indicates that the parameter is optional
```

Parameters

- **oms_name**

  Name of the OMS from where the logging properties have to be retrieved.

- **details**

  Displays the details about from where the property value has been derived, and also the global and default values for the logging property.

Examples

Example 1

The following example lists the entire set of logging properties.

```
list_oms_logging_properties
```

Example 2

The following example lists all the logging properties set on the management server myhost:1159_Management_Service.

```
list_oms_logging_properties -oms_name="myhost:1159_Management_Service"
```

list_patch_plans

Lists existing patch plans. You can list all the existing patch plans and can also list the existing patch plans whose names match the specified pattern.

Format

```
emcli list_patch_plans
        [-name="name"]
        [-noheader]
        [-script | -format=
                        [name:<pretty|script|csv>];
                        [column_separator:"column_sep_string"];
                        [row_separator:"row_sep_string"];
        ]
```

[ ] indicates that the parameter is optional

Parameters

- **name**

  Plan name used for searching patch plans. If you do not specify this parameter, the patch plan whose name is the same as the specified name, or contains the specified name string, will be listed. If you do not specify this option, all of the existing patch plans are listed.

- **noheader**

  Suppresses printing of column headers.

- **script**

  This option is equivalent to -format='name: script' .

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  - `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - `row_separator:"row_sep_string"` row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_patch_plans
    emcli list_patch_plans -name="plan name" -noheader
    emcli list_patch_plans -name="plan name" -noheader -script
    emcli list_patch_plans -name="plan name" -noheader -format="name:pretty"
    emcli list_patch_plans -name="plan name" -noheader
    -format="name:pretty";column_separator="separator"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

list_plugins_on_agent

Lists all of the plug-ins deployed on the management Agents.

Format

```
emcli list_plugins_on_agent
        [-agent_names="agent1,agent2,agent3 "
        [-all]
        [-include_discovery]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **agent_names**

  All of the management Agents(host:port) whose deployed plug-ins need to be listed. If you do not provide any Agent names, plug-ins on all Agents are listed. String literals with a wildcard (*) expression are accepted. For example:

  ```
  emcli list_plugins_on_agent -agent_names='adc*,st*93'
  ```

- **all**

  Lists plug-ins on all the management's Agents.

- **include_discovery**

  Includes discovery components of the plug-ins. By default, discovery components of the plug-ins are ignored.

Examples

Example 1
The following example lists plug-ins on the Agent abc.example.com.

```
emcli list_plugins_on_agent -agent_names=abc.us.oracle.com:3872
```

Example 2
The following example lists plug-ins for both of the Agents as well as their discovery components.

```
emcli list_plugins_on_agent -agent_names=
abcd.example.com:3872,efgh.example.com:3872 -include_discovery
```

Example 3
The following example lists plug-ins for all Agents with the name that matches one of the regular expressions adc* or st*93.

```
emcli list_plugins_on_agent -agent_names='adc*,st*93'
```

Example 4
The following example lists plug-ins for all of the management Agents.

```
emcli list_plugins_on_agent -all
```

list_privilege_delegation_settings

Lists privilege delegation setting templates available on the server that apply to targets.

Format

```
emcli list_privilege_delegation_settings
      [-setting_type="SUDO/POWERBROKER]"
      [-noheader]
      [-script | -format=
              [name:<pretty|script|csv>];
              [column_separator:"column_sep_string"];
              [row_separator:"row_sep_string"];
      ]
```

[ ] indicates that the parameter is optional

Parameters

- **setting_type**

  Setting type. All applicable settings are displayed if you do not specify this option.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_privilege_delegation_settings
  -setting_type="SUDO"
```

list_sla

Lists the SLA life-cycle status and version information for a target. If you specify the slaName, the command prints the summary information of the different versions. If you do not specify the slaName, the command prints all the available SLA version series for a target. When you specify the version, this commands prints only summary information for the specified version.

Format

```
emcli list_sla
    -targetName=<target_name>
    -targetType=<target_type>
    [-slaName=<SLA_name>]

[ ] indicates that the parameter is optional
```

Parameters

- **targetName**

  Name of the target.

- **targetType**

  Type of target.

- **slaName**

  Name of the SLA.

Examples

Example 1
The following example prints the SLA information for one SLA.

```
emcli list_sla
      -targetName='my_service' -targetType='generic_service'
      -slaName='gold_sla' -version=2
```

Example 3
The following example prints the SLA information for all SLAs of a target.

```
emcli list_sla
      -targetName='my_service' -targetType='generic_service'
```

list_swlib_entities

Lists the entities in the software library based on the specified filter criteria. The results are printed in the following order:

Display Name, Revision, Description, Status, Type, Subtype, Maturity, Owner, [Folder Path, Folder Id, Entity Rev Id]

Format

```
emcli list_swlib_entities
        [-name="entity_name"]
        [-folder_id="folder_internal_id"]
        [-desc="entity_desc"]
        [-attr="<attr_name>:<attr_value>"]
        [-type]="type_internal_id"]
        [-subtype]="subtype_internal_id"]
        [-maturity]="maturity"]
        [-owner]="owner"]
        [-status]="status"]
        [-show_folder_path]
        [-show_folder_id]
        [-show_entity_rev_id]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the entity. The value specified for this option is considered for a case-insensitive match.

- **folder_id**

  Internal identifier of the parent folder. The value specified for this option is considered for an exact match.

- **desc**

  Description of the entity. The value specified for this option is considered for a case-insensitive match.

- **attr**

  An attribute and its value, separated by a colon (:). For specifying values for multiple attributes, repeat the option. The value specified for this option is considered for an exact match.

  You can only use this parameter with the type parameter.

- **type**

  Internal identifier of the entity type. Use the list_swlib_entity_types verb to identify the type.

- **subtype**

  Internal identifier of the entity sub-type. Use the list_swlib_entity_subtypes verb to identify the sub-type.

- **maturity**

  Maturity of the entity revision. Can be one of:

MAT_Untested

MAT_Beta

MAT_Production

- **owner**

    Owner of the entity revision.

- **status**

    Status of the entity revision. Can be one of:

    STATE_Incomplete

    STATE_Ready

    STATE_Deleted

- **show_folder_path**

    Enables printing of the internal path of each entity's folder.

- **show_folder_id**

    Enables printing of the internal ID of each entity's folder. If specified, the value is printed after the value for show_folder_path.

- **show_entity_rev_id**

    Enables printing of the internal ID of each entity. If specified, the value is printed after the value for show_folder_id.

Examples

The following example lists all folders under the specified parent folder, and also prints the internal identifier for each folder in the list.

```
emcli list_swlib_entities
      -name="myEntity"
      -type="COMP_Component"
      -attr="PRODUCT:Oracle Database"
      -show_folder_id
```

list_swlib_entity_subtypes

Lists the entity subtypes available in the software library for a specified entity type.

Format

```
emcli list_swlib_entity_subtypes
      [-entity_type_id="type_internal_name"]
      [-show_subtype_id]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_type_id**

  Internal identifier of the type.

- **show_subtype_id**

  Enables printing of the internal identifier for the subtype.

Examples

The following example lists all subtypes available in the software library for the type 'COMP_Component.'

```
emcli list_swlib_entity_subtypes
      -entity_type_id="COMP_Component"
      -show_subtype_id
```

list_swlib_entity_types

Lists the entity types available in the software library.

Format

```
emcli list_swlib_entity_types
      [-show_type_id]

[ ] indicates that the parameter is optional
```

Parameters

- **show_type_id**

  Enables printing of the internal identifier for the type.

Examples

The following example lists all of the types available in the software library.

```
emcli list_swlib_entity_types
      -show_type_id
```

list_swlib_folders

Lists folders in the software library.

Format

```
emcli list_swlib_folders
        [-parent_id="parent_folder_id"]
        [-show_folder_path]
        [-show_folder_id]

[ ] indicates that the parameter is optional
```

Parameters

- **parent_id**

  Internal identifier of the parent folder.

- **show_folder_path**

  Enables printing of the internal path for the folder.

- **show_folder_id**

  Enables printing of the internal identifier for the folder.

Examples

The following example lists all folders under the specified parent folder, and prints the internal identifier for each folder in the list.

```
emcli list_swlib_folders
        -parent_id=
"oracle:defaultService:em:provisioning:1:cat:B13B3B7B086458CFE040E80A19AA560C"
        -show_folder_id
```

list_swlib_storage_locations

Lists storage locations configured in the software library.

Format

```
emcli list_swlib_storage_locations
      [-type="OmsShared|OmsAgent|Http|Nfs|ExtAgent"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Type of the storage location. The default is OmsShared.

Examples

The following example lists all locations configured for storage type 'OmsAgent.'

```
emcli +_locations
      -type="OmsAgent"
```

list_target_privilege_delegation_settings

Lists current privilege delegation settings for targets.

Format

```
emcli list_target_privilege_delegation_settings
      -target_names="name1;name2;name3"
      [-input_file="FILE:file_path"]
      [-noheader]
      [-script | -format=
              [name:<pretty|script|csv>];
              [column_separator:"column_sep_string"];
              [row_separator:"row_sep_string"];
      ]
```

[ ] indicates that the parameter is optional

Parameters

- **target_names**

  List of targets. All targets must be of the host type. Either target_names or input_file must be present.

- **input_file**

  Path of the file that has the list of targets. The file should have one target name per line.

- **noheader**

  Display tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is -format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli list_target_privilege_delegation_settings
        -target_names="host.example.com;host2.example.com;

emcli list_target_privilege_delegation_settings
```

```
            -input_file="FILE:/home/nqureshi/targets.txt"

emcli list_target_privilege_delegation_settings
            -target_names="host.example.com;host2.example.com;
```

list_target_property_names

Lists property names for the global properties.

Format

```
emcli list_target_property_names
```

Parameters

None.

list_templates

Lists monitoring templates and their display names.

Format

```
emcli list_templates
        [-target_type="target_type"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Template's target type. If specified, all templates defined for this target type are displayed.

Examples

Example 1

The following example lists all templates.

```
emcli list_templates
```

Exmaple 2

The following example lists all templates defined for the host target type.

```
emcli list_templates -target_type="host"
```

list_trace

Displays the list of OMS traces for the Oracle Management System.

Format

```
emcli list_trace
```

Parameters

None.

list_unconverted_udms

Retrieves the list of UDMs that are not yet in a migration session.

Format

```
emcli list_unconverted_udms
      [-templates_only]

[ ] indicates that the parameter is optional
```

Parameters

- **templates_only**

  Only lists unconverted UDMs in templates.

Examples

Example 1
The following example displays all the UDMs that are not part of a migration session.

```
emcli list_unconverted_udms
```

Exmaple 2
The following example displays all the UDMs that are in a template and not part of a migration session.

```
emcli list_unconverted_udms -templates_only
```

loader_perf

Executes a performance test to determine the network bottleneck between OMS and the Enterprise Manager repository.

Format

```
emcli loader_perf
      [-batchSize="batch size 1" -batchSize="batch size 3"
       -batchSize="batch size 3" ...]
      [-commitSize="commit size 1" -commitSize="commit size 2"
       -commitSize="commit size 3" ...]
      [-dataSize="data size"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **batchSize**

  Batch size for the performance test. Multiple values are allowed. Default values are {14,50,1}.

- **commitSize**

  Commit size for the performance test. Multiple values are allowed. Default values are {500,50,5}.

- **dataSize**

  Number of records to be inserted for a test. This should be greater than and a multiple of 100. If this is not a multiple of 100, a multiple of 100 less than the given value is considered as the dataSize value. The default value is 10000.

Examples

Example 1

Display the time required to load 10,000 records for the default values of batchSize and commitSize.

```
emcli loader_perf
```

Exmaple 2

Display the time required to load 10,0000 records for a batchSize of {15,40} and a commitSize of {10,100}.

```
emcli loader_perf -b=15 -b=40 -c=10 -c=100 -d=100000
```

login

Logs into Enterprise Manager with the given credentials and sets up a session with the OMS.

> **Note:** To avoid an uncommon occurrence in which multiple emcli sessions are created on the OMS, Oracle recommends that you enter the login command before running a script containing EMCLI commands.

> **Tip:** See also

Format

```
emcli login
      -username=<EM_Console_Username>
      [-password=<EM_Console_Password>]
      [-force]

[ ] indicates that the parameter is optional
```

Parameters

- **username**

  Enterprise Manager user name to be used by all subsequent EM CLI commands when contacting the OMS.

- **password**

  Enterprise Manager user password. If you do not specify this , you are prompted for the password interactively.

  > **Note:** Providing a password on the command line is insecure and should be avoided.

- **force**

  Force a login even if there is an existing session.

Examples

The following example shows a login as a different user using newly specified credentials, then a subsequent login using the previous credentials.

```
emcli logut
emcli login -user=new_user -pass=new_user_pass
emcli <verb-name>
emcli logout
emcli login -user=old_user -pass=old_user_pass
```

logout

Terminates the existing session with the OMS. This verb and the login verb are useful when you need to run a particular verb as a different user. After a logout, you need to invoke either the setup verb or login verb before invoking any other emcli verb.

> **Tip:** See also login on page 2-291.

> **Note:** Verbs executed after 'emcli logout' may fail with the message "Error: Session expired. Run emcli login to establish a session." You need to run the login verb to log in to EM CLI after an 'emcli logout'.

Format

```
emcli logout
```

Parameters

None.

Examples

The following example shows a login as a different user using newly specified credentials, then a subsequent login using the previous credentials.

```
emcli logut
emcli login -user=new_user -pass=new_user_pass
emcli <verb-name>
emcli logout
emcli login -user=old_user -pass=old_user_pass
```

merge_credentials

Merges all the references of named credentials provided in the source_credential_list into the destination_credential. The verb expects all the named credentials provided to be equivalent. You can list equivalently named credentials using the command `emcli get_duplicate_credentials`. All the matching duplicate credentials can be merged using the flag merge_all.

Format

```
emcli merge_credentials
        -destination_credential="destination_cred_name[:destination_cred_owner]"
        [-source_credential_list="source_credential_list"]
        [-merge_all]
        [-merge_without_testing]
```

[ ] indicates that the parameter is optional.

Parameters

- **destination_credential**

  Destination credentials to merge the references.

- **source_credential_list**

  Source-named credential list.

- **merge_all**

  Finds all the duplicate credentials and merges.

- **merge_without_testing**

  Merges the credentials without testing the destination credential.

Examples

Example 1

The following example merges the named credentials MyOracleCredential2 and MyOracleCredential3 into MyOracleCredential1. If MyOracleCredential1 is equivalent to MyOracleCredential2 and MyOracleCredential3, all the usages of MyOracleCredential2 and MyOracleCredential3 are replaced with MyOracleCredential1.

```
emcli merge_credentials
        -destination_credential="MyOracleCredential1:ADMIN1"
        -source_credential_list=
          "MyOracleCredential2:ADMIN1;MyOracleCredential3:ADMIN3"
```

Example 2

The following example finds all the named credentials equivalent to MyOracleCredential1 and merges their usages with MyOracleCredential1.

```
emcli merge_credentials
        -destination_credential=MyOracleCredential1
        -merge_all
```

metric_control

For the specified target type, lists the metrics whose alerts are stateless and therefore can be manually cleared. Both the metric name and metric internal name are provided in the output of this command. To clear the stateless alerts associated with the specified metric, use the clear_stateless_alerts verb.

> **Tip:** See also clear_stateless_alerts on page 2-41.

Format

```
emcli metric_control
        -command=command
        -target_type=type
        -metric_name=name
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **command**

  Can be one of the following:

  - disable_metric — Disables loading of the specified metric .

  - enable_metric — Reenables loading of the specified metric.

  - list_disabled_metrics — Lists the metrics currently disabled for loading.

  - flush_metadata_cache — Flushes the metric API metadata cache target_type.

- **target_type**

  Internal target type identifier (host, oracle_database, oc4j, oracle_emrep, oracle).

- **metric_name**

  Internal name of the metric (for example, load for the host target type).

Example

The following example disables the loading of the Load metric on the host target type.

```
emcli metric_control -command=disable_metric -target_type=host -metric_name=Load
```

migrate_to_lifecycle_status

Migrates to the lifecycle state from the deployment type.

Format

```
emcli migrate_to_lifecycle_status
      -deployment_values="value1;value2;value3
      -lifecycle_stage_values="Stage;Stage;Production
```

Parameters

- **deployment_values**

  Deployment type values.

- **lifecycle_stage_values**

  Lifecycle stage values

modify_aggregate_service

Modifies an aggregate service instance.

Format

```
emcli modify_aggregate_service
      -name="name"
      -type="type"
      [-add_sub_services="name1:type1;name2:type2;..."]
      [-del_sub_services="name1:type1;name2:type2;..."]
      [-avail_eval_func="function_to_evaluate_availability."]
      [-timezone_region="timezone_region"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Aggregate service name.

- **type**

  Aggregate service type.

- **add_sub_services**

  Sub-services to be added.

- **del_sub_services**

  Sub-services to be deleted.

- **avail_eval_func**

  PL/SQL function to evaluate the availability of the aggregate service. Use [or | and] for the predefined evaluation helper function.

- **timezone_region**

  Time zone region of the service.

Examples

```
emcli modify_aggregate_service -name="My_Name"
      -type="aggregate_service"
      -add_sub_services="sub1:type1;sub2:type2"
      -del_sub_services="sub3:type3"
      -avail_eval_func="my_pkg.my_eval_func"
      -timezone_region="CST"
```

modify_collection_schedule

Modifies the collection schedule of a collection setup for metrics and policies for the specified set of targets. Combining all the metrics, running a script, and collecting the data is referred to as a collection. The collection has various attributes associated with it, such as the collection schedule, upload frequency, and so forth.

Format

```
emcli modify_collection_schedule
      -targetType=ttype
      -targetNames=tname1;tname2;tname3...
      -collectionName=collname
      [-collectionStatus=Enabled or Disabled]
      [-freqType={Minute}{Hour}{Day}{Week}{Weekly}{Month}
      [-freqValue={any integer value for Minute/Hour/Day/Week}{One or more from
       Mon...Sun for Weekly}{One or more from 1;2..31 or Last for Month}
      [-preview=Y or N]

[ ] indicates that the parameter is optional
{ } indicates that you can select one of the s in the series shown

Note: All of the parameters and choices are case-insensitive
```

Parameters

- **targetType**

  You must specify a single target type value, and it should be the same as specified in the repository.

  > **Note:** Only individual target types are currently supported.

- **targetNames**

  The target name should be the same as exists in the repository. All of the targets should be the same target type you specified in the targetType parameter. Use a semicolon (;) to separate the names. Changes to the collection schedule will be executed for only valid target name and target type combinations. For example:

  host1;host2;host3

- **collectionName**

  The collection name should be exactly the same as exists in the repository or the corresponding collections .xml file present on the Management Agent.

  Access files from the following locations to determine the collection to be modified. Select the desired collection and provide it as input to the EM CLI utility.

  – $AGENT_HOME/sysman/admin/metadata/<targetType>.xml

    This file is shipped as a part of the setup and contains information regarding the metrics for this target type.

  – $AGENT_HOME/sysman/admin/default_collection/
    <targetType>.xml

This file is shipped as a part of the setup and contains the collections shipped by default.

– $AGENT_HOME/sysman/emd/collection/ <targetType_targetName>.xml

Whenever changes have occurred for any particular target, this file is automatically generated. Collections for user-defined metrics are available in this file.

■ **collectionStatus**

Enables or disables the collection. The default is Enabled. If Disabled, freqType and freqValue are ignored.

■ **freqType**

You can specify one of the following values:

Minute (default)
Hour
Day
Week
Weekly
Month

For Week, you must specify an integer value as the frequency value. For instance, if you specify freqType='WEEK' and freqValue='2', the collection occurs every two weeks.

For Weekly, the possible values are Mon, Tue, Wed, Thu, Fri, Sat, Sun. For instance, if you specify freqType='Weekly' and freqValue='Tue;Thu;Sun', the collection occurs every Tuesday, Thursday, and Sunday of a week.

The schedule is modified based on your selection. You do not need to specify a value (and the value will be ignored) if the collectionStatus parameter is set to Disabled.

If you use this parameter, you must also use the freqValue parameter.

■ **freqValue**

You can specify one of the following values:

– You must specify an integer value if the freqType is any one of Minute, Hour, Day, or Week. The default value is 5.

– For Weekly, specify one or more choices from Mon, Tue, Wed, Thu, Fri, Sat, and Sun. If the collection occurs on any particular day(s) of the week, you must specify the corresponding value(s) against the Weekly option.

– For Monthly, specify one or more choices from 1...31 or Last. If the collection occurs on any particular date(s) in a month, you must specify the corresponding value(s) against the Monthly option.

You do not need to specify a value (and the value will be ignored) if the collectionStatus parameter is set to Disabled.

If you use this parameter, you must also use the freqType parameter.

■ **preview**

Provides a preview of the changes that would occur if this verb is executed. The default value for this option is Y (Yes), whether you specify the option or not. If

you specify N, the changes to the collection schedule are executed for both the repository and Management Agent.

Examples

Example 1

The following example changes the collection schedule to collect once every 5 minutes for hosts host1, host2, and host3. DiskActivity is a collection item associated with a host target type. The preview flag is set to Y, so the changes are not executed, but you can see the metrics affected if the changes were implemented.

```
emcli modify_collection_schedule -targetType="host"
     -targetNames="host1;host2;host3" -collectionName="DiskActivity"
     -freqType="Minute" -freqValue="5" -preview="Y"
```

Example 2

The following example changes the collection schedule to collect once every 15 hours for host host1. Inventory is a collection item associated with a host target type. The preview flag is set to N, so the changes are executed for the associated metrics for both the repository and Management Agent.

```
emcli modify_collection_schedule -targetType="host"
     -targetNames="host1" -collectionName="Inventory"
     -freqType="Hour" -freqValue="15" -preview="N"
```

Example 3

The following example changes the collection schedule to collect on Monday and Thursday every week for hosts host1 and host2. Inventory is a collection item associated with a host target type. The preview option is not specified, but since the value is Y whether you specify the option or not, the changes are not executed, but you can see the metrics affected if the changes were implemented.

```
emcli modify_collection_schedule -targetType="host"
     -targetNames="host1;host2" -collectionName="Inventory"
     -freqType="Weekly" -freqValue="Mon;Thu"
```

Example 4

The following example changes the collection schedule to collect on the 1st, 5th, 23rd, and last day of every month for hosts host1 and host2. Inventory is a collection item associated with a host target type.

```
emcli modify_collection_schedule -targetType="host"
     -targetNames="host1;host2" -collectionName="Inventory"
     -freqType="Month" -freqValue="1;5;23;Last"
```

Example 5

The following example disables the collection schedule for hosts host1 and host2. Inventory is a collection item associated with a host target type.

```
emcli modify_collection_schedule -targetType="host"
     -targetNames="host1;host2" -collectionName="Inventory"
     -collectionStatus="Disabled"
```

modify_group

Adds or removes targets from an existing group.

An error is not generated when attempting to delete a non-existent target in the group or when attempting to add a target that already exists in the group.

Format

```
emcli modify_group
      -name="name"
      [-type=<group>]
      [-add_targets="name1:type1;name2:type2;..."]...
      [-delete_targets="name1:type1;name2:type2;..."]...
      [-privilege_propagation=true|false]
      [-drop_existing_grants=yes|no]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the group to modify.

- **type**

  Group type: group. Defaults to group.

- **add_targets**

  Targets to add, each specified as target_name:target_type. You can specify this option more than once.

- **delete_targets**

  Targets to delete, each specified as target_name:target_type. You can specify this option more than once.

- **privilege_propagation**

  Enables or disables the privilege propagation flag for the group. Converts the normal group to a privilege propagating group and vice versa.

- **drop_existing_grants**

  Drops the existing grants on a group during privilege propagation conversion. This parameter is only applicable with the privilege_propagation parameter. The default value is yes.

Examples

Example 1

The following example modifies group db2_group by adding database database:oracle_database and deleting database database2:oracle_database from the group.

```
emcli modify_group -name=db2_group
      -add_targets=database:oracle_database
      -delete_targets=database2:oracle_database
```

Example 2

The following example modifies group `my_hosts` by adding host
`yourhost.example.com:host` to the group.

```
 emcli modify_group -name=my_hosts
      -add_targets=yourhost.example.com:host
```

Example 3

The following example modifies group `my_group` by adding targets `group_a:group`
and `database:oracle_database` and deleting the nonexistent target
`nogroup:group` from the group.

```
emcli modify_group -name=my_group
      -add_targets=group_a:group
      -add_targets=database:oracle_database
      -delete_targets=nogroup:group
```

Example 4

The following example converts group my_group to privilege propagating, ignores if
already converted, and drops all of its existing grants.

```
emcli modify_group -name=my_group
          -privilege_propagation=true
```

Example 5

The following example converts group my_group to non-privilege propagating,
ignores if already converted, and retains all of its existing grants on my_group.

```
emcli modify_group -name=my_group
          -privilege_propagation=false
          -drop_existing_grants=no
```

modify_lifecycle_stage_name

Changes the life-cycle stage name. Only super users can run this command.

Format

```
emcli modify_lifecycle_stage_name
      -name="current_name"
      -new_name="new_name"
```

Parameters

- **name**

  Current life-cycle stage name. The available list in the order of decreasing priority is:

  – MissionCritical

  – Production

  – Stage

  – Test

  – Development

- **new_name**

  New life-cycle stage name. The new name is not translated into your locale and will be displayed as is. The new name should only contain alpha characters.

  When you change the existing name to a new name, all existing targets are upated with the new property value. For instance, if name=MissionCritical and new_name=Production, all existing targets are upated with Production.

Examples

```
emcli modify_lifecycle_stage_name
      -name="Test"
      -new_name="Test_staging"
```

modify_named_credential

Updates an existing named credential. You can provide input parameters using command line arguments or an input properties file. It also supports the input_file tag for passwords and parameter values.

Format

```
emcli modify_named_credential
      -cred_name=<name>
      -new_cred_name<name>
      -cred_type=<credential_type>
      -cred_scope=<credential_scope>
      -cred_desc=<credential_description>
      -target_name=<target_name>
      -target_type=<target_type>
      -test
      -test_target_name=<test_target_name>
      -test_target_type=<test_target_type>
      -input_file=<tag|value>
      -properties_file=<filename>
      -attributes=<p1:v1;p2:v2;...>
      -remove_old_attributes
```

Parameters

- **cred_name**

  Credential name, such as MyBackUpCreds. This is required if you do not use the properties_file option.

- **new_cred_name**

  New credential name.

- **cred_type**

  Credential type.

- **cred_scope**

  Possible values are global instance. The default is global.

- **cred_desc**

  Credential description.

- **target_name**

  This is required when cred_scope is instance.

- **target_type**

  This is required when cred_scope is instance.

- **test**

  Use this parameter to test the credential before saving.

- **test_target_name**

  Use this parameter to supply the target name to test a global credential. This is mandatory when the scope is global and the test option is used.

- **test_target_type**

Use this parameter to supply the target type to test a global credential. This is mandatory when the scope is global and the test option is used.

- **input_file**

  Use this option to supply sensitive property values from the file.

- **properties_file**

  Use this option to pass all parameters from the file. Values given on the command line take precedence.

- **attributes**

  Specify credential columns as follows:

  ```
  colname:colvalue;colname:colvalue
  ```

  You can change the separator value using -separator=attributes=<newvalue>, and you can change the sub-separator value using -subseparator=attributes=<newvalue>.

- **remove_old_attributes**

  Unsets all existing credential column values.

Examples

Example 1
The following example updates credentials to foo and bar:

```
emcli modify_named_credential
      -cred_name=NC1
      -attributes="HostUserName:foo;HostPassword:bar"
```

Example 2
The following example updates the password to bar:

```
emcli modify_named_credential
      -cred_name=NC1
      -attributes="HostPassword:bar"
```

Example 3
The following example reads the password from the mypasswordfile.txt file.

```
emcli modify_named_credential
      -cred_name=NC1
      -attributes="HostUserName:foo;HostPassword:tag"
      -input_file="tag:mypasswordfile.txt"
```

Example 4
The following example prompts for the password from standard input:

```
emcli modify_named_credential
      -cred_name=NC1
      -attributes="HostUserName:foo;HostPassword:"
```

Example 5
The following example specifies prop1.txt as a multi-line Java properties file, in which each line contains a parameter=value format. You can provide the password in the same file or not specify it. If not specified, you are prompted for it.

```
emcli modify_named_credential
        -properties_file=prop1.txt
```

modify_red_group

Adds or removes targets from an existing redundancy group. An error is not generated when attempting to delete a non-existent target in the redundancy group.

Format

```
emcli modify_red_group
      -name="name"
      -type=<generic_redundancy_group>
      [-add_targets="name1:type1;name2:type2;..."]...
      [-delete_targets="name1:type1;name2:type2;..."]...
      [-owner=<redundancy_group_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the group to modify.

- **type**

  Redundancy Group type: generic_redundancy_group. Defaults to `generic_redundancy_group`.

- **add_targets**

  Targets to add, each specified as `target_name:target_type`. You can specify this option more than once.

- **delete_targets**

  Targets to delete, each specified as `target_name:target_type`. You can specify this option more than once.

- **owner**

  Owner of the redundancy group.

Examples

The following example modifies redundancy group servers by adding `Server1:generic_apache` and deleting `Server5:generic_apache` from the redundancy group.

```
emcli modify_red_group -name=Servers
      -add_targets=HTTP_Server1:generic_apache
      -delete_targets=Server5:generic_apache
```

modify_redundancy_group

Modifies a redundancy group.

Format

```
emcli modify_redundancy_group
      -redundancyGroupName="redGrpName"
      [-owner="new_owner"]
      [-memberTargetType="tType"]
      [-add_targets="tName1;tName2"]
      [-delete_targets="tName3;tName4"]
      [-group_status_criterion="NUMBER" or "PERCENTAGE"]
      [-group_status_tracked="UP" or "DOWN"]
      [-group_status_value=<status_value>]
      [-privilege_propagation=true|false]
      [-drop_existing_grants=yes|no]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **redundancyGroupName**

  Name of the redundancy group.

- **owner**

  Valid owner to be specified.

- **memberTargetType**

  Target type of the constituent member targets. You need to specify this parameter if you specify either add_targets or delete_targets.

- **add_targets**

  Member targets to be added to this redundancy group.

- **delete_targets**

  Member targets to be deleted from this redundancy group.

- **group_status_criterion**

  This option and the next two calculate the status of the Redundancy Group. Consequently, you need to specify all three options together. If this is not to be a capacity group, you need to specify the following combination:

  ```
  -group_status_criterion='NUMBER' -group_status_tracked='UP'
  -group_status_value='1']
  ```

- **group_status_tracked**

  See the option above.

- **group_status_value**

  See the group_status_criterion .

  You can specify any value between 1 and 100 if -group_status_criterion="PERCENTAGE", or any value between 1 and the number of targets present if -group_status_criterion="NUMBER".

- **privilege_propagation**

   Enables or disables the privilege propagation flag for the group. Converts the normal group to a privilege-propagating group and vice versa.

- **drop_existing_grants**

   Drops the existing grants on a group during privilege propagation conversion. This parameter is only applicable with the privilege_propagation parameter. The default value is yes.

Examples

The following example changes the configuration of the 'redGrp1' redundancy group to add listener, listener2, and listener3 to its existing members, and delete listener4 and listener5 from its existing members.

```
emcli modify_redundancy_group -redundancyGroupName='redGrp1'
        -memberTargetType='oracle_listener'
        -add_targets='listener;listener2;listener3'
        -delete_targets='listener4;listener5'
        -group_status_criterion='NUMBER'
        -group_status_tracked='UP'
        -group_status_value='2'
```

modify_resolution_state

Modifies an existing resolution state that describes the state of incidents or problems. Only super administrators can execute this command. You need to specify the updated label as well as the updated position. The position can be between 2 and 98, and cannot be in use by another resolution state.

You can also optionally indicate that the state should apply to both incidents and problems. A success message is reported if the command is successful. An error message is reported if the change fails.

Format

```
emcli modify_resolution_state
      -label="old_label_of_state"
      -new_label="new_label_for_display"
      -position="new_display_position"
      [-applies_to=BOTH]

[ ] indicates that the parameter is optional
```

Parameters

- **label**

  Old label of the state to be modified.

- **new_label**

  End-user visible label of the state. The label cannot exceed 32 characters.

- **position**

  Position of this state within the overall list of states. This is used when displaying the list of states in the user interface. The position can be between 2 and 98.

  It is recommended that you set the position with sufficient gaps to facilitate moving states around. For example, if you set the positions to 5, 10, and 15 instead of 2, 3, and 4, it is easier to move a state from position 15 to 9, for instance, in contrast to the latter scheme, in which you would have to move all states to provide space for the reordering.

- **applies_to**

  Indicates that the state is applicable for incidents and problems. The only supported value is "BOTH."

Examples

Example 1

The following example updates the resolution state with the old label "Waiting for TT" with the new label "Waiting for Ticket," and if necessary, changes the position to 25.

```
emcli modify_resolution_state  -label="Waiting for TT" -new_label="Waiting for
Ticket" -position=25
```

Example 2

The following example updates the resolution state with the old label "SR Waiting" with the new label "Waiting for SR," and if necessary, changes the position to 35. It also makes the state applicable to incidents and problems.

```
emcli modify_resolution_state  -label="SR Waiting" -new_label="Waiting for SR"
-position=35 -applies_to=BOTH
```

modify_role

Modifies an existing Enterprise Manager administrator role.

> **Note:** Omit an argument to leave its value unchanged.
>
> To update a role and add targets to the role, use the grant_privs verb.

Format

```
emcli modify_role
      -name="role_name"
      [-description="description"]
      [-roles="role1;role2;..."]
      [-privilege="name[;secure-resource-details]]"
      [-separator=privilege="sep_string"]
      [-subseparator=privilege="subsep_string"]
      [-users="user1;user2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Role name.

- **description**

  Replaces the description of the role.

- **roles**

  Replaces the list of roles assigned to this existing role. Currently, the only built-in role is PUBLIC.

- **privilege**

  Replaces privileges granted to this role. You can specify this option more than once. Specify <secure_resource_details> as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
  name2=resource_column_value2]..]"
  ```

- **separator**

  Specify a string delimiter to use between name-value pairs for the value of the -privilege option. The default separator delimiter is a semi-colon (;).

- **subseparator**

  Specify a string delimiter to use between name and value in each name-value pair for the value of the -privilege option. The default subseparator delimiter is a colon (:).

- **users**

  Replaces the list of users to whom this role is assigned.

Examples

Example 1

The following example modifies a role named `existing_role` with the one-sentence description "This role was changed." The role combines three existing roles: `role1`, `role2`, and `role3`. The role also has two added privileges: to view the job with ID `923470234ABCDFE23018494753091111` and to view the target `host1.example.com:host`. The role is granted to `johndoe` and `janedoe`.

```
emcli modify_role
     -name="existing_role"
     -desc="This role was changed"
     -roles="role1;role2;role3"
     -privilege="view_job;923470234ABCDFE23018494753091111"
     -privilege="view_target;host1.example.com:host"
     -users="johndoe;janedoe"
```

Example 2

The following example modifies a role named `existing_role` by assigning `role4`, `role5`, and `role6` to it. The description, privileges, and users associated with this role remain unchanged.

```
emcli modify_role
     -name="existing_role"
     -roles="role4;role5;role6"
```

modify_system

Adds or removes targets from an existing system. An error is not generated when attempting to delete a non-existent target in the system or when attempting to add a target that already exists in the system.

If you specify both the -add_members and -delete_members options in the same command, the members specified by -delete_members are deleted first, then the members specified by -add_members are added.

Format

```
emcli modify_system
        -name="name"
        [-type=<generic_system>]
        [-add_members="name1:type1:key_member|non_key_member;name2:type2;..."]...
            [-separator=add_members="sep_value"]
            [-subseparator=add_members="subsep_value"]
        [-delete_members="name1:type1;name2:type2;..."]...
            [-separator=delete_members="sep_value"]
            [-subseparator=delete_members="subsep_value"]
        [-owner="new_owner"]
        [-privilege_propagation=true|false]
        [-drop_existing_grants=yes|no]
        [-availability_type="ALL/ANY"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name of the system to modify.

- **type**

  System type: generic_system. Defaults to generic_system.

- **add_members**

  Targets to add, each specified as target_name:target_type. You can specify this more than once. key_member specifies that this target is a part of the systems availability calculation. non_key_member specifies that this target is not a part of the systems availability calculation.

- **delete_members**

  Member targets to be removed from the system, each specified as target_name:target_type. You can specify this option more than once.

- **owner**

  New owner of the system.

- **privilege_propagation**

  Enables or disables the privilege propagation flag for the group. Converts the normal group to a privilege propagating group and vice versa.

- **drop_existing_grants**

Drops existing grants on a group when conversion occurs in privilege propagation nature. This parameter is only applicable with the privilege_propagation parameter. The default value is yes.

- **availability_type**

  Availability calculation method of the system. Defining this is required if key_member is defined. ALL denotes that all key members must be up in order to establish the system as UP. ANY denotes that at least one of the key members must be up in order to establish the system as UP.

Examples

Example 1

The following example modifies system db2_system by adding database database:oracle_database and deleting database database2:oracle_database from the system. The new owner of the system is user2.

```
emcli modify_system -name=db2_system
      -add_members=database:oracle_database
      -delete_members=database2:oracle_database
      -owner=user2
```

Example 2

The following example modifies system my_hosts by adding host yourhost.example.com:host to the system.

```
emcli modify_system -name=my_hosts
      -add_members=yourhost.example.com:host
```

Example 3

The following example modifies system my_system by adding targets system_a:generic_system and database:oracle_database, and deleting the nonexistent target nosystem:generic_system from the system.

```
emcli modify_system -name=my_system
      -add_members=system_a:generic_system
      -add_members=database:oracle_database
      -delete_members=nosystem:generic_system
```

Example 4

The following example modifies system db2_system by adding database database1 as a key member, adding databases database2 and database3 as non-key members, and deleting database4 and database5. The availability computation is impacted, since database1 is now part of the availability computation for the db2_system. If database4 and database5 were key members, they are no longer part of the availability computation for the db2_system.

Specifying separator and subseparator is optional. Separator defaults to ; and subseparator defaults to : .

```
emcli modify_system -name=db2_system -type=generic_system
  [add_members=database1:oracle_database:key_member,database2:oracle_database]
    [separator=add_members=","]
    [subseparator=add_members=":"]
  [add_members=database3:oracle_database:non_key_member]
  [delete_members=database4:oracle_database,database5:oracle_database]
    [separator=delete_members=","]
    [subsrparator=delete_members=":]"
```

modify_target

Modifies a target instance definition.

Format

```
emcli modify_target
    -name="name"
    -type="type"
    [-properties="pname1:pval1;pname2:pval2;..."]...
    [-separator=properties="sep_string"]
    [-subseparator=properties="subsep_string"]
    [-credentials="userpropname:username;pwdpropname:password;..."]
    [-input_file="parameter_tag:file_path"]
    [-display_name="display name"]
    [-on_agent]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Target name.

- **type**

  Target type.

- **properties**

  Name-value pair list of properties for the target instance. The "name"(s) are identified in the target-type metadata definition. They must appear exactly as they are defined in that file. Metadata files are located in `$AGENT_ORACLE_HOME/sysman/admin/metadata`.

  > **Note:** This verb does not support setting global target properties. It is recommended that you use set_target_property_values to set target properties.

- **separator=properties**

  Specifies a string delimiter to use between name-value pairs for the value of the `-properties` option. The default separator delimiter is ";".

- **subseparator=properties**

  Specifies a string delimiter to use between name and value in each name-value pair for the value of the `-properties` option. The default subseparator delimiter is ":".

- **credentials**

  Monitoring credentials (name-value pairs) for the target instance. The "name"(s) are identified in the target-type metadata definition as credential properties. They must appear exactly as they are defined in that file. Metadata files are located in `$AGENT_ORACLE_HOME/sysman/admin/metadata`.

- **input_file**

Used in conjunction with the `-credentials` option, this option enables you to store specific target monitoring credential values, such as passwords, in a separate file. The `-input_file` option specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific monitoring credentials of the `-credentials` option. The tag must not contain colons (:) or semi-colons (;).

- **display_name**

  Sets the target display name.

- **on_agent**

  Propagates changes to the Management Agent collecting this target's metrics.

Examples

Example 1

The following example modifies the display name to `New Name DB` for the database with the internal name `database`.

```
emcli modify_target
     -name="database"
     -type="oracle_database"
     -display_name="New Name DB"
```

Example 2

The following example modifies the credentials for the `oracle_database` target with the name `database`. This example illustrates the use of the `input_file` to camouflage the credentials. The password is actually in a file named `at_pwd_file`. The `input_file` argument replaces `PWD_FILE` with the contents of the `at_pwd_file` in the `credentials` argument. The `on_agent` flag ensures that the changes are propagated to the Management Agent collecting for this target.

```
emcli modify_target
     -name="database"
     -type="oracle_database"
     -credentials="UserName:newuser;password:PWD_FILE;Role:SYSDBA"
     -input_file="PWD_FILE:at_pwd_file"
     -on_agent
```

Example 3

The following example modifies the display name and properties for the `oracle_database` target with the name `database`. The `on_agent` flag ensures that the changes are propagated to the Management Agent collecting for this target.

```
emcli modify_target
     -name="database"
     -type="oracle_database"
     -display_name="New Name DB"
     -properties="SID=newsid|Port=15091|OracleHome=/oracle"
     -properties="MachineName=smpamp-sun1.example.com"
     -separator=properties="|"
     -subseparator=properties="="
     -on_agent
```

Example 4

The following example modifies an `oracle_database` target type with the name `payroll_db`. In this example, the display name for this database (target name that is displayed in the Enterprise Manager UI) is being changed to `payroll`. The port

number is being changed to `15067`, and the Oracle Home is being changed to `/oradb`. The administrator (`dbsnmp`), whose previous default role was `normal`, is being changed to `sysdba`. This example also illustrates the use of the `input_file` to camouflage the credentials. The password is actually in a file named `at_pwd_file`. The `-input_file` argument replaces `PWD_FILE` with the contents of `at_pwd_file` in the `-credentials` option.

```
emcli modify_target
      -name="payroll_db"
      -type="oracle_database"
      -credentials="UserName:Fred;password:PWD_FILE;Role:sysdba"
      -properties="Port:15067;OracleHome:/oradb"
      -input_file="PWD_FILE:at_pwd_file"
      -display_name=payroll
      -on_agent
```

modify_user

Modifies an existing Enterprise Manager administrator.

Format

```
emcli modify_user
        -name="name"
        [-type="type_of_user"]
        [-password="password"]
        [-roles="role1;role2;..."]
        [-email="email1;email2;..."]
        [-privilege="name[;secure_resource_details]]"
        [-separator=privilege="sep_string"]
        [-subseparator=privilege="subsep_string"]
        [-profile="profile_name"]
        [-desc="user_description"]
        [-expired="true|false"]
        [-prevent_change_password="true|false"]
        [-department="department_name"]
        [-cost_center="cost_center"]
        [-line_of_business="line_of_business"]
        [-contact="contact"]
        [-location="location"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Administrator name.

- **type**

  Converts the type of user from a repository user to an external user and vice versa. Possible values for this parameter are EM_USER or EXTERNAL_USER.

- **password**

  Replaces the administrator password with the specified password.

- **roles**

  Replaces current roles with the specified list of Enterprise Manager roles to grant to this administrator. Currently, the built-in roles include PUBLIC.

- **email**

  Replaces current email addresses for this administrator with the specified list. To delete all email addresses for this administrator, specify an empty string.

- **privilege**

  Privilege to grant to this administrator. You can specify this option more than once. The original administrator privileges will be revoked. Specify <secure_resource_details> as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
  name2=resource_column_value2]..]"
  ```

  To retrieve the list of system privileges that do not require resource information, execute the get_supported_privileges command.

- **profile**

  Database profile name. When not passed, this value is not altered.

- **desc**

  User description

- **expired**

  True immediately expires the password. The default is false.

- **prevent_change_password**

  True prevents a user from updating his/her password. The default is false.

- **department**

  Department name of the administrator.

- **cost_center**

  Cost center of the administrator in the organization.

- **line_of_business**

  Line of business of the administrator.

- **contact**

  Contact information for the administrator.

- **location**

  Location of the administrator.

Examples

Example 1

The following example modifies the `new_admin` administrator. The user will have two privileges: to view the job with ID `923470234ABCDFE230184947530911111` and to view the target `host1.example.com:host`. The user will also be granted role `PUBLIC`. The user email addresses will be set to `first.last@example.com` and `joe.shmoe@shmoeshop.com`.

```
emcli modify_user
      -name="new_admin"
      -password="oracle"
      -email="first.last@example.com;joe.shmoe@shmoeshop.com"
      -roles="public"
      -privilege="view_job;923470234ABCDFE230184947530911111"
      -privilege="view_target;host1.example.com:host"
```

Example 2

The following example deletes all the email addresses and privileges for administrator `new_admin`. Note that `-privilege=""` and `-privilege` are equivalent if specified at the command line in a UNIX shell.

```
emcli modify_user
      -name="new_admin"
      -email=""
      -privilege=""
```

provision

Provisions a hardware server using configuration properties from the input file. The configuration properties required for a component can be viewed from the Cloud Control console. After you make a provisioning request, you can view the status of the request from the Enterprise Manager Cloud Control console by using the assignment name (specified by you or the automatically generated name returned to you).

Format

```
emcli provision
      -image="path_to_image"
      -network="network_profile_path"
      -bootserver="boot_server_name"
      -stageserver="stage_server_name"
      -stgcredentials="username"
      -schedule="type:immediate/onetime;timezone:zone;
       startdt:startdate;starttm:time"
      -resettimeout="time"
      -target="hardware_server_label"
      -input_file="config_properties:file_path"
      -assignment="assignment_name"
      [-desc="assignment_description"]

[ ] indicates that the parameter is optional
```

Parameters

- **image**

  Path to the image (includes the image name). This is the image used for provisioning.

- **network**

  Path name of the network profile.

- **bootserver**

  Name of the boot server.

  Format: hostName:Directory Path

- **stageserver**

  Name of the stage server. hostName:Directory Path.

- **Stgcredentials**

  User name of the stage server.

- **schedule**

  Time when provisioning should be scheduled. This is a string argument that contains multiple name-value pairs separated by `;'. This is used to schedule the provisioning operation. "type" can be `immediate' or `onetime'. If "type" is not immediate, the other values are expected in the Time Zone: string, which is a timezone ID of the format:

  zone Sign TwoDigitHours:Minutes

  zone: Time zone ID (GMT, PDT, and so forth)

  Sign: one of "+ -"

TwoDigitHours: Digit Digit

Minutes: Digit Digit

Digit: One of 0 1 2 3 4 5 6 7 8 9

Startdt: Date string of the format: MM/DD/YY

Starttm: Time string of the format: HH:MM

- **resettimeout**

  Reset timeout for the hardware server in minutes.

- **target**

  Target hardware server is specified using the hardware label type.

- **input_file**

  File containing configuration properties.

- **assignment**

  Name of the assignment.

- **desc**

  Assignment description. The description is automatically generated if not specified.

Examples

The following example submits a job to provision `myimage` on a target with the label of `mylabel`. The job runs immediately with a reset timeout of 100 minutes. Image properties are picked from `properties.txt` that overrides the default image. `properties.stageserver` is used as the staging server, and `/private/share` as the staging storage with `joe` as the user name.

```
emcli provision
      -image="Images/myimage"
      -network="Networks/networkprofile"
      -bootserver="booservername.example.com"
      -stageserver="stageserver.example.com:/private/share"
      -stgcredentials="joe"
      -schedule="type:immediate"
      -resettimeout="100"
      -target="mylabel"
      -input_file="config_properties:properties.txt"
      -assignment="provision mylabel"
```

publish_change_request_ccc

Sends change request data to the Change Management Connector, and data processed into the Configuration Change Console. Some of the properties (such as connector_guid, target, and facet) are to be specified as part of customization. All of the data should be able to be mapped to the data required in publishChangeRequest.xsd after XSLT.

Format

```
emcli publish_change_request_ccc
    -connector_guid="ConnectorGUID"
    -change_id="change_ID"
    -last_modified_date="last_modified_date"
    -properties_list="list_of_Change_Management_specific_properties"
    -date_format="Date_format_in_Change_Management_System"
```

Parameters

- **connector_guid**

- **change_id**

- **last_modified_date**

- **properties_list**

  Specify all relevant properties of the Change Management System required for CCC to process a change request.

  The properties are name,value pairs to be specified as prop_name1=value1;prop_name2=value2 with no quotes for values.

  prop_name and values cannot contain the equals sign (=) or semi-colons (;).

- **date_format**

  Specify a date format in the Change Management System:

  MM/dd/yyyy hh:mm:ss if the date field in change management is "09/14/2011 5:38:24 AM"

publish_event

Publishes a user-reported event to Enterprise Manager. This event is published as an event of the "User-reported event" class. Only users with Manage Target privilege can publish these events for a target. An error message is reported if the publish fails.

After an event is published with a severity other than CLEAR (see below), end-users with appropriate privileges can manually clear the event from the user interface, or you can publish a new event using a severity level of CLEAR and the same details to report clearing of the underlying situation.

Format

```
emcli publish_event
        -target_name="target_name"
        -target_type="target_type_internal_name"
        -message="message_for_event"
        -severity="severity_level"
        -name="event_name"
        [-key="sub_component_name"
        [-context="name1=value1;name2=value2;.."]
        [-separator=context="alt._pair_separator"]
        [-subseparator=context="alt._name-value_separator"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Target name.

- **target_type**

  Target type name.

- **message**

  Message to associate for the event. The message cannot exceed 4000 characters.

- **severity**

  Numeric severity level to associate for the event. The supported values for severity level are as follows:

  "CLEAR"
  "MINOR_WARNING"
  "WARNING"
  "CRITICAL"
  "FATAL"

- **name**

  Name of the event to publish. The event name cannot exceed 128 characters.

  This is indicative of the nature of the event. Examples include "Disk Used Percentage," "Process Down," "Number of Queues," and so on. The name must be repeated and identical when reporting different severities for the same sequence of events. This should not have any identifying information about a specific event; for example, "Process xyz is down." To identify any specific components within a target that the event is about, see the key below.

- **key**

  Name of the sub-component within a target this event is related to. Examples include a disk name on a host, name of a tablespace, and so forth. The key cannot exceed 256 characters.

- **context**

  Additional context that can be published for a given event. This is a series of strings of format name:value separated by a semi-colon. For example, it might be useful to report the percentage size of a disk when reporting space issues on the disk. You can override the default separator ":" by using the sub-separator , and the pair separator ";" by using the separator .

  The context names cannot exceed 256 characters, and the values cannot exceed 4000 characters.

- **separator**

  Set to override the default ";" separator. You typically use this option when the name or the value contains ";". Using "=" is not supported for this option.

- **subseparator**

  Set to override the default ":" separator between the name-value pairs. You typically use this option when the name or value contains ":". Using "=" is not supported for this .

Examples

Example 1

The following example publishes a warning event for "my acme target" indicating that a HDD restore failed, and the failure related to a component called the "Finance DB machine" on this target.

```
emcli publish_event  -target_name="my acme target" -target_type="oracle_acme"
-name="HDD restore failed" -key="Finance DB machine" -message="HDD restoration
failed due to corrupt disk" -severity=WARNING
```

Example 2

The following example publishes a minor warning event for "my acme target" indicating that a HDD restore failed, and the failure related to a component called the "Finance DB machine" on this target. It specifies additional context indicating the related disk size and name using the default separators. Note the escaping of the \ in the disk name using an additional "\".

```
emcli publish_event  -target_name="my acme target" -target_type="oracle_acme"
-name="HDD restore failed" -key="Finance DB machine" -message="HDD restoration
failed due to corrupt disk" -severity=MINOR_WARNING -context="disk
size":800GB\;"disk name":\\uddo0111245
```

Example 3

The following example publishes a critical event for "my acme target" indicating that a HDD restore failed, and the failure was related to a component called the "Finance DB machine" on this target. It specifies additional context indicating the related disk size and name. It uses alternate separators, because the name of the disk includes the ":" default separator.

```
emcli publish_event  -target_name="my acme target" -target_type="oracle_acme"
-name="HDD restore failed" -key="Finance DB machine" -message="HDD restoration
failed due to corrupt disk" -severity=CRITICAL -context="disk size"^800GB\;"disk
name"^\\sdd1245:2 -subseparator=context=^
```

publish_metric_extension

Publishes a metric extension for use by all administrators. The metric extension must currently be a deployable draft.

Format

```
emcli publish_metric_extension
      -target_type=<metric_extension_target_type>
      -name=<metric_extension_name>
      -version=<metric_extension_version>
```

Parameters

■ **target_type**

Target type of the metric extension.

■ **name**

Name of the metric extension.

■ **version**

Version of the metric extension to be published.

Example

The following example publishes a metric extension of a given target type, name, and version.

```
emcli publish_metric_extension -target_type=<target type of the metric extension>
-name=<name of the metric extension -version=<version of the metric extension>
```

reassoc_masking_definition

Reassociates an existing masking definition with another database target.

Format

```
emcli reassoc_masking_definition
    -definition_name=masking definition name
    -target_name=database target name
    -target_type=database target type
    [-parameters=name1:value1;name2:value2;...]
    [-credential_name=credential_name]
    [-input_file=parameter_tag:file_path]
```

[ ] indicates that the parameter is optional

Parameters

- **definition_name**

  Masking definition name.

- **target_name**

  New database target name with which to associate the masking definition.

- **target_type**

  New database target type with which to associate the masking definition.

- **parameters**

  List of name-value pairs that represent the credentials required for connecting to the database instance. The supported parameters are db_username, db_password, and db_role.

- **credential_name**

  Name of the database credential. This parameter is mandatory when the db_username and db_password parameters are not specified.

- **input_file**

  Used in conjunction with the parameters option, this option enables you to store parameter values, such as username and password, in a separate file. This option specifies a mapping between a tag and a local file path. The tag is specified in lieu of specific parameter values for the parameters . The tag must not contain colons (:) or semi-colons (;).

Output

Success or failure message along with the details.

Examples

Example 1

The following example reassociates the masking definition mask_hr_data with the new database target testdb2 :

```
emcli reassoc_masking_definition
    -definition_name=mask_hr_data
    -target_name=testdb2
```

```
      -parameters="db_username:system;db_password:password;db_role:NORMAL"
```

Example 2

The following example reassociates the masking definition mask_hr_data with the new database target testdb2. The database password is read from the pwd.txt file.

```
emcli reassoc_masking_definition
      -definition_name=mask_hr_data
      -target_name=testdb2
      -parameters="db_username:system;db_password:PWD_FILE;db_role=SYSDBA"
      -input_file="PWD_FILE:pwd.txt"
```

Example 3

The following example reads the credentials from the preferred credential set DBCredsNormal and reassociates the masking definition.

```
emcli reassoc_masking_definition
-definition_name=mask_hr_data
-target_name=testdb2
```

Example 4

The following example reads the credentials from the preferred credential set DBCredsSYSDBA and reassociates the masking definition.

```
emcli reassoc_masking_definition
-definition_name=mask_hr_data
-target_name=testdb2
-credential_set_name=DBCredsSYSDBA
```

refer_swlib_entity_files

Refers one or more files from an entity revision in the software library.

Format

```
emcli refer_swlib_entity_files
      -entity_rev_id="entity_rev_id"
      -file="<relative_file_path>[;<new_file_name>]"
      -refer_storage="<storage_location_name>;<storage_type>"
      [-use_latest_revision]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **entity_rev_id**

  Identifier of the entity revision. The Software Library home page exposes the identifier for folders and entities as a custom column (Internal ID) and is hidden by default.

- **file**

  Relative path of the file to be referred from the specified storage location. The file name stored in the software library is defaulted to the name of the file being referred. You can optionally specify a different file name, separated by a semi-colon (;).

- **refer_storage**

  The storage location and type for referring to files, separated by a semi-colon (;). The location specified must be in 'active' status. The storage type can be Http, Nfs, or ExtAgent.

- **use_latest_revision**

  Indicates that the latest revision of the entity be used instead of the revision identified by entity_rev_id.

Example

The following example refers the file 'scripts/perl/script1.pl' in the HTTP reference file location 'myScripts' from the entity revision identified. The file name associated will be 'new_script.pl'. The identifier of the updated revision is output.

```
emcli refer_swlib_entity_files
      -entity_rev_id="oracle:defaultService:em:provisioning:1:cmp:
       COMP_Component:SUB_Generic:B1B1880C6A8C62AAE040548C42832D14:0.1"
      -file="scripts/perl/script1.pl;new_script.pl"
      -refer_storage="myScripts;Http"
      -use_latest_revision
```

refresh_coherence

Refreshes one or more Coherence clusters.

Format

```
emcli refresh_coherence
        -input_file=coherence_refresh_file:file_path
        [-debug]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **input_file**

  Fully-qualified path to a CSV-formatted file listing Coherence cluster target per line. For example:

  ```
  ClusterA
  ClusterB
  ```

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Examples

The following example reads the my_clusters_name.csv file to determine the clusters to be refreshed to Cloud Control, and then refreshes them.

```
emcli refresh_coherence
  -input_file=coherence_refresh_file:c:\emcli\my_clusters_names.csv
```

refresh_wls

Enables/disables a refresh for one or more Oracle WebLogic Server Domains (target type --> weblogic_domain). This verb reads a file labeled domain_refresh_file in order to refresh the WebLogic Server. The domain_refresh_file is required; refresh cannot occur without it. You must create the file prior to performing refresh.

Format

```
emcli refresh_wls
        -input_file=domain_refresh_file:file_path
        [-debug]

[ ] indicates that the parameter is optional
```

Parameters

- **input_file=domain_refresh_file**

  Fully-qualified path of the CSV(Comma-Separated Values) file that contains multiple lines of the Target name and Refresh action (Enable/Disable refresh of the WLS domains/farms to be refreshed).

  Note the following advisory information about the format of domain_refresh_file:

  - The target name should be the fully-qualified name of the domain target.

  - Every target is treated as type weblogic_domain.

  - Valid values of the refresh option are "E", "D", and "R". "E" enables a refresh for the WLS Domain, "D" disables the refresh for the WLS Domain, and "R" removes targets that are deleted from the WebLogic Domain.

  - A comma (,) is used as the delimiter.

  - The total number of tokens in each line is fixed, and should be equal to 2.

  - The order of parameters is fixed. You must provide the parameters in the same order as specified below in the sample file structure for domain_refresh_file:

    ```
    /Farm01_base_domain/base_domain,D
    /Farm02_base_domain/base_domain,E
    /Farm03_base_domain/base_domain,R
    ```

    The first entry disables the refresh for target /Farm01_base_domain/base_domain, the second entry enables a refresh for target /Farm02_base_domain/base_domain, and the third entry removes targets from Enterprise Manager that are deleted from /Farm03_base_domain/base_domain.

- **debug**

  Runs the verb in verbose mode for debugging purposes.

Example

```
$emcli refresh_wls
        -input_file=domain_refresh_file:/tmp/refresh/emcli/
            domain_refresh_file.csv -debug
```

reimport_swlib_metadata

Re-imports software library metadata from the OMS and deployed plug-in Oracle Homes. Any Oracle-owned entity with missing files is restored to the corresponding upload storage location.

Format

```
emcli reimport_swlib_metadata
```

Parameters

None.

relocate_targets

Moves all of the collections and blackouts for targets from the source Agent to the destination Agent, and makes the destination Agent the monitoring Agent for these targets in Enterprise Manager.

Format

```
emcli relocate_targets
      -src_agent=<source_agent_target_name>
      -dest_agent=<dest_agent_target_name>
      -target_name=<name_of_target_to_be_relocated>
      -target_type=<type_of_target_to_be_relocated>
      -copy_from_src
      -changed_param=<propName>:<propValue>
      -input_file:dupTargets=<targets_contents>
      -input_file:moveTargets="complete path to file containing targets with
        overridden property values"
      -copy_from_src [-changed_param=<propName>:<propValue>]*
      [-ignoreRelatedTargets]
      [-noHostColumnUpdate]
      [-ignoreTimeSkew=yes]
      [-force=yes]
```

```
[ ] indicates that the parameter is optional
```

> **Note:** To relocate a composite target, you must specify the input_file:dupTargets , and you cannot combine -target_type or -target_name.

Modes

There are two modes for this verb:

- **Create Mode**

  This mode creates a list of targets on the destination Management Agent that already exists and is monitored by the source Management Agent in Enterprise Manager. It moves all the collections and blackouts for these targets from the source Management Agent to the destination Management Agent, and makes the destination Agent the monitoring Agent for these targets in Enterprise Manager.

  ```
  emcli relocate_targets -src_agent=<source_agent>
        -dest_agent=<destination_agent>
        -input_file=dupTarget:<complete_path_to_file>;
        [-ignoreTimeSkew=yes]
  ```

  > **Tip:** See the Examples section for more samples of the create mode.

- **Exist Mode**

  In this mode, the target also exists at the destination.

  ```
  emcli relocate_targets
        -src_agent=<source_agent_target_name>
        -dest_agent=<destination_agent_target_name>
        -target_name=<target_name>
        -target_type=<target_type>
        [-ignoreTimeSkew=yes]
  ```

```
[-force=yes]
```

In all cases, relocation moves all collections and blackouts for these targets from the source Agent to destination Agent, and makes the destination Agent the monitoring Agent for these targets in Enterprise Manager.

Parameters

- **src_agent**

  Management Agent currently monitoring the targets. If srcAgent is not known, enter currentOwner as the argument.

- **dest_agent**

  Management Agent that should monitor the targets.

- **target_name**

  Name of the target that needs to be moved.

- **target_type**

  Type of target that needs to be moved.

- **changed_param**

  The value of the propName property in the target should be changed to propValue.

- **input_file=dupTargets**

  Takes a file name that contains all the targets and its properties as seen in `targets.xml`. The contents of the file must have the same format as `targets.xml`.

  To relocate a composite target, you must specify the input_file:dupTargets, and you cannot combine -target_type or -target_name.

- **input_file=moveTargets**

  Takes a file name that contains a list of targets, one per line, in the following format:

  ```
  <targetType>:<targetName>[;<propName>=<propValue>]*
  ```

- **copy_from_src**

  Copies target properties from the source Agent.

- **ignoreTimeSkew**

  If specified, the target is relocated, ignoring the time skew between the source and destination Agent.

- **ignoreRelatedTargets**

  Moves related targets when not specified. Specified to move onlhy the targets on the command line.

- **noHostColumnUpdate**

  Preserves the host of the relocated tartet when specified. Otherwise, the host is updated to be the new Agent's host.

- **force**

If the command is executed with the `-force=yes` switch, the composite target is automatically relocated with its related targets. If the command is executed without this switch, an error message appears if it is a composite target.

Output

Output message of the command execution.

Examples

Example 1

The following Create Mode example creates a target on the destination Agent by copying the target property content from the source Agent, while allowing some property values to be changed.

```
emcli relocate_targets
      -src_agent=<source_agent>
      -dest_agent=<destination_agent>
      -target_name=<target_name>
      -target_type=<target_type>
      -copy_from_src
      [-ignoreTimeSkew=yes]
      [-changed_param=<Propname>:<Value>]*
```

Example 2

The following Create Mode example creates a list of targets on the destination Agent specified in the moveTargets file. You can specify property value overrides.

```
emcli relocate_targets
      -src_agent=<source_agent>
      -dest_agent=<destination_agent>
      -input_file=moveTargets:<complete_file_path>
      [-ignoreTimeSkew=yes]
```

Example 3

The following example creates a list of targets on the destination Agent that already exists and is monitored by the source Agent in Enterprise Manager.

```
emcli relocate_targets
        -src_agent=<source agent>
        -dest_agent=<destination agent>
        {-ignoreTimeSkew=yes}
        -input_file=dupTarget:<complete file path>;
```

Example 4

The following exampole creates a target on the destination Agent by copying the target property content from the source Agent while allowing some property values to be changed.

```
emcli relocate_targets
        -src_agent=<source agent>
        -dest_agent=<destination agent>
        -target_name=<target name>
        -target_type=<target type>
        -copy_from_src
        {-ignoreRelatedTargets}
        {-noHostColumnUpdate}
        {-ignoreTimeSkew=yes}
```

```
                    [-changed_param=<Propname>:<Value>]*
```

Example 5

The following example creates a list of targets on the destination Agent specified in
the moveTargets file. You can specify property value overrides.

```
emcli relocate_targets
        -src_agent=<source agent>
        -dest_agent=<destination agent>
        {-ignoreTimeSkew=yes}
        -input_file=moveTargets:<complete file path>;
```

remove_beacon

Removes a beacon from the monitoring set of beacons.

Format

```
emcli remove_beacon
      -name=<target_name>
      -type=<target_type>
      -bcnName=<beacon_name>
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **bcnName**

  Beacon name to remove.

Examples

The following example removes MyBeacon from the MyTarget service target of type generic_service.

```
emcli remove_beacon -name='MyTarget' -type='generic_service'
      -bcnName='MyBeacon'
```

remove_service_system_assoc

Removes the system for a given service.

Format

```
emcli remove_service_system_assoc
      -name='name'
      -type='type'
```

Parameters

- **name**

  Service name.

- **type**

  Service type.

Examples

The following example removes the system for the generic service named my service.

```
emcli remove_service_system_assoc
      -name='my service' -type='generic_service'
```

remove_swlib_storage_location

Removes a storage location from the software library. The alternate storage location where the existing files need to be migrated should also be specified. For upload file storage types, OMS shared and the OMS Agent file system, a job is submitted to perform the migration of files, subsequent to which the location is removed. For these upload file storage types, the alternate location need not be of the same storage type, which is not the case for locations of referenced file storage types.

Format

```
emcli remove_swlib_storage_location
        -name="src_location_name"
        -type="OmsShared|OmsAgent|Http|Nfs|ExtAgent"
        -migrate_to_loc="dest_location_name"
        [-migrate_to_type="OmsShared|OmsAgent|Http|Nfs|ExtAgent"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the storage location to be removed.

- **type**

  Type of storage location, which can be one of:

  ```
  OmsShared
  OmsAgent
  Http
  Nfs
  ExtAgent
  ```

- **migrate_to_loc**

  Name of the alternate storage location where existing files need to be migrated.

- **migrate_to_type**

  Type of the alternate storage location, which can be one of:

  ```
  OmsShared
  OmsAgent
  Http
  Nfs
  ExtAgent
  ```

  The default is the storage type of the location being removed.

  > **Note:** This option can be different from the type option specified only for OmsShared and OmsAgent storage types. For all other storage types, migrating files across storage types is not supported, and therefore, type and migrate_to_type (if specified) must be the same.

Examples

Example 1

The following example removes an OMS shared file system storage location named 'myOMSSharedLocation' and migrates all of its files to another OMS shared file system storage location named 'myNewOMSSharedLocation'. A job is submitted for performing the file migration. The location being removed will be moved to 'Inactive' status during file migration and subsequently removed.

```
emcli remove_swlib_storage_location
      -name="myOMSSharedLocation"
      -type="OmsShared"
      -migrate_to_loc="myNewOMSSharedLocation"
```

Example 2

The following example removes an OMS shared file system storage location named 'myOMSSharedLocation' and migrates all of its files to an OMS Agent file system storage location named 'myNewAGTLocation'. A job is submitted for performing the file migration. The location being removed will be moved to 'Inactive' status during file migration and subsequently removed.

```
emcli remove_swlib_storage_location
      -name="myOMSSharedLocation"
      -type="OmsShared"
      -migrate_to_loc="myNewAGTLocation"
      -migrate_to_type="OmsAgent"
```

Example 3

The following example removes an HTTP storage location named 'myHTTPLocation' and migrates all of its files to another HTTP storage location named 'myNewHTTPLocation'.

```
emcli remove_swlib_storage_location
      -name="myHTTPLocation"
      -type="Http"
      -migrate_to_loc="myNewHTTPLocation"
```

remove_target_property

Removes the target property from all targets of the specified target type. This also removes all values associated with this target property.

Format

```
emcli remove_target_property
      -target_type="target_type"
      -property="property_name"
```

Parameters

- **target_type**

  Target type for which you want to remove this property. To remove this property from all target types for which it is defined, you can specify the "*" wildcard character.

- **property**

  Name of the property you want to remove. Property names are case-sensitive. You cannot remove the following Oracle-provided target properties:

  Comment, Deployment Type, Line of Business, Location, Contact

Examples

Example 1

The following example removes the target property Owner from all targets of type oracle_database. This also removes all values associated with this target property.

```
emcli remove_target_property  -target_type="oracle_database" -property="Owner"
```

Example 2

The following example removes the target property Owner from all targets. This also removes all values associated with this property for all target types.

```
emcli remove_target_property  -target_type="*" -property="Owner"
```

rename_target

Renames the repository-side target.

Format

```
emcli rename_target
      -target_type=<type1>
      -target_name=<old_target1>
      -new_target_name=<new_target1>
```

Parameters

- **target_type**

  Target type of the target being renamed.

- **target_name**

  Existing name of the target.

- **new_target_name**

  New name of the target.

Examples

The following example renames the repository-side target.

```
emcli rename_target
      -target_type="oracle_em_service"
      -target_name="TestService1"
      -new_target_name="NewTestService1"
```

reschedule_instance

Reschedules a submitted procedure instance. You can only reschedule scheduled instances.

Format

```
emcli reshedule_instance
      -instance=<instance_guid>
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      -schedule=
          start_time:yyyy/MM/dd HH:mm;
          [tz:<java_timezone_ID>];
          [grace_period:xxx]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  GUID of the instance to execute.

- **exec**

  Execution GUID.

- **name**

  Execution name.

- **owner**

  Execution owner.

- **schedule**

  Schedule for the procedure instance:

  – **start_time** — When the procedure should start.

  – **tz —** Optional time zone ID.

  – **grace_period** — Optional grace period in minutes.

Examples

```
emcli reschedule_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-schedule="start_time:2011/8/21 21:23;tz:America/New_York;grace_period:60"
```

resecure_agent

Resecures a Management Agent already secured. This verb requires operator privilege or full privilege on the Management Agent.

Format

```
emcli resecure_agent
        -agent_name="agent_target_name"
        -registration_pwd="registration_password"
        [-host_username="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name="credential_name"]
        [-credential_setname="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **registration**

  Registration password to securely communicate with OMS.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds".

Examples

Example 1

```
emcli resecure_agent -agent_name="agent.example.com:1234"
                     -registration_pwd="test_pwd"
                     -host_username="test_user"
                     -host_pwd="test"
```

Example 2

```
emcli resecure_agent -agent_name="agent.example.com:1234"
                     -registration_pwd="test_pwd"
                     -credential_name="MyMachineCredential"
```

Example 3

```
emcli resecure_agent -agent_name="agent.example.com:1234"
                     -registration_pwd="test_pwd"
                     -credential_setname="HostCreds"
```

restart_agent

Restarts a Management Agent. This verb requires operator privilege or full privilege on the Management Agent.

Format

```
emcli restart_agent
       -agent_name="agent_target_name"
       [-host_username="agent_host_username" -host_pwd="agent_host_password"]
       [-credential_name="credential_name"]
       [-credential_setname="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds".

Examples

Example 1
```
emcli restart_agent -agent_name="agent.example.com:1234"
                        -host_username="test_user"
                        -host_pwd="test"
```

Example 2
```
emcli restart_agent -agent_name="agent.example.com:1234"
                        -credential_name="MyMachineCredential"
```

Example 3
```
emcli restart_agent -agent_name="agent.example.com:1234"
                        -credential_setname="HostCreds"
```

resume_instance

Resumes a suspended deployment instance.

Format

```
emcli resume_instance
      -instance=<instance_guid>
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- **name**

  Name of the execution.

- **owner**

  Owner of the execution.

Examples

```
emcli resume_instance -instance=16B15CB29C3F9E6CE040578C96093F61
```

resync_agent

Performs a Management Agent recovery. A message is issued if the specified Management Agent does not exist.

Format

```
 emcli resyncAgent
        -agent="agent_name"
        [-keep_blocked]
```

[ ] indicates that the parameter is optional

Parameters

- **agent**

  Name of the Management Agent for which to perform the Agent recovery.

- **keep_blocked**

  Leaves the Management Agent blocked even if the resync succeeds. By default, the Agent becomes unblocked if the resync is successful.

Examples

```
emcli resyncAgent -agent="ushost1.mycompany.com:3872"
```

retry_instance

Retries a failed instance or failed step.

Format

```
emcli retry_instance
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      [-stateguid=<state_guid>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- **name**

  Name of the execution.

- **owner**

  Owner of the execution.

- **stateguid**

  Comma-separated list of state GUIDs.

Examples

```
emcli retry_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168

emcli retry_instance -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid='51F762417C4943DEE040578C4E087168,51F762417C4944DEE040578C4E087168'
```

retry_job

Restarts a previously failed job execution.

Format

```
emcli retry_job
      -exec_id="executionID"
      [-noheader]
      [-script | -format=
            [name:<pretty|script|csv>];
            [column_separator:"column_sep_string"];
            [row_separator:"row_sep_string"];
      ]

[ ] indicates that the parameter is optional
```

Parameters

- **exec_id**

  ID of the job execution to be retried. Use the get_jobs verb to obtain specific job execution IDs.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **format**

  Format specification (default is –format="name:pretty").

  - format="name:pretty" prints the output table in a readable format not intended to be parsed by scripts.

  - format="name:script" sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - format="name:csv" sets the column separator to a comma and the row separator to a newline.

  - format=column_separator:"column_sep_string" column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Output Columns:

Execution ID

Examples

The following example restarts the job execution with Id 123456789012345678901234567890012 and displays a new execution ID.

```
emcli retry_job -exec_id=123456789012345678901234567890012
```

revoke_license_no_validation

Revokes licenses on a set of user-specified packs, or all packs to a set of user-specified targets, or all targets belonging to the input licensable target type.

For 11g database targets, you cannot enable or disable the Database Diagnostic and Tuning Packs through the user interface. You need to set the control_management_pack_access initialization parameter to manage your licenses. For information about this parameter, see the Enterprise Database Management chapter of *Oracle Enterprise Manager Licensing Information*.

> **Tip:** You can use this verb to revoke licenses for standalone target types, such as hosts and databases, but you cannot use this verb to revoke licenses for the parent Application Server (oracle_ias) target type, which has dependent target types of OC4J, Jserv, Web Cache, and so forth. To do this, use the revoke_license_with_validation verb instead.
>
> For example, for pack ias_config and an Application Server target of AS1 with an associated dependent target of OC4J1, this verb revokes the license to AS1, but this does not propagate to OC4J1.

Format

```
emcli revoke_license_no_validation
      -type="target_type"
      [-targets="tname1;tname2;..."]
      [-packs="pack1;pack2;..."]
      [-file="file_name"]
      [-displayAllMessages]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Target type as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks. You can specify only one target type at a time; for example, -type="oracle_database".

- **targets**

  Targets should be specified in the following sequence:

  ```
  TargetName1;TargetName2;
  ```

  For example:

  ```
  -targets="database1;database2;database3;"
  ```

  The semi-colon (;) is the target separator.

  See the "Examples" section below for information about providing arguments for the targets .

- **packs**

  License packs should be specified in the following sequence:

  ```
  pack1;pack2;
  ```

For example:

```
-packs="db_diag;db_config;"
```

The semi-colon (;) is the pack separator.

See the "Examples" section below for information about providing arguments for the pack .

- **file**

   Specify the file name, including the complete path. For example:

   ```
   -file="/usr/admin1/db_license.txt"
   ```

   The file should contain the list of targets and packs according to the following cases:

   – If you only need to provide a list of targets, use the following format:

   ```
   targets=database1;database2;database3;
   ```

   – If you only need to provide a list of packs, use the following format:

   ```
   packs=db_diag;db_config;
   ```

   – If you need to provide a list of both targets and packs, use the following format:

   ```
    targets=database1;database2;database3;
    packs=db_diag;db_config;
   ```

- **displayAllMessages**

   Displays all messages. Only error messages are displayed by default. "=value" is not allowed on the command line.

Examples

Example 1 and Example 2 below revoke licenses of specific packs for specific targets. In order to know which target types and pack names you can pass as arguments, you can use the view named mgmt_license_view to see a list of licensable targets, their target types, and the list of packs licensed on them.

To obtain this information, do the following:

1. Access SQL*Plus with your username and password, using sysman or other user that has access to sysman.mgmt_license_view.

2. Select a distinct pack name from sysman.mgmt_license_view, where:

   ```
   target_type=<oracle_database>
   ```

The following example shows pack names for an Oracle database you specify as the target type.

```
PACK_NAME
------------------
db_config
provisioning
db_sadm
db_tuning
db_diag
provisioning_db
db_chgmgt
```

```
7 rows selected.
```

Based on this information, to revoke a license to the database1 target for the db_chgmgt pack, you would enter the following command:

```
emcli revoke_license_no_validation -type="oracle_database" -targets="database1"
-packs="db_chgmgt"
```

The only limitation of mgmt_license_view is that it only lists the packs for a target type where the pack is granted to at least one target of that type. That is, if the pack is not granted to any target of that type, mgmt_license_view cannot provide any information.

Example 1

The following example revokes the license of the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_no_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 2

The following example revokes the license of the db_diag and db_config packs to all database targets in the setup:

```
emcli revoke_license_no_validation -type="oracle_database"
        -packs="db_diag;db_config;"
```

Example 3

The following example revokes the license of all packs (applicable to database targets) to database1, database2, and database3 targets in the setup:

```
emcli revoke_license_no_validation -type="oracle_database"
          -targets="database1;database2;database3;"
```

Example 4

The following example revokes the license of all packs (applicable to database targets) to all database targets in the setup:

```
emcli revoke_license_no_validation -type="oracle_database"
```

Example 5

The following example uses a text file to pass targets and pack names as the argument. It revokes the license of the db_diag and db_config packs to the database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_no_validation -type="oracle_database"
          -file="/usr/admin1/db_license.txt"
             targets=database1;database2;database3;
                   packs=db_diag;db_config;
```

where the content of the "/usr/admin1/license/db_license.txt" file is as follows:

```
targets=database1;database2;database3;
packs=db_diag;db_config;
```

revoke_license_with_validation

Revokes licenses on a set of user-specified packs, or all packs to a set of user-specified targets, or all targets belonging to the input licensable target type as per business rules.

For 11g database targets, you cannot enable or disable the Database Diagnostic and Tuning Packs through the user interface. You need to set the control_management_ pack_access initialization parameter to manage your licenses. For information about this parameter, see the Enterprise Database Management chapter of *Oracle Enterprise Manager Licensing Information*.

> **Tip:** You can use this verb to revoke licenses for standalone target types, such as hosts and databases, and you also use this verb to revoke licenses for the parent Application Server (oracle_ias) target type, which has dependent target types of OC4J, Jserv, Web Cache, and so forth.
>
> For example, for pack ias_config and an Application Server target of AS1 with an associated dependent target of OC4J1, this verb revokes the license to AS1 and also propagates to OC4J1 (and all other dependent targets associated with AS1).
>
> To revoke licenses for only standalone target types, use the revoke_ license_no_validation verb.

Format

```
emcli revoke_license_with_validation
      -type="target_type"
      [-targets="tname1;tname2;..."]
      [-packs="pack1;pack2;..."]
      [-file="file_name"]
      [-displayAllMessages]

[ ] indicates that the parameter is optional
```

Parameters

- **type**

  Target type as it exists in the database. Names cannot contain colons (:), semi-colons (;), or any leading or trailing blanks. You can specify only one target type at a time; for example, -type="oracle_database".

- **targets**

  Targets should be specified in the following sequence:

  ```
  TargetName1;TargetName2;
  ```

  For example:

  ```
  -targets="database1;database2;database3;"
  ```

  The semi-colon (;) is the target separator.

  See the "Examples" section below for information about providing arguments for the targets .

- **packs**

  License packs should be specified in the following sequence:

```
pack1;pack2;
```

For example:

```
-packs="db_diag;db_config;"
```

The semi-colon (;) is the pack separator.

See the "Examples" section below for information about providing arguments for the packs.

- **file**

  Specify the file name, including the complete path. For example:

  ```
  -file="/usr/admin1/db_license.txt"
  ```

  The file should contain the list of targets and packs according to the following cases:

  - If you only need to provide a list of targets, use the following format:

    ```
    targets=database1;database2;database3;
    ```

  - If you only need to provide a list of packs, use the following format:

    ```
    packs=db_diag;db_config;
    ```

  - If you need to provide a list of both targets and packs, use the following format:

    ```
     targets=database1;database2;database3;
     packs=db_diag;db_config;
    ```

- **displayAllMessages**

  Displays all messages. Only error messages are displayed by default. "=value" is not allowed on the command line.

Examples

Example 1 and Example 2 below revoke licenses of specific packs for specific targets. In order to know which target types and pack names you can pass as arguments, you can use the view named mgmt_license_view to see a list of licensable targets, their target types, and the list of packs licensed on them.

To obtain this information, do the following:

1. Access SQL*Plus with your username and password, using sysman or other user that has access to sysman.mgmt_license_view.

2. Select a distinct pack name from sysman.mgmt_license_view, where:

   ```
   target_type=<oracle_database>
   ```

The following example shows pack names for an Oracle database you specify as the target type.

```
PACK_NAME
-------------------
db_config
provisioning
db_sadm
db_tuning
db_diag
```

```
provisioning_db
db_chgmgt

7 rows selected.
```

Based on this information, to revoke a license to the database1 target for the db_chgmgt pack, you would enter the following command:

```
emcli revoke_license_with_validation -type="oracle_database" -targets="database1"
-packs="db_chgmgt"
```

The only limitation of mgmt_license_view is that it only lists the packs for a target type where the pack is granted to at least one target of that type. That is, if the pack is not granted to any target of that type, mgmt_license_view cannot provide any information.

Example 1

The following example revokes the license of the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_with_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 1

The following example revokes the license of the db_diag and db_config packs to database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_with_validation -type="oracle_database"
          -targets="database1;database2;database3;" -packs="db_diag;db_config;"
```

Example 2

The following example revokes the license of the db_diag and db_config packs to all database targets in the setup:

```
emcli revoke_license_with_validation -type="oracle_database"
       -packs="db_diag;db_config;"
```

Example 3

The following example revokes the license of all packs (applicable to database targets) to database1, database2, and database3 targets in the setup:

```
emcli revoke_license_with_validation -type="oracle_database"
          -targets="database1;database2;database3;"
```

Example 4

The following example revokes the license of all packs (applicable to database targets) to all database targets in the setup:

```
emcli revoke_license_with_validation -type="oracle_database"
```

Example 5

The following example uses a text file to pass targets and pack names as the argument. It revokes the license of the db_diag and db_config packs to the database1, database2, and database3 targets (oracle_database target type):

```
emcli revoke_license_with_validation -type="oracle_database"
          -file="/usr/admin1/db_license.txt"
              targets=database1;database2;database3;
```

```
                                      packs=db_diag;db_config;
```

where the content of the "/usr/admin1/license/db_license.txt" file is as follows:

```
targets=database1;database2;database3;
packs=db_diag;db_config;
```

revoke_privs

Revokes the privileges from an existing Enterprise Manager user or Enterprise Manager role.

Format

```
emcli revoke_privs
        -name="username|rolename"
        [-privilege="name[;secure-resource-details]]"
        [-separator=privilege="sep_string"]
        [-subseparator=privilege="subsep_string"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  User name or role name from which privileges will be revoked.

- **privilege**

  Privilege to grant to this administrator. You can specify this option more than once. The original administrator privileges will be revoked. Specify <secure_resource_details> as:

  ```
  resource_guid|[resource_column_name1=resource_column_value1[:resource_column_
  name2=resource_column_value2]..]"
  ```

- **separator**

  Specify a string delimiter to use between name-value pairs for the value of the -privilege option. The default separator delimiter is a semi-colon (;).

- **subseparator**

  Specify a string delimiter to use between name and value in each name-value pair for the value of the -privilege option. The default subseparator delimiter is a colon (:).

Examples

Example 1

For user1, the following example revokes full control of the jobs with ID 923470234ABCDFE23018494753091111, and revokes full control on the target host1.example.com:host:

```
emcli revoke_privs
        -name="user1"
        -privilege="FULL_JOB;923470234ABCDFE23018494753091111"
        -privilege="FULL_TARGET;host1.example.com:host"
```

Example 2

The following example revokes the target privileges from Enterprise Manager role Role1:

```
emcli revoke_privs
        -name="Role1"
        -privilege="FULL_TARGET;host1.example.com:host"
```

revoke_roles

Revokes the roles to an existing Enterprise Manager user or Enterprise Manager role.

Format

```
emcli revoke_roles
      -name="username|rolename"
      [-roles="role1;role2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  User name or role name from which roles will be revoked.

- **roles**

  Roles, which will be revoked from the Enterprise Manager user or role. You can specify this option more than once.

Examples

```
emcli revoke_roles
      -name="user1"
      -roles="SUPER_USER"

emcli revoke_roles
      -name="Role1"
      -roles="BLACKOUT_ADMIN;MAINTAIN_TARGET"
```

run_avail_diag

Runs diagnostics for an availability algorithm for a test-based service. This is mostly useful when the "last calculated" time stamp is running behind the current time, and the service status has been unresponsive for some time.

Format

```
emcli run_avail_diag
      -name=<target_name>
      -type=<target_type>
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

Examples

```
emcli run_avail_diag  -name='MyTarget' -type='generic_service'
```

run_prechecks

Submits the pre-check operation for any given operation plan.

Format

```
emcli run_prechecks
      -operation_plan=<operation_plan_name>
```

Parameters

- **operation_plan**

  Name of the operation plan.

Examples

```
emcli run_prechecks
      -operation_plan="BISystem1-switchover"
```

run_promoted_metric_diag

Runs promoted metric diagnostics.

Format

```
emcli run_promoted_metric_diag
      -name=<target_name>
      -type=<target_type>
      -promotedMetricName=<metric_name>
      -promotedColumn=<metric_type>
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **promotedMetricName**

  Promoted metric name.

- **promotedColumn**

  Promoted metric type.

Examples

```
emcli run_promoted_metric_diag  -name='MyTarget' -type='generic_service'
-promotedMetricName='metric1' -promotedColumn='Performance'
```

save_masking_script

Saves a masking script already generated to the specified path or file.

Format

```
emcli save_masking_script
     -definition_name=<masking_definition_name>
     [-path=file path]
     [-file=file name]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **definition_name**

  Masking definition name.

- **path**

  Path for the file name to save the masking script. File name is automatically generated. The path and file options are mutually exclusive. Only an absolute path is allowed.

- **file**

  File name to save the masking script. The file name must include the absolute path. Either the path or file option must be specified.

Output

Success or error messages

Examples

Example 1

The following example saves the masking script for the definition named mask_hr_data to the /tmp directory:

```
emcli save_masking_script
      -definition_name=mask_hr_data
      -path=/tmp/
```

Example 2

The following example saves the masking script for the definition named mask_hr_data to /tmp/abc.sql :

```
emcli save_masking_script
      -definition_name=mask_hr_data
      -file=/tmp/abc.sql
```

save_metric_extension_draft

Save a deployable draft of a metric extension. The metric extension must currently be in an editable state. Once saved as a draft, the metric extension is no longer editable.

Format

```
emcli save_metric_extension_draft
      -target_type=<metric_extension_target_type>
      -name=<metric_extension_name>
      -version=<metric_extension_version>
```

Parameters

- **target_type**

  Target type of the metric extension.

- **name**

  Name of the metric extension.

- **version**

  Version of the metric extension to be saved to the draft.

save_procedure_input

Configures a deployment procedure for execution.

Format

```
emcli save_procedure_input
      [-name="procedure_configuration_name"]
      [-owner="procedure_configuration_owner"]
      [-procedure="procedure_guid"]
      -input_file="file_path\file_name"
      [-grants="access_levels_for_users"]
      [-schedule=
          start_time:yyyy/MM/dd HH:mm;
          tz:{java timezone ID};
          grace_period:xxx;
      ]
      [-notification="procedure status"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the configuration for the procedure.

- **owner**

  Owner of the Procedure configuration.

- **procedure**

  GUID of the procedure to execute.

- **input_file**

  GUID of the procedure to execute. The file_path should point to a file containing the data property file.

- **grants**

  Specifies users and their corresponding access levels as a string of user:privilege pairs, each separated by a semi-colon (;). The user is an Enterprise Manager user name, and the privilege is either VIEW_JOB or FULL_JOB.

  See the example below.

- **schedule**

  Schedule for the deployment procedure. If not specified, the procedure is executed immediately.

  - **start_time** — When the procedure should start.

  - **tz** — Optional timezone ID.

  - **grace_period** — Optional grace period in minutes.

- **notification**

  Status of the procedure.

Example

```
emcli save_procedure_input
    -name=configProcedure -procedure=16B15CB29C3F9E6CE040578C96093F61
    -input_file=/home/data.properties -grants="user1:VIEW_JOB;user2:FULL_JOB"
    -schedule="start_time:2011/8/21 21:23;tz:America/New_York;grace_period:60"
    -notification="scheduled, action required, running"
```

search_patches

Searches patches from the ARU site or software library with the specified search criteria.

Format

```
emcli search_patches
        [-swlib]
        [-patch_name="patch_name"]
        [-product="product_id" [-include_all_products_in_family]]
        [-release="release_id"]
        [-platform="platform_id" | -language="language_id"]
        [-type="patch | patchset"]
        [-noheader]
        [-script | -xml | -format=
                            [name:<pretty|script|csv>];
                            [column_separator:"column_sep_string"];
                            [row_separator:"row_sep_string"];
        ]
```

`[ ]` indicates that the parameter is optional

Parameters

- **swlib**

    Searches patches in the software library if this parameter is provided, whether the current connection mode is online or offline.

- **patch_name**

    Patch name, number, or Sun CR ID. This option is only valid in Simple Search mode. If you provide this option, the Simple Search mode is enabled. If the options specific to Advanced Search mode are provided along with this option, they will not take effect.

- **product**

    Patch product/product family ID. Run the command "emcli list_aru_products" to search the product ID.

- **include_all_products_in_family**

    Takes the specified product ID as a product family ID and includes all products in this product family while searching patches. This option is valid only when you provide the 'product' option.

- **release**

    Patch release ID. Run the command "emcli list_aru_releases" to search for the release ID.

- **platform**

    Patch platform ID. Run the command "emcli list_aru_platforms" to search for the platform ID.

- **language**

    Patch language ID. Run the command "emcli list_aru_languages" to search for the language ID.

- **type**

  Patch type.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This option is equivalent to -format="name:script".

- **xml**

  Displays the patch information in XML format.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  - `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

  - row_separator:"row_sep_string" row-separates the verb output by <row_sep_string>. Rows are separated by the tab character.

Examples

```
emcli search_patches -patch_name=6880880  -platform=226 -swlib

emcli search_patches -patch_name=6880880  -platform=226 -language=0 -xml

emcli search_patches -product=9480  -release=80102030  -platform=226 -type=patch
-format=name:pretty

emcli search_patches -product=9480  -release=80102030 type=patch -xml

emcli search_patches -product=9480  -release=80102030 -script

emcli search_patches -product=9480 -release=80102030 type=patchset
-format=name:csv
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans

set_connection_mode
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

secure_agent

Secures an Agent.

Format

```
emcli secure_agent
        -agent_name="agent_target_name"
        -registration_pwd="registration_password"
        [-host_username ="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name ="credential_name"]
        [-credential_setname ="credential_setname_of_agent"]

[ ] incicates that the paramter is optional
```

Parameters

- **agent_name**

  Name of the Agent target.

- **regustration_pwd**

  Registration password to secure the communication with OMS.

- **host_username**

  User name of the OS user (on the host) who owns the Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Agent. Example: "HostCreds".

Examples

Example 1

```
emcli secure_agent -agent_name="agent.example.com:1234"
                   -registration_pwd="test_pwd"
                   -host_username="test_user"
                   -host_pwd="test"
```

Example 2

```
emcli secure_agent -agent_name="agent.example.com:1234"
                   -registration_pwd="test_pwd"
                   -credential_name="MyMachineCredential"
```

Example 3

```
emcli secure_agent -agent_name="agent.example.com:1234"
                   -registration_pwd="test_pwd"
                   -credential_setname="HostCreds"
```

secure_agents

Secures Agents by providing a list of Agent names, a group name, and input file. If a group name is provided, Enterprise Manager resolves this to a list of Agents that monitor targets in this group. You can also provide an Agent list with an input file to this EM CLI command. For all of these options, you must provide either a user name or password, or the user must have been configured with preferred credentials on Agent targets. This verb submits a job with the list of Agents and the credentials provided as input, and outputs the Job Name and Job ID that you can use to track the status of the job.

This verb also calculates the list of Agents to resecure by filtering out invalid Agents, Agents that are not secure, Agents that are down, and Agents that already have an active job execution. This verb also filters out Agents that are already secured by the correct CA, but you can disable this particular filter by using the -disable_ca_check option .

Format

```
emcli secure_agents
        [-agt_names="agt1;agt2;..."] [-agt_names_file="<file>"]
        [-group_name="group_name"]
        [-use_pref_creds]
        [-username="username"]
        [-password="password"]
        [-disable_ca_check]

[ ] incicates that the paramter is optional
```

Parameters

- **agt_names**

  Semicolon-separated list of Agent names.

- **agt_names_file**

  Absolute path of file containing list of Agent names, each on a new line.

- **group_name**

  Identifies the list of Agents to secure. Enterprise Manager resolves the list of Agents that monitor (not just members of the group) the list of targets in the group.

- **use_pref_creds**

  Uses preferred credentials configured for the Agent to execute the secureAgent job.

- **username**

  User name to execute the secureAgent job at the Agent.

- **password**

  User password to execute the secureAgent job at the Agent.

- **disable_ca_check**

  Disables the check to verify if the Agents are secured with the latest CA.

Examples

```
emcli secure_agents -agt_names="agent_host1:1831;agent_host2:3872" -use_pref_creds

emcli secure_agents -agt_names="agent_host1:1831;agent_host2:3872"
-username=oracleagt

emcli secure_agents -agt_names_file=/tmp/agents_list.txt -use_pref_creds

emcli secure_agents -agt_names_file=/tmp/agents_list.txt -username=oracleagt
```

set_agent_property

Modifies a specific Management Agent property. You can use this command if you have operator privilege for the Management Agent.

Format

```
emcli set_agent_property
      -agent_name="<agent_target_name>"
      -name="<agent_property_name>"
      -value="<agent_property_value>"
      [-new]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **name**

  Name of the Management Agent property you want to modify.

- **value**

  New value for the Management Agent property.

- **new**

  Denotes whether this is a new Agent property being added.

Examples

Example 1
The following example sets the value of the UploadInterval property to 15.

```
emcli get_agent_property -agent_name="agent.example.com:11850"
        -name=UploadInterval
        -value=15
```

Example 3
The following example sets the value of new property 'newprop' in emd.properties to 15.

```
emcli set_agent_property -agent_name="agent.example.com:1234"
        -name=newprop
        -value=15
        -new
```

set_availability

Changes the availability definition of a given service.

Format

```
emcli set_availability
      -name=<target_name>
      -type=<target_type>
      -availType=<availability_type>
      -availOp=<availability_operator>
      [-sysAvailType=<availability_type>]
      [-keycomponents=<'keycomp1name:keycomp1type;
        keycomp2name:keycomp2type;...'>]
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **availType**

  Switches the availability to either test-based or system-based. Can be 'test' or 'system'.

- **availOp**

  If `and`, it uses all key tests/components to decide availability.

  If `or`, it uses any key tests/components to decide availability.

- **sysAvailType**

  Type of availability when the availType is system-based. Sets the availability to either system target directly or selected components of a system.

  – If availability is set to 'system target directly', the system associated with the service needs to define availability[status], systemname, and systemtype are required arguments.

  – If availability is set to 'selected components of a system', systemname, systemtype, and keycomponents are required arguments.

  – If availability is set to 'system target directly', and if availability[status] is not defined, the availability set is invalid. Therefore, the only option that can be set is 'selected components of a system'.

- **keycomponents**

  If `and`, uses all key tests/components to decide availability.

  If `or`, uses any key tests/components to decide availability.

Examples

Example 1

The following example sets the availability of service MyTarget to be based on all key-tests.

```
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='test' -availOp='and'
```

Example 2

The following example sets the availability of service MyTarget to be based on any key-test.

```
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='test' -availOp='or'
```

Example 3

The following example sets the availability of service MyTarget to be based on any key components of a system.

```
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='system' -availOp='or'
                       -keycomponents='database:oracle_database; host1:host'
```

Example 4

The following example sets the availability of service MyTarget to be based on system targets availability.

```
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='system' -availOp='and'
                       -sysAvailType='system target directly'
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='system' -availOp='and'
                       -sysAvailType='selected components of a system'
                       -keycomponents='database:oracle_database; host1:host'
emcli set_availability -name='MyTarget' -type='generic_service'
                       -availType='system' -availOp='or'
                       -sysAvailType='selected components of a system'
                       -keycomponents='database:oracle_database; host1:host'
```

set_connection_mode

Sets the new MOS connection mode.

Format

```
emcli set_connection_mode
      -mode="online | offline"
```

Examples

```
emcli set_connection_mode -mode="offline"
```

```
emcli set_connection_mode -mode="online"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_patch_plan_data
show_patch_plan
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

set_credential

Sets preferred credentials for given users.

> **Note:** This command does not support the COLLECTION credential sets.

Format

```
emcli set_credential
      -target_type="ttype"
      [-target_name="tname"]
      -credential_set="cred_set"
      [-user="user"]
      -columns="col1:newval1;col2:newval2;PDP:SUDO/POWERBROKER;RUNAS:oracle;
         PROFILE:user1..."
      [-input_file="tag1:file_path1;tag2:file_path2;..."]
      [-oracle_homes="home1;home2"]
      [-monitoring]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target. This must be "host" if the -oracle_homes parameter is specified.

- **target_name**

  Name of the target. Omit this argument to set enterprise preferred credentials. This must be the host name if the -oracle_homes parameter is specified.

- **credential_set**

  Credential set affected.

- **user**

  Enterprise Manager user whose credentials are affected. If omitted, the current user's credentials are affected.

- **columns**

  Name and new value of the column(s) to set. Every column of the credential set must be specified. Alternatively, a tag from the -input_file argument can be used so that the credential values are not seen on the command line. You can specify this argument more than once.

- **input_file**

  Path of the file that has the -columns argument(s). This is used to hide passwords. Each path must be accompanied by a tag referenced in the -columns parameter. You can specify this option more than once.

- **oracle_homes**

  Name of Oracle homes on the target host. Credentials will be added/updated for all specified homes.

> **Note:** The list of columns and the credential sets they belong to is included in the metadata file for each target type. This and other credential information is in the `<CredentialInfo>` section of the metadata.

- **monitoring**

  Flag indicating that credentials affected are monitoring credentials. If omitted, the credentials affected are preferred credentials. Monitoring credentials require specifying the target_name option.

Examples

Example 1

```
emcli set_credential
      -target_type=oracle_database
      -target_name=myDB
      -credential_set=DBCredsNormal
      -user=admin1
      -column="username:joe;password:newPass;role:newRole"
```

Example 2

In this example, `FILE1` is a tag to refer to the contents of `passwordFile`. Note that Example 2 has the same effect as Example 1.

```
emcli set_credential
      -target_type=oracle_database
      -target_name=myDB
      -credential_set=DBCredsNormal
      -user=admin1
      -column=FILE1
      -input_file=FILE1:passwordFile
```

Example 3

In this example, the contents of the `passwordFile`: is
`username:joe;password:newPass;role:newRole`

```
emcli set_credential
      -target_type=host
      -target_name=host.example.com
      -credential_set=OHCreds
      -user=admin1
      -column="OHUsername:joe;OHPassword:newPass"
      -oracle_homes="database1;mydb"
```

set_default_pref_cred

Sets a named credential as a default preferred credential. If you decide to use preferred credentials for an Enterprise Manager operation and preferred credentials are not set for the target, the default credentials for this target type that you set are used. Default credentials are set at the target-type level.

Format

```
emcli set_default_pref_cred
        -set_name="set_name"
        -target_type="ttype"
        -credential_name="cred_name"
        [-credential_owner ="owner]"
        [-test]
        [-test_target_name="test_target_name"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **set_name**

    Sets the preferred credential for this credential set.

- **target_type**

    Target type for the credential set.

- **credential_name**

    Name of the credential.

- **credential_owner**

    Owner of the credential. This defaults to the currently logged-in user.

- **test**

    Tests the credential before setting it as the default credential.

- **test_target_name**

    Tests the target name if the global credential is set as the default preferred credential.

Examples

Example 1

The following example sets the named credential MyHostCredentials as the default preferred credential for the target type host as HostCredsNormal.

```
emcli set_default_pref_credential
        -set_name=HostCredsNormal
        -target_type=host
        -credential_name=MyHostCredentials
        -credential_owner="Joe"
```

Example 2

The following example sets the named credential MyHostCredentials as the default preferred credential for the target type host as HostCredsNormal. The command tests

the named credential MyHostCredentials against server1.example.com before setting it as a default preferred credential.

```
emcli set_default_pref_cred
      -set_name=HostCredsNormal
      -target_type=host
      -credential_name=MyHostCredential
      -credential_owner="Joe"
      -test
      -test_target_name=server1.example.com
```

set_key_beacons_tests

Defines key beacons and tests of the service.

Format

```
emcli set_key_beacons_tests
      -name=<target_name>
      -type=<target_type>
      [-beacons=<beacon_names>]+
      [-tests='test1:type1;test2:type2;...']+
      [-removeKey]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **beacons**

  Names of beacons to set as key (or non-key).

- **tests**

  Names and types of tests to set as key (or non-key).

- **removeKey**

  If specified, the mode is (remove key); that is, the specified tests and beacons will be set as non-key.

  If not specified, the mode is (add key); that is, the specified tests and beacons will be set as key.

Examples

Example 1

The following example sets `MyTest/HTTP`, `MyTest2/FTP` and `MyBeacon` as non-key elements of service `MyTarget/generic_service`.

```
emcli set_key_beacons_tests -name='MyTarget' -type='generic_service'
      -tests='MyTest:HTTP;MyTest2:FTP'
      -beacons='MyBeacon' -removeKey
```

Example 2

The following example sets `MyBeacon` and `MyBeacon2` as key beacons of service `MyTarget/generic_service`.

```
emcli set_key_beacons_tests -name='MyTarget' -type='generic_service'
      -beacons='MyBeacon;MyBeacon2'
```

set_logging_property

Sets the property value corresponding to the specified logging property name.

Format

```
emcli set_logging_property
      -property_name="propertyName"
      [-oms_name="omsName"]
      -property_value="propertyValue"

[ ] indicates that the parameter is optional
```

Parameters

- **property_name**

  Name of the logging property whose value needs to be set.

- **oms_name**

  Name of the management server where the logging property needs to be set.

- **property_value**

  Value to be set.

Examples

Example 1

The following example sets the value for the property name "propName" on the management server myhost:1159_Management_Service to "propValue."

```
set_logging_property -property_name=propName -property_value=propValue
-oms_name="myhost:1159_Management_Service"
```

Example 2

The following example sets the value for the property name "propName" to "propValue" on all of the management servers.

```
set_logging_property -property_name=propName -property_value=propValue
```

set_metric_promotion

Creates or edits a metric promotion based on a test or system.

Format

```
emcli set_metric_promotion
      -name=<service_target_name
      -type=<service_target_type
      [-category=Usage/Performance/Business]
      -basedOn=system/test
      -aggFunction=AVG|MAX|MIN|SUM|COPY
      [-promotedMetricName=<promoted_metric>]
      [-promotedMetricColumn=<promoted_metric_column>]
      -promotedMetricKey=<key_value_of_promoted_metric>
      [-metricName=<dependent_metric_name>]
      -column=<dependent_metric_column>
      *[-depTargetType=<target_type_of_dependent_targets>]
      *[-depTargets='target1;target2...']
      *[-depTargetKeyValues='target1:key11|key12|key13..;
         target2:key21|key22|key23..']
      *[-depMetricKeyColumn=<dependent_metric_key_column>]
      **[-testname=<dependent_test_name]
      **[-testtype=<dependent_test_type]
      **[-metricLevel=TXN|STEP|STEPGROUP]
      **[-beacons='bcn1;bcn2..']
      **[-depTestComponent=<step_or_stepgroup_name>]
      [-threshold='critical_threshold_value;warning_threshold_value;
         threshold_operator (EQ|LE|LT|GT|GE)']
      -mode=CREATE|EDIT
```

[ ] indicates that the parameter is optional

* — Might be required if `basedOn` is set to `system`.
** — Might be required if `basedOn` is set to `test`.

Parameters

- **category**

  Defines whether the promoted metric is a usage, performance, or business metric of a service. Category is used to determine the promoted metric name and metric column. If you do not specify this option, you must specify the `promotedMetricName` and `promotedMetricColumn` options.

- **basedOn**

  Determines whether the promotion is test-based or system-based.

- **aggFunction**

  Determines the aggregate function to be used to compute the promoted metric. `AVG/MAX/MIN/SUM` takes average, max, min, and sum of the dependent metrics, respectively. `COPY` only copies over a single dependent metric to the promoted metric.

- **promotedMetricName**

  Promoted metric name. This is optional if the category is specified.

- **promotedMetricColumn**

  Promoted metric column. This is optional if the category is specified.

- **promotedMetricKey**

  Required argument that determines the key value of the promoted metric. It is equivalent to the displayed name of the promoted metric in the UI.

- **metricName**

  Required argument if the dependent metric column is collected by more than one metric.

- **column**

  Dependent metric column.

- **depTargetType**

  All dependent targets should be of this target type.

- **depTargets**

  Specifies the dependent targets. This argument is ignored if you specify `depTargetKeyValues`.

- **depTargetKeyValues**

  Specifies the key values associated with the dependent targets. Specify multiple key values for a single target by repeating the entry in the following format: `'tgt1:key1;tgt1:key2...'`

- **depMetricKeyColumn**

  Required if the dependent metric is a transpose metric. It is the key value that applies to all the dependent targets.

- **testname**

  Defines the name of the test to be used in promoting the metric.

- **testtype**

  Defines the type of test to be used in promoting the metric.

- **metricLevel**

  Some metrics can be promoted on step-level. This option defines the level to be used during promotion.

- **beacons**

  List of beacons to be used for promoting the metric data.

- **depTestComponent**

  If `metricLevel` is not `TXN`, this option is required to specify which step or which step group is being promoted.

- **threshold**

  Defines a threshold on the promoted metric.-mode: The mode can be CREATE or EDIT.

Examples

Example 1

The following example creates a promoted Performance metric with key value `mymetric1` on service `MyTarget` using `MyTest/HTTP`. The promoted metric takes the maximum of the `dns_time` metric column returned by the `MyBeacon` and `mybcn1` beacons. It also has a threshold with 'greater or equal to' operator (GE) with the critical value set to 200 and warning value set to 100.

```
emcli set_metric_promotion -name='MyTarget' -type='generic_service'
      -category=Performance -basedOn=test -aggFunction=MAX
      -testname='MyTest' -testtype=HTTP
      -beacons='MyBeacon, mybcn1'
      -promotedMetricKey=mymetric1 -column=dns_time -metricName=http_response
      -metricLevel=TXN -threshold='200;100;GE' -mode=CREATE
```

Example 2

The following example creates a promoted Usage metric with key value `mymetric1` on service `MyTarget`. The dependent target is `'myhost.mydomain.com'` with type `host`. The promoted metric just copies the `cpuUtil` column of the `Load` metric.

```
emcli set_metric_promotion -name='MyTarget' -type='generic_service'
      -category=Usage -basedOn=system -aggFunction=COPY
      -promotedMetricKey=mymetric1 -column=cpuUtil -metricName=Load
      -depTargets='myhost.mydomain.com' -depTargetType=host
      -mode=CREATE
```

Example 3

The following example creates a promoted `Usage` metric with the key value `AppServerComponentUsage` on service `MyTarget`. The dependent target is `'myapp_server'` with type `'oracle_ias'`. The promoted metric computes the average value of the `cpu.component` metric column for the specified key values.

```
emcli set_metric_promotion -name='MyTarget' -type='generic_service'
      -category=Usage -basedOn=system -aggFunction=AVG
      -promotedMetricKey=AppServerComponentUsage -depTargetType=oracle_ias
      -column=cpu.component
      -metricName=opmn_process_info
      -depTargetKeyValues='myapp_server:petstore;myapp_server:http_server'
      -mode=CREATE
```

Example 4

The following example creates a promoted business metric with key value ordersCount on service MyTarget. The dependent targets are 'onlineOrderService1' and 'onlineOrderService2' with type 'generic_service'. The promoted metric computes the total number of orders from all dependent targets. Note that if the category is business, the category of the metrics from dependent targets should also be business. In the example below, the metric category of the metric named 'Business' is 'Business'.

```
emcli set_metric_promotion -name='MyTarget' -type='generic_service'
        -category=Business -basedOn=system -aggFunction=SUM
        -promotedMetricKey='ordersCount'
        -depTargets='onlineOrderService1;onlineOrderService2'
        -depTargetType='generic_service'
        -metricName='Business' -column='BusinessValue'
        -depMetricKeyValue='Number of Orders Placed'
        -mode=CREATE
```

set_monitoring_credential

Sets a monitoring credential set for a target. You can provide input parameters using command line arguments or the input properties file. It also supports the input_file parameter for passwords and parameter values.

Format

```
emcli set_monitoring_credential
        -target_name=<target_name>
        -target_type=<ttype>
        -set_name=<set_name>
        -cred_type=<credential_type>
        -auth_target_type=<auth_ttype>
        -test
        -input_file=<tag|value>
        -properties_file=<filename>
        -attributes=<p1:v1;p2:v2;...>
```

Parameters

- **target_name**

  Sets the monitoring credential for this target.

- **target_type**

  Target type for the target.

- **set_name**

  Sets the monitoring credential for this credential set name.

- **cred_type**

  Credential type for the credential to set as the monitoring credential.

- **auth_target_type**

  Authenticating target type. Defaults to target_type.

- **test**

  Tests the credential against the target(s) before setting the monitoring credential.

- **input_file**

  Supplies sensitive property values from the file.

- **properties_file**

  Passes all parameters from the file. Values provided on the command line take precedence.

- **attributes**

  Specify credential columns as follows:

  ```
  colname:colvalue;colname:colvalue
  ```

  You can change the separator value using -separator=attributes=<newvalue>, and you can change the sub-separator value using -subseparator=attributes=<newvalue>.

Examples

Example 1

The following example sets the monitoring credential set DBCredsMonitoring for the target testdb.example.com:oracle_database with user name foo, password bar, and role normal.

```
emcli set_monitoring_credential
        -target_name=testdb.example.com
        -target_type=oracle_database
        -set_name=DBCredsMonitoring
        -cred_type=DBCreds
        -attributes="DBUserName:foo;DBPassword:bar;DBRole:normal"
```

Example 2

The following example reads the password from the mypasswordfile.txt file.

```
emcli set_monitoring_credential
        -target_name=testdb.example.com
        -target_type=oracle_database
        -set_name=DBCredsMonitoring
        -cred_type=DBCreds
        -attributes="DBUserName:foo;DBPassword:tag;DBRole:normal"
        -input_file="tag:mypasswordfile.txt"
```

Example 3

The following example prompts for the password from standard input.

```
emcli set_monitoring_credential
        -target_name=testdb.example.com
        -target_type=oracle_database
        -set_name=DBCredsMonitoring
        -cred_type=DBCreds
        -attributes="DBUserName:foo;DBRole:normal;DBPassword:"
```

Example 4

The following example specifies prop1.txt as a multi-line Java properties file, in which each line contains a parameter=value format. You can provide the password in the same file or not specify it. If not specified, you are prompted for it.

```
emcli set_monitoring_credential
        -properties_file=prop1.txt
```

Example 5

The following example sets the monitoring credential set DBCredsMonitoring for the target testdb.oracle.com:oracle_database with a user name of foo, password of bar, and role of normal. The credential is tested before setting the monitoring credential.

```
emcli set_monitoring_credential
        -target_names="testdb1;testdb2"
        -target_type=oracle_database
        -set_name=DBCredsMonitoring
        -cred_type=DBCreds
        -attributes="DBUserName:foo;DBPassword:bar;DBRole:normal"
        -test
```

set_oms_property

Sets the property value corresponding to the specified property name.

Format

```
emcli set_oms_property
      -property_name="propertyName"
      [-oms_name="omsName"]
      -property_value="propertyValue"

[ ] indicates that the parameter is optional
```

Parameters

- **property_name**

  Name of the property whose value needs to be set.

- **oms_name**

  Name of the management server for which the property needs to be set.

- **property_value**

  Property value to be set.

Examples

Example 1

The following example sets the value for the property name "propName" on the management server myhost:1159_Management_Service to "propValue."

```
set_oms_property -property_name=propName -property_value=propValue -oms_
name="myhost:1159_Management_Service"
```

Example 2

The following example sets the value for the property name "propName" to "propValue" on all of the management servers.

```
set the value for the property name "propName" to "propValue" on all the
management servers
```

set_patch_plan_data

Sets user-editable data. The get_patch_plan_data verb is useful when used preceding this verb.

Format

```
emcli set_patch_plan_data
      -name="name"
      -input_file=data:"file_path"
      [-impact_other_targets="add_all|add_original_only|cancel"]
      [-problems_assoc_patches="ignore_all_warnings|cancel"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Sets the preferred credential for this credential set.

- **input_file**

  Sets the preferred credential for this target.

- **impact_other_targets**

  Target type for the target/credential set.

- **problems_assoc_patches**

  Name of the credential.

Examples

```
emcli set_patch_plan_data -name="plan name"
-input_file=data:"/tmp/patchplan.pros"

emcli set_patch_plan_data -name="plan name"
-input_file=data:"/tmp/patchplan.pros" -impact_other_targets="add_all"

emcli set_patch_plan_data -name="plan name"
-input_file=data:"/tmp/patchplan.pros" -impact_other_targets="add_all"
-problems_assoc_patches="ignore_all_warnings"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
show_patch_plan
submit_patch_plan

upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

set_preferred_credential

Sets a named credential as a target-preferred credential for the user.

Format

```
emcli set_preferred_credential
      -set_name="set_name"
      -target_name="target_name"
      -target_type="ttype"
      -credential_name="cred_name"
      [-credential_owner ="owner]"
      [-test]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **set_name**

  Sets the preferred credential for this credential set.

- **target_name**

  Sets the preferred credential for this target.

- **target_type**

  Target type for the target/credential set.

- **credential_name**

  Name of the credential.

- **credential_owner**

  Owner of the credential. This defaults to the currently logged in user.

- **test**

  Tests the credential against the target_name before setting the preferred credential.

Examples

Example 1

The following example sets the named credential MyHostCredentials as the target preferred credential for the target test.example.com:host as HostCredsNormal.

```
emcli set_preferred_credential
      -set_name=HostCredsNormal
      -target_name=test.oracle.com
      -target_type=host
      -credential_name=MyHostCredentials
      -credential_owner="Joe"
```

Example 2

The following example sets the named credential MyDBCredentials as the target preferred credential for the target myDB:oracle_database as Normal Database Credentials. The command tests the named credential against myDB:oracle_database before setting the preferred credential.

```
emcli set_preferred_credential
        -target_type=oracle_database
        -target_name=myDB
        -set_name=DBCredsNormal
        -credential_name=MyDBCredentials
        -credential_owner="Joe"
        -test
```

Example 3

The following example sets the named credential MyDBCredentials as the target preferred credential for the target myDB:oracle_database as SYSDBA database credentials.

```
emcli set_preferred_credential
        -target_type=oracle_database
        -target_name=myDB
        -set_name=DBCredsSYSDBA
        -credential_name=MyDBCredentials
        -credential_owner="Joe"
```

set_properties

Sets the property for a test or beacons.

Format

```
emcli set_properties
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name
      -testtype=<test_type>
      [-beacons=<beacon_names>]
      [-properties='prop1:value1;prop2:value2;..']+

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **testname**

  Name of the test to set the property on.

- **testtype**

  Type of test to set the property on.

- **beacons**

  Names of the beacons to set the property on.

- **properties**

  Names and values of the properties to be set (can be multiple).

Examples

Example 1

The following example sets the property timeout to 30000 and granularity to transaction for the test MyTest defined on MyTarget for all beacons.

```
emcli set_property -name='MyTarget' -type='generic_service'
      -testname='MyTest' -testtype='HTTP'
      -propertyName='timeout:30000;granularity:transaction'
```

Example 2

The following example sets the property value to 30000 of the test MyTest defined on MyTarget for only MyBeacon and MyBeacon2. This only works if the specified properties can be set on a per beacon level.

```
emcli set_property -name='MyTarget' -type='generic_service'
      -testname='MyTest' -testtype='HTTP'
      -bcnName='MyBeacon;MyBeacon2'
      -propertyName='timeout' -propertyValue='30000'
```

set_reverse_ping_interval

Modifies the maximum waiting time for the management Agents. You need to provide Agent names for the modification.

Format

```
emcli set_reverse_ping_interval
        -agent_names="agent1[;agent2...]"|-all_agents
        -value=" "|-reset_to_default
```

[ ] indicates that the parameter is optional

Parameters

- **agent_names**

    Management agents (host:port) on which the modification needs to be performed.

- **all_agents**

    Use only when all Agents need to be modified with the new value.

- **value**

    New value to which the existing waiting time needs to be updated.

- **reset_to_default**

    Use when the value needs to be reset to the default value.

Examples

Example 1
The following example modifies the existing waiting time with the new value provided, which in this case is 240.

```
emcli set_reverse_ping_interval -agent_names="myhost1.us.oracle.com:1838"
-value=240
```

Example 2
The following example modifies the existing waiting time for the provided Agents with the default value in the Ping System.

```
emcli set_reverse_ping_interval -agent_
names="myhost1.us.oracle.com:1838;myhost2.us.oracle.com:4352" -reset_to_default
```

set_standby_agent

Permits targets to relocate from one Management Agent to another. This verb always populates a table that determines which targets from the source Management Agent to the destination Management Agent are permitted to relocate for the Enterprise Manager target.

Format

```
emcli set_standby_agent
      -src_agent=<source_agent>
      -dest_agent=<destination_agent>
      -target_name=<target_name>
      -target_type=<target_type>

[ ] indicates that the parameter is optional
```

Parameters

- **src_agent**

  Management Agent currently monitoring the targets. If srcAgent is not known, enter currentOwner as the argument.

- **dest_agent**

  Management Agent for which you want to monitor the targets.

- **target_name**

  Name of the target to be moved.

- **target_type**

  Type of target to be moved.

Output

Output message of the command execution.

set_target_property_value

Sets the value of a target property for a specified target. Any prior values of the target property are overwritten. When assigning values to the Oracle-provided target properties, use the English names of these target properties:

Comment, Deployment Type, Line of Business, Location, Contact

> **Note:** You can only set up and propagate one property at a time to members.

Format

```
emcli set_target_property_value
      -property_records="target_name:target_type:property_name:property_value"
      [-separator=property_records="sep_string"]
      [-subseparator=property_records="subsep_string"]
      [-input_file="parameter_tag:file_path"]
      [-propagate_to_members]

[ ] indicates that the parameter is optional
```

Parameters

- **property_records**

  List of property records. The following parts comprise each property record:

  <target_name>:<target_type>:<property_name>:property_value>

  – target_name — Target name of the target for which you want to update the property.

  – target_type — Target type of the target.

  – property_name — Name of the property whose value you want to update. Property names are case sensitive.

  – property_value — Value to be assigned/updated for the property.

- **separator**

  When specifying multiple property records, use the separator string delimiter as a delimiter between property records. The default separator delimiter is ";".

- **subseparator**

  String delimiter to be used between parts of a property record. The default subseparator delimiter is ":".

- **input_file**

  Used in conjunction with the -property_records option, this option enables you to provide the property records in a file. This option specifies a mapping between a tag and a local file path. The tag is specified in lieu of property records. The tag cannot contain colons (:) or semi-colons (;) .

- **propagate_to_members**

  Used for group and system targets to also propagate the property to all of its members.

Examples

Example 1

The following example sets the 'Owner Name' property to Jane Smith for the database test_database.

```
emcli set_target_property_value
      -property_records="test_database:oracle_database:Owner Name:Jane Smith"
```

Example 2

The following example sets the Owner property to Jane Smith for the database test_db, and also sets the Asset Number property to 100 for the database test_db1.

```
emcli set_target_property_value
        -property_records="test_db:oracle_database:Owner:Jane Smith;
         test_db1:oracle_database:Asset Number:100"
```

Example 3

The following example takes the input of the property records from the specified file /temp/rec_file.

```
emcli set_target_property_value
        -property_records="REC_FILE" -input_file="REC_FILE:/temp/rec_file"
```

The file /temp/rec_file would contain entries such as:

```
test_db:oracle_database:Owner:Jane Smith;test_db1:oracle_database:Asset Number:100
```

Example 4

The following example sets the Owner property to Jane Smith for the test_db database, and sets the Asset Number property to 100 for the test_db1 database. The separator used within the records is "," and the subseparator is "@" .

```
emcli set_target_property_value
        -property_records="test_db@oracle_database@Owner@
         Jane Smith,test_db1@oracle_database@AssetNumber@100"
```

Example 5

```
emcli set_target_property_value -property_records="MyProdGroup:composite:LifeCycle
Status:Production" -propagate_to_members
```

set_test_threshold

Sets a test threshold.

Format

```
emcli set_test_threshold
      -name=<target_name>
      -type=<target_type>
      -testname=<test_name>
      -testtype=<test_type>
      -metricName=<metric_name>
      -metricColumn=<metric_column>
      -occurrences=<occurrences>
      [-warningThres=<warning_threshold>]
      [-criticalThres=<critical_threshold>]
      [-operator=<operator>]
      [-beaconName=<beacon_name>]
      [-stepName=<step_name>]
      [-stepGroupName=<stepgroup_name>]

[ ] indicates that the parameter is optional
```

Examples

```
emcli set_test_threshold -name="Service Name"
          -type="generic_service"
          -testname="Test Name"
          -testtype="HTTP"
          -metricName="http_response"
          -metricColumn="timing"
          -occurrences=1
          -warningThres=100000
```

setup

Configures EM CLI to work with a specific management server.

You can set up the EM CLI client either in secure mode by specifying the -noautologin option, or unsecure mode by specifying the -autologin otpion. -noautologin is the default, so if you do not specify either option, the EM CLI client is automatically set up in secure mode.

The configuration directory will contain log files generated by EM CLI to record informational and error messages generated during operations. For more details, refer to Section 1.3.2.4, "Using EM CLI Log Files" on page 1-6.

Format

```
emcli setup
        -url="http[s]://host:port/em"
        -username=<EM_console_username]
        [-password=<password_of_user>]
        -dir=<local_emcli_config_directory>
        [-localdirans=yes|no]
        [-licans=yes|no]
        [-trustall]
        [-certans=yes|no]
        [-nocertvalidate]
        [-novalidate]
        [-autologin]
        [-noautologin]
        [-noregister]
        [-custom_attrib_file=<custom_attr_file_path>]

[ ] indicates that the parameter is optional
```

Parameters

- **url**

  URL of the Oracle Management Server (OMS). host specifies the host of the OMS. port specifies the listening port of the OMS. Both http and https protocols are supported. (https is recommended for security reasons).

- **username**

  Enterprise Manager user name to be used by all subsequent EM CLI commands when contacting the OMS.

  If the SSO user is also an Enterprise Manager user (that is, authenticated in LDAP/OID), you can only register EM CLI with the ssousername. After you enable SSO for the OMS, you cannot subsequently register EM CLI with only username.

- **password**

  Enterprise Manager user password. If you do not specify this option, you are prompted for the password interactively.

  > **Note:** Providing a password on the command line is insecure and should be avoided.

- **dir**

  Directory where an EM CLI configuration directory will be created. This directory must be on a locally mounted file system. A warning and confirmation is issued for an HTTPS URL if the directory is not heuristically identified as such (unless you specify `trustall`). The directory can be relative to the working directory where setup is called, or it can be absolute. This option defaults to the user's home directory.

- **localdirans**

  Indicates whether the setup directory given with the -dir option is a local directory. Specify yes to indicate that the setup directory is local, and specify no to indicate that the setup directory is non-local.

- **licans**

  Indicates whether the license is accepted or not accepted by the user. Specify yes to accept the license, or specify no to not accept the license.

- **trustall**

  Automatically accepts any server certificate from the OMS, which results in lower security.

- **certans**

  Indicates whether the certificate needs to be trusted without having to prompt the user. Specify yes to trust the certificate, and specify no to not trust the certificate.

- **nocertvalidate**

  Does not validate the host name in the SSL certificate provided by the OMS.

- **novalidate**

  Does not authenticate the Enterprise Manager user name or SSO user name against the OMS. Assume the given user name is valid. This enables the configuration to be stored (Enterprise Manager URL and user) without validating or connecting to Enterprise Manager. This might be useful in scenarios where Enterprise Manager is not up when you do run the setup command.

- **autologin**

  In this mode, credentials are stored on the EM CLI client system. Autologin mode is preserved until emcli logout is executed. If the session has expired when a verb is executed, login is automatically performed and the verb is executed.

  Verbs executed after emcli logout may fail with the message "Error: Session expired. Run emcli login to establish a session." You need to run the login verb to log in to EM CLI after an emcli logout. After the Enterprise manager user's password has changed, you need to log in with the ID and the new password. The new password will subsequently be stored.

  Note that noautologin is the default mode.

- **noautologin**

  In this default mode, credentials are not stored on the EM CLI client system. If the session has expired when a verb is executed, you have to explicitly run the login verb and then run the required verb.

- **noregister**

  Does not register this EM CLI instance.

- **custom_attrib_file**

  Path name of a file containing Audit Custom Attribute values. This option is required when the OMS is configured for Audit Custom Attributes. If you do not provide custom_attrib_file, you are prompted to enter the values of the custom attributes.

  The file can contain up to three lines, each containing the description of one custom attribute. Each line should be of the form:

  ```
  <attr-name>#<attr-displayname>#<isMandatory>#<attr-value>
  ```

  - **#** — Field separator.
  - **attr-name** — Name of the attribute.
  - **attr-displayname** — Display name of the attribute.
  - **isMandatory** — 1 if the attribute is mandatory, otherwise 0.
  - **attr-value** — Value of the custom attribute.

Examples

```
emcli setup -url=http://myworkstation.example.com:7770/em -username=sysman
```

To configure the EM CLI Client to function with multiple OMSes by implementing multiple setups, do the following:

1. Set up the EM CLI client for OMS1 at location dir1:

   ```
   emcli setup -dir=<dir1> -url=<Url of OMS1> -user=<EM Username for OMS1>
   ```

2. Set up the EM CLI client for OMS2 at location dir2:

   ```
   emcli setup -dir=<dir2> -url=<Url of OMS1> -user=<EM Username for OMS2>
   ```

3. Set the environment variable EMCLI_STATE_DIR to point to the setup directory for OMS1:

   ```
   setenv EMCLI_STATE_DIR <dir1>
   ```

   This sets the EM CLI Client to function with OMS1.

4. Set the environment variable EMCLI_STATE_DIR to point to the setup directory for OMS2:

   ```
   setenv EMCLI_STATE_DIR <dir2>
   ```

   This sets the EM CLI Client to function with OMS2.

setup_bipublisher

Sets up a relationship between Enterprise Manager and a BI Publisher Web Application. If a relationship already exists, you must provide the -force option. The Enterprise Manager System Reports are deployed to the newly configured BI Publisher Web Application. To just change the registration details without deploying the reports, use the -nodeploy option. Detailed status messages are provided for all operations.

Use the -force option to overwrite existing copies of reports if they exist. If you do not want to deploy following setup, you can specify the -nodeploy option.

Format

```
emcli setup_bipublisher
    [-force]
    -protocol=http|https
    -host=<hostname>
    -port=<portnumber>
    -uri=xmlpserver
    [-nodeploy]

[ ] indicates that the parameter is optional
```

Parameters

- **force**

  Overwrites existing copies of reports if they exist. The following scenarios are affected by this option:

  – If a relationship exists between Enterprise Manager and a BI Publisher Web Application, this option overrides it with a new relationship (-host, -port, and so forth).

  – If you do not specify -nodeploy, this option causes deployed reports to overwrite any that may already exist for the BI Publisher Web Application in the "Enterprise Manager Cloud Control" folder.

- **protocol**

  Must be either http or https.

- **host**

  Name of the host that is running, or server load balancer fronting the BI Publisher Web Application.

- **port**

  Port number of the web service.

- **uri**

  Web application context root, which must be xmlpserver.

- **nodeploy**

  Specify if you do not want to deploy following setup. Suppresses the Enterprise Manager BI Publisher System Reports to the BI Publisher Web Application. You can do this later with the deploy_bipublisher_reports cli.

Examples

Example 1

```
emcli setup_bipublisher
     -protocol=https
     -host=www.somehost.com
     -port=7801
     -uri=xmlpserver
```

Example 2

The following example reconfigures the BI Publisher Managed Server (BIP) inside the WebLogic Server console to listen on a different port (9704):

```
emcli setup_bipublisher -protocol=https -host=somehost.com -port=9704
-uri=xmlpserver -force -nodeploy
```

Example 3

The following example sets up BI Publisher behind a load balancer with a host name of slb.somedomain.com on port 9754:

```
emcli setup_bipublisher -proto=https -host=slb.somedomain.com -port=9754
-uri=xmlpserver -force -nodeploy
```

show_audit_settings

Shows the following details of the current audit settings:

— Audit Switch
— Externalization Switch
— Directory
— File Prefix
— File Size
— Data Retention Period

Format

```
emcli show_audit_settings
      -view="SUMMARY|DETAIL"
```

show_credential_set_info

Displays the parameters of credential sets defined with target types.

Format

```
emcli show_credential_set_info
      [-target_type="<target_type>"]
      [-set_name="<credential_set_name>"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target. The default is to display the credential set defined for all target types.

- **set_name**

  Name of the credential set. The default is to display all credential sets defined for a target type.

Examples

Example 1
The following example displays the details of all credential sets defined with all target types:

```
emcli show_credential_set_info
```

Example 2
The following example displays all credential sets defined with the oracle_database target type:

```
emcli show_credential_set_info -target_type=oracle_database
```

Example 3
The following example displays the details of the HostUDMCreds credential set defined for the host target type.

```
emcli show_credential_set_info -target_type=host
     -set_name=HostUDMCreds
```

show_credential_type_info

Displays the parameters of credential types defined for target types.

Format

```
emcli show_credential_type_info
      [-target_type="<target_type>"]
      [-type_name="<credential_type_name>"]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target. The default is to display the credential set defined for all target types.

- **type_name**

  Name of the credential type. The default is to display all credential types defined for a target type.

Examples

Example 1
The following example displays the details of all credential types defined with all target types:

```
emcli show_credential_type_info
```

Example 2
The following example displays all credential types defined with the oracle_database target type:

```
emcli show_credential_type_info -target_type=oracle_database
```

Example 3
The following example displays the details of the HostUDMCreds credential type defined for the oracle_database target type.

```
emcli show_credential_type_info -target_type=oracle_database
      -type_name=HostUDMCreds
```

show_operations_list

Shows the list of all auditable Enterprise Manager operations names.

Format

```
emcli show_operations_list
```

Output

Output appears as shown in the following example:

```
ADD_AGENT_REGISTRATION_PASSWORD
AGENT_REGISTRATION_PASSWORD_USAGE
AGENT_RESYNC
APPLY_TEMPLATE
AUDIT_EXPORT_SETTINGS
AUDIT_SETTINGS
CHANGE_PASSWORD
CHANGE_PREFERRED_CREDENTIAL
CREATE_PG_SCHED
CREATE_ROLE
CREATE_TEMPLATE
CREATE_UDP
CREATE_UDPG
CREATE_USER
DELETE_AGENT_REGISTRATION_PASSWORD
DELETE_JOB
DELETE_PG_EVAL
DELETE_PG_SCHED
DELETE_ROLE
DELETE_TEMPLATE
DELETE_UDP
DELETE_UDPG
DELETE_USER
EDIT_AGENT_REGISTRATION_PASSWORD
EDIT_JOB
EDIT_PG_SCHED
EDIT_TEMPLATE
EDIT_UDP
EDIT_UDPG
EVALUATE_UDP
FILE_TRANSFER
GET_FILE
GRANT_JOB_PRIVILEGE
GRANT_ROLE
GRANT_SYSTEM_PRIVILEGE
GRANT_TARGET_PRIVILEGE
IMPORT_UDP
JOB_OUTPUT
LOGIN
LOGOUT
MODIFY_METRIC_SETTINGS
MODIFY_POLICY_SETTINGS
MODIFY_ROLE
MODIFY_USER
PUT_FILE
REMOTE_OPERATION_JOB
REMOVE_PRIVILEGE_DELEGATION_SETTING
REPOSITORY_RESYNC
```

```
REVOKE_JOB_PRIVILEGE
REVOKE_ROLE
REVOKE_SYSTEM_PRIVILEGE
REVOKE_TARGET_PRIVILEGE
SAVE_MONITORING_SETTINGS
SET_PRIVILEGE_DELEGATION_SETTING
SUSPEND_JOB
```

show_patch_plan

Shows the details of a particular patch plan.

Format

```
emcli show_patch_plan
        -name="name"
        [-info [-showPrivs]] [-actions [-onlyShowEnabled]]
        [-patches]
        [-targets]
        [-deplOptions]
        [-analysisResults]
        [-conflictFree]
        [-impactedTargets]
        [-deploymentProcedures]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Plan name. If you only provide this parameter with no other options, the full details of the patch plan are shown.

- **info**

  Shows the generic information of the given patch plan.

- **show_Privs**

  Shows the user privileges on the given patch plan along with the generic information.

- **actions**

  Show the actions that are possible to be taken on the given patch plan.

- **onlyShowEnabled**

  Only show the enabled actions on the given patch plan.

- **patches**

  Shows details of the patches contained in the given patch plan.

- **targets**

  Shows details of the targets contained in the given patch plan.

- **deplOptions**

  Shows details of the deployment options contained in the given patch plan.

- **analysisResults**

  Shows details of the analysis results of the given patch plan.

- **conflictFree**

  Shows details of the conflict-free patches of the given patch plan.

- **impactedTargets**

  Shows details of the impacted targets of the given patch plan.

■ **deploymentProcedures**

Shows the deployment procedure of the given patch plan.

Examples

```
emcli show_patch_plan -name="plan name"

emcli show_patch_plan -name="plan name" -info

emcli show_patch_plan -name="plan name" -actions -onlyShowEnabled

emcli show_patch_plan -name="plan name" -info -showPrivs
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
submit_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

start_agent

Starts up a Management Agent. This verb requires operator privilege or full privilege on the Management Agent.

Format

```
emcli start_agent
        -agent_name="agent_target_name"
        [-host_username="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name="credential_name"]
        [-credential_setname="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds".

Examples

Example 1

```
emcli start_agent -agent_name="agent.example.com:1234"
                          -host_username="test_user"
                          -host_pwd="test"
```

Example 2

```
emcli start_agent -agent_name="agent.example.com:1234"
                          -credential_name="MyMachineCredential"
```

Example 3

```
emcli start_agent -agent_name="agent.example.com:1234"
                          -credential_setname="HostCreds"
```

status

Shows whether EM CLI is configured or not, and shows the EM CLI setup details. It also displays the Java home, version, EM CLI home, and all of the EM CLI configuration detials if it is configured.

Format

```
emcli status
```

Parameters

None.

Output

The following example shows output when EM CLI setup has not been done:

```
Oracle Enterprise Manager Cloud Control 12c Release 12.1.0.0.0.
Copyright (c) 1996, 2011 Oracle Corporation and/or its affiliates. All rights
reserved.

Instance Home : /home/sumadas
Status       : Not Configured
```

The following example shows output after EM CLI setup has been done:

```
Oracle Enterprise Manager Cloud Control 12c Release 12.1.0.0.0.
Copyright (c) 1996, 2011 Oracle Corporation and/or its affiliates. All rights
reserved.

Instance Home           : /ade/sumadas_emcli/oracle/work/.emcli
Status                  : Configured
EMCLI Home              : /ade/sumadas_emcli/emcore/emcli/bin
EMCLI Version           : 12.1.0.0.0
Java Home               : /ade_autofs/nfsdo_base/EMGC/MAIN/LINUX/110811/jdk6/jre
Java Version            : 1.6.0_24
Log file                : /ade/sumadas_emcli/oracle/work/.emcli/.emcli.log
EM URL                  : https://dadvma0121.example.com:14487/em
EM user                 : SYSMAN
Auto login              : true
Trust all certificates  : true
```

stop_agent

Shuts down a Management Agent. This verb requires operator privilege or full privilege on the Agent.

Format

```
emcli stop_agent
        -agent_name="agent_target_name"
        [-host_username="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name="credential_name"]
        [-credential_setname="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds".

Examples

Example 1

```
emcli stop_agent -agent_name="agent.example.com:1234"
                        -host_username="test_user"
                        -host_pwd="test"
```

Example 2

```
emcli stop_agent -agent_name="agent.example.com:1234"
                        -credential_name="MyMachineCredential"
```

Example 3

```
emcli stop_agent -agent_name="agent.example.com:1234"
                        -credential_setname="HostCreds"
```

stop_blackout

Stops a blackout.

You can stop a blackout before it has fully started, for example, when it has a "Scheduled" status. You can also stop a blackout while it is in effect.

Format

```
emcli stop_blackout
     -name="name"
     [-createdby="blackout_creator"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the blackout to stop.

- **createdby**

  Enterprise Manager user who created the blackout. The default is the current user. The SUPER_USER privilege is required to stop a blackout created by another user.

Examples

Example 1
The following example stops blackout `backup_db3` created by the current user.

```
emcli stop_blackout -name=backup_db3
```

Example 2
The following example stops blackout `weekly_maint` created by user `joe`. The current user must either be user `joe` or a user with the SUPER_USER privilege.

```
emcli stop_blackout -name=weekly_maint -createdby=joe
```

stop_instance

Stops a scheduled, failed, or running deployment instance.

Format

```
emcli stop_instance
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
```

Parameters

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- **name**

  Name of the execution.

- **owner**

  Owner of the execution.

Examples

```
emcli stop_instance -instance=16B15CB29C3F9E6CE040578C96093F61
```

stop_job

Stops a specified job. You can use the `get_jobs` verb to obtain a list of job IDs and names.

Format

```
emcli stop_job
      -job_id="jobID"|-name="jobName"
```

Parameters

- **job_id**

  Job ID to identify the job to stop.

- **name**

  Name of the job to stop. To uniquely identify the job, the current administrator is used.

Examples

Example 1

The following example stops a job with the specified ID.

```
emcli stop_job -job_id=123456789012345678901234567890012
```

Example 2

The following example stops a job named `Backup_Wednesday`, which is owned by the current Enterprise Manager administrator and scheduled to execute in the future.

```
emcli stop_job -name=Backup_Wednesday
```

submit_agent_patch

Patches the Management Agent. All of the inputs should be present in the targets_file.xml file.

Format

```
emcli submit_agent_patch
-input_file="data:targets_file.xml"
-schedule="start_time:<value>;tz:<value>;grace_period:<value>"
```

Parameters

- **input_file**

  XML file name containing the necessary input.

- **schedule**

  Specify the starting time, time zone, and grace period.

Examples

```
emcli submit_agent_patch -input_file="data:targets_file.xml"
-schedule="start_time:2011/8/21 21:23;tz:America/New_York;grace_period:15"
```

submit_masking_job

Submits a masking job and returns the display job ID and execution ID.

Format

```
emcli submit_masking_job
    -definition_name=<masking_defn_name>
    -target_name=<database_target_name>
    [-encryption_key=<encryption_key_string]
    [-host_preferred_creds=<preferred_credentials_name>]
    [-host_cred_name=<credential_name>]
    [-db_preferred_creds=<preferred_credentials_name>]
    [-db_cred_name=<credential_name>]
    [-parameters=name1:value1;name2:value2;...]
    [-script_file_location=<script_file_location>]
    [-script_file_name=<script_file_name>]
    [-input_file=PWD_FILE_TAG:<credentials_file_name>]
    [-script | -format=[name:<pretty|script|csv>];
                        [column_separator:"column_sep_string"];
                        [row_separator:"row_sep_string"];
    ]

[ ] indicates that the parameter is optional
```

Parameters

- **definition_name**

  Masking definition name.

- **target_name**

  Database target name to mask.

- **encryption_key**

  Specify an encryption key if the masking definition involves usage of a substitute format.

- **host_preferred_creds**

  Type of preferred credentials to use to connect to the database host, which can either be HostCredsNormal or HostCredsPriv.

- **host_cred_name**

  Credential name to use to connect to the database host.

- **db_preferred_creds**

  Type of preferred credentials to use to connect to the database instance, which can either be DBCredsNormal or DBCredsSYSDBA.

- **db_cred_name**

  Credential name to use to connect to the database instance.

- **parameters**

  List of name-value pairs that represent the credentials required to connect to the database instance. The supported parameters are 'db_username', 'db_password', 'db_role', 'db_cred_name', 'host_username', 'host_password', and 'host_cred_name'. If PDP needs to be used, additional parameters to be specified are

'PDP','RUNAS', and 'PROFILE'. The 'PROFILE' option is only applicable for
Powerbroker.

- **script_file_location**

    Location where the SQL script is to be copied and executed. Default values of
    $ORACLE_HOME/dbs are used if a value is not specified.

- **script_file_name**

    Name of the script file to store the masking SQL script. If you do not specify a
    name, a system-generated file name is used.

- **input_file**

    Used in conjunction with the 'parameters' option, this option enables you to store
    parameter values, such as user name and password, in a separate file. The
    'input_file' option specifies a mapping between a tag and a local file path. The tag
    is specified in lieu of specific parameter values of the 'parameters' option. You can
    specify multiple -input_file parameters. The result would be a combination of all
    of the files.

- **script**

    This option is equivalent to -format="name:script" .

- **format**

    Format specification (default is `-format="name:pretty"`).

    - `format="name:pretty"` prints the output table in a readable format not
      intended to be parsed by scripts.

    - `format="name:script"` sets the default column separator to a tab and the
      default row separator to a newline. The column and row separator strings can
      be specified to change these defaults.

    - `format="name:csv"` sets the column separator to a comma and the row
      separator to a newline.

    - `format=column_separator:"column_sep_string"` column-separates the verb
      output by <column_sep_string>. Rows are separated by the newline character.

    - `row_separator:"row_sep_string"` row-separates the verb output by
      <row_sep_string>. Rows are separated by the tab character.

Examples

Example 1
The following example submits a masking job for the definition name email1 and
returns the job ID and execution ID:

```
emcli submit_masking_job -definition_name=email1 -parameters="db_username:sys;
    db_password:password;db_role:SYSDBA;host_username:test;host_password:password"
```

Example 2
The following example assumes the default credential set as DBCredsNormal and
returns job ID and execution ID.

```
emcli submit_masking_job -definition_name=email1
```

Example 3

The following example picks up credentials from the files host_creds.txt and db_creds.txt.

```
emcli submit_masking_job -definition_name=email1 -parameters="HOST_CREDS;DB_CREDS"
-input_file=HOST_CREDS:host_creds.txt -input_file=DB_CREDS:db_creds.txt
```

It is also possible to specify both of the credentials in one file and use only one -input_file tag. If PDP must be used, you need to provide values in the parameters/input_file as follows:

- SUDO:

  ```
  db_username:sys;db_password:password;db_role:SYSDBA;host_username:user2;host_
  password:password;PDP:SUDO;RUNAS:user1
  ```

- POWERBROKER:

  ```
  db_username:sys;db_password:password;db_role:SYSDBA;host_username:user2;host_
  password:password;PDP:POWERBROKER;RUNAS:user1;PROFILE:profile
  ```

Example 4

The following example uses the named database credential DB_NC, named host credential HOST_NC, and submits the masking job. If the masking definition involves usage of the substitute format, uses the encryption key as 'abcd'. Overrides the default script file name and location by the values specified. Submits a masking job for the given definition name and returns job id and execution id.

```
emcli submit_masking_job -definition_name=email2 -target_name=testdb  -db_cred_
name=DB_NC -host_cred_name=HOST_NC -encryption_key=abcd -script_file_location=/tmp
-script_file_name=email1.sql
```

submit_operation_plan

Submits the specified operation plan for execution.

Format

```
emcli submit_operation_plan
      -name=<operation_plan_name>
      [-run_prechecks=true|false]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the operation plan.

- **run_prechecks**

  Optionally run pre-checks by specifying either true or false.

Examples

```
emcli submit_operation_plan
      -name="austin-switchover"
      -run_prechecks="true"
```

See Also

emcli create_operation_plan

emcli get_operation_plans

submit_patch_plan

Submits action on a given patch plan, such as analyzing, preparing, deploying, and switchbacking, or finds the next action automatically, then runs it.

Format

```
emcli submit_patch_plan
        -name="name"
        -action="action name"
```

Parameters

- **name**

  Patch plan name.

- **action**

  Action to submit on the given patch plan.

Examples

```
emcli submit_patch_plan -name="plan name"
```

```
emcli submit_patch_plan -name="plan name" -action="analyze"
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan
upload_patches

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

submit_procedure

Submits a deployment procedure or a pre-saved procedure configuration.

Format

```
emcli submit_procedure
      -input_file=data:"file_path"
      [-procedure="procedure_guid"]
      [-name="procedure_name"]
      [-owner="procedure_owner"]
      [-parent_proc="procedure_of_procedure_config"]
      [-instance_name="procedure_instance_name"]
      [-grants="users_and_their_corresponding_access_levels"]
      [-schedule=
          start_time:yyyy/MM/dd HH:mm;
          tz:{java timezone ID};
          grace_period:xxx;
      ]
```

`[ ]` indicates that the parameter is optional

Parameters

- **input_file**

  Input data for the Deployment Procedure. The `file_path` should point to a file containing the data properties file.

- **procedure**

  GUID of the procedure to execute.

- **name**

  Name of the procedure or procedure configuration.

- **owner**

  Owner of the procedure or procedure configuration.

- **parent_proc**

  Procedure of the procedure configuration, this applies to a procedure configuration when there is both a procedure and a procedure configuration with the same name.

- **instance_name**

  Name of the procedure instance.

- **grants**

  Users and their corresponding access levels designated as a string of user:privilege pairs each separated by ; .

  where:

  user = Enterprise Manager user name

  privilege = VIEW_JOB or FULL_JOB

- **schedule**

Schedule for the deployment procedure. If not specified, the procedure is executed immediately.

`start_time` — When the procedure should start

`tz` — Optional time zone ID

Output Columns

Instance GUID

Examples

```
emcli submit_procedure -input_file=data:data.properties
-procedure=16B15CB29C3F9E6CE040578C96093F61 -grants="user1:VIEW_JOB;user2:
FULL_JOB" -schedule="start_time:2006/6/21 21:23;tz:America/New_York;
grace_period:60" -instance_name="MyProcedureInstance_0001"
```

subscribeto_rule

Subscribes the user to a rule with email notification.

It is not an error to specify email addresses that are already in the assignto user's preferences.

A message appears if the outgoing mail server (SMTP) has not been set up. When you specify the -fail_if_no_mail_server, this condition is an error and prevents the subscribe from occurring; otherwise, this condition is a warning that does not affect the success of this command.

Format

```
emcli subscribeto_rule
        -ruleset_name="ruleset_name"
        -rule_name="rule_name"
        -owner="rule_owner"
        [-assignto="em_username"]
        [-email="email_address";...]
        [-fail_if_no_mail_server]
```

[ ] indicates that the parameter is optional

Parameters

- **ruleset_name**

  Name of the incident rule set.

- **rule_name**

  Name of the rule.

- **owner**

  Owner of the rule set.

- **assignto**

  User to subscribe to the notification rule. If the assignto user is not the current user, or if the owner of the rule is not the current user, the super-user privilege is needed. The default is the current user.

- **email**

  List of email addresses to associate with the rule to which the assignto user is being subscribed. These addresses are first added to the preferences of the assignto user (duplicates are ignored) before being assigned to the notification rule. The email addresses are added only if the current user has the privilege to subscribe the assignto user to the rule.

- **fail_if_no_mail_server**

  A message appears if the outgoing mail server (SMTP) has not been set up. When you specify the -fail_if_no_mail_server option, this condition is an error and prevents the subscribe from occurring; otherwise, this condition is a warning that does not affect the success of this command.

Examples

Example 1

The following example subscribes the current user to the rule `"Agent Upload Problems"` using the current user's email addresses for notification. The current user must have the `SUPER_USER` (or have `sysman`) privilege for this to succeed, since `sysman` owns the rule. Also, the current user must already have at least one email address in his/her preferences for this command to succeed.

```
emcli subscribeto_rule -name="Agent Upload Problems" -owner=sysman
```

Example 2

The following example first adds the two specified email addresses to the preferences for user `joe`. Then user `joe` is subscribed to the rule `"Agent Upload Problems"` using joe's email addresses for notification. The current user must have `SUPER_USER` privilege (or be `joe`) for this command to succeed.

```
emcli subscribeto_rule -name="Agent Upload Problems" -owner=sysma
      -assignto=joe -email="joe@work.com;joe@home.com"
```

suspend_instance

Suspends a running deployment instance.

Format

```
emcli suspend_instance
      -instance=<instance_guid>
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]

[ ] indicates that the parameter is optional
```

Parameters

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- **name**

  Name of the execution.

- **owner**

  Owner of the execution.

Examples

```
emcli suspend_instance -instance=16B15CB29C3F9E6CE040578C96093F61
```

sync

Synchronizes the EM CLI client with an OMS. After synchronization, all verbs and associated command-line help available to this OMS become available at the EM CLI client.

Synchronization occurs automatically during a call to setup.

Format

```
emcli sync
```

Parameters

None.

sync_alerts

Synchronizes all alerts for the specified target between the Agent and the repository. You typically use this command when you think that the Agent has not uploaded the latest alert to the repository, and the repository is therefore out of sync with the Agent state.

To determine if alerts are out of sync between the Agent and the repository for the specified target, run the get_unsync_alerts command.

Format

```
emcli sync_alerts
        -target_type=type
        -target_name=name
        -agent_name=agent
```

Parameters

- **target_type**

  Internal target-type identifier (host, oracle_database, emrep, and so forth).

- **target_name**

  Name of the target.

- **agent_name**

  Name of the Agent.

Examples

Example 1

The following example synchronizes alert states for target_type "host" and target_name "hostname.oracle.com".

```
emcli sync_alerts -target_type=host -target_name=hostname.oracle.com
```

Example 2

The following example synchronizes alert states for all targets that the Agent "hostname.xyz.com:port" monitors.

```
emcli sync_alerts -agent_name=hostname.xyz.com:port
```

sync_beacon

Synchronizes a beacon that is monitoring the target (reloads all collections to the beacon).

Format

```
emcli sync_beacon
      -name=target name
      -type=target type
      -bcnName=beacon name
```

Parameters

- **name**

  Service target name.

- **type**

  Service target type.

- **bcnName**

  Beacon name to synchronize.

Examples

The following example synchronizes `MyBeacon`, which is monitoring the `MyTarget` target of type `generic_service`.

```
emcli sync_beacon -name='MyTarget' -type='generic_service'
      -bcnName='MyBeacon'
```

test_named_credential

Tests the named credentials provided in the list. Instance credentials are tested against the credential target. Global credentials are tested against the target provided.

Format

```
emcli test_named_credential
      -cred_names=<cred_name_list>
      [-target_name=<target_name>]
      [-target_type=<target_type>]
```

Parameters

- **cred_names**

  List of credential names to be tested.

- **target_name**

  Target name to test the global credentials. Instance credentials are tested against their respective targets.

- **target_type**

  Target type to test the global credentials.

Examples

Example 1

The following example tests the instance named credentials NC1 owned by the current logged in user and NC2 owned by ADMIN1.

```
emcli test_named_credential
      -cred_names="NC1;NC2:ADMIN1"
```

Example 2

The following example tests the global host named credentials NC1, NC2, and NC3 against the target testhost.us.oracle.com.

```
emcli test_named_credential
      -cred_names="NC1;NC2;NC3"
      -target_name="testhost.us.oracle.com"
      -target_type="host
```

trace

Enables or disables tracing for OMS.

Format

```
emcli trace
       -enable="true|false"
       -user="username"
```

Parameters

- **enable**

  Specify true to enable and false to disable.

- **user**

  Name of the user.

Example

The following example enables tracing for user sysman.

```
emcli trace -enable=true -user=sysman
```

udmmig_list_matches

Lists all the metric extensions that match the UDMs in a given migration session.

Format

```
emcli udmmig_list_matches
      -session_id=<sessionId>
```

Parameters

- **session_id**

    Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

udmmig_request_udmdelete

Deletes the UDMs that have been replaced by Metric Extensions.

Format

```
emcli udmmig_request_udmdelete
      -session_id=<sessionId>
      -input_file=metric_tasks:<complete_path_to_file>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Specify a file name that contains a target, UDM, one per line in the following format:

  ```
  <targetType>,<targetName>,<collection name>
  ```

Example

The following example attempts to delete the UDM from all relevant targets. This step is indicative of the end of the migration process. The file input_tasks lists the locations where the UDM is present.

```
emcli udmmig_request_udmdelete -session_id=<sessionId> -input_file=metric_
tasks:input_tasks
```

udmmig_retry_deploys

Retries the deployment of metric extensions to a target.

Format

```
emcli udmmig_retry_deploys
      -session_id=<sessionId>
      -input_file=metric_tasks:<complete path to file>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Specify a file name that contains a target, UDM, one per line in the following format:

  ```
  <targetType>,<targetName>,<collection name>
  ```

Example

The following example attempts to deploy the metric extension to all targets where the UDM was present. The file input_tasks lists these locations.

```
emcli udmmig_retry_deploys -session_id=<sessionId> -input_file=metric_tasks:input_
tasks
```

udmmig_session_details

Provides details of the specified migration session, including the targets, templates, UDMs, and metric extensions involved.

Format

```
emcli udmmig_session_details
        -session_id=<sessionId>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

udmmig_submit_metricpicks

Supply the metric picks to use to replace UDMs per target in a session.

Format

```
emcli udmmig_submit_metricpicks
      -session_id=<sessionId>
      -input_file=metric_picks:<complete_path_to_file>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Specify a file name that contains a target, UDM, metric pick, one per line in the following format:

  ```
  <targetType>,<targetName>,<collection name>,[N/E],<metric>,<column>
  ```

  Use N if a new metric should be created, or E if an existing metric is referenced.

Example

The following example provides the mapping from UDM to the metric extension through the file input_picks.

```
emcli udmmig_submit_metricpicks -session_id=<sessionId> -input_file=metric_
picks:input_picks
```

udmmig_summary

Displays all the active migration sessions in the system.

Format

```
emcli udmmig_summary
      [-showAll]

[ ] indicates that the parameter is optional
```

Parameters

- **showAll**

  Prints out all sessions including those that are complete. By default, only in-progress sessions are listed.

udmmig_update_incrules

Updates incident rules that reference UDMs with a reference to replacing a metric extension.

Format

```
emcli udmmig_update_incrules
        -session_id=<sessionId>
        -input_file=udm_inc_rules:<complete_path_to_file>
```

Parameters

- **session_id**

  Specify the ID that was returned when the session was created, or from the output of udmmig_summary.

- **input_file**

  Specify a file name that contains a rule, UDM, metric, one per line in the following format:

  ```
  <ruleset id>,<rule id>,<udm name>,<metric name>
  ```

undeploy_diagchecks

Undeploys diagcheck scripts for targets.

Format

```
emcli undeploy_diagchecks
      {-target_name=<target_name_to_be_updated>
       -target_type=<target_type_to_be_updated> }
      | {-input_file=targetList:<complete_path_to_file>};
```

Parameters

- **target_name**

  Name of the target to be updated.

- **target_type**

  Type of target to be updated.

- **input_file**

  Specify a file name that contains a list of targets, one per line in the following format:

  ```
  <targetType>:<targetName>
  ```

undeploy_plugin_from_agent

Undeploys an Enterprise Manager plug-in from the Management Agents. Undeploying a plug-in from a Management Agent removes all targets of any type belonging to this plug-in from Enterprise Manager.

Defaults to the version currently deployed on the given Management Agent.

Format

```
emcli undeploy_plugin_from_agent
      -plugin="pluginId[:pluginVersion]"
      -agent_names="agent1;agent2"
```

Parameters

- **plugin**

  Plug-in ID and version to be undeployed. Version is optional, and it defaults to the latest version deployed on the management server.

- **agent_names**

  Management Agents (host:port) from which the plug-in is to be undeployed.

Examples

Example 1
The following example undeploys the oracle.sysman.db2 plug-in of version 11.2.0.1.0 from Management Agents myhost1.example.com:1159 and myhost2.example.com:1159.

```
undeploy_plugin_from_agent -plugin=oracle.sysman.db2:11.2.0.1.0
-agent_names="myhost1.example.com:1159;myhost2.example.com:1159"
```

Example 2
The following example undeploys the oracle.sysman.db2 plug-in of the latest version from the Agent myhost1.example.com:1159.

```
undeploy_plugin_from_agent -plugin=oracle.sysman.db2
-agent_names="myhost1.example.com:1159"
```

undeploy_plugin_from_server

Undeploys a plug-in from the Oracle Management Server.

> **Note:** You need to undeploy the plug-in from all Management
> Agents before you can undeploy it from the management server.

Format

```
emcli undeploy_plugin_from_server
        -plugin="plug-inId"[:"pluginVersion"]
        [-sys_password="sys_password"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **plugin**

  This is of the form -plugin=<oracle.sysman.db:12.1.0.1.0> where the plug-in id
  (like oracle.sysman.db) is a required parameter and the version is optional.

  You do not need to provide a version in the -plugin="plugin_id" field, because at
  any given time, only one version of the plug-in can be deployed on the
  management server. Therefore, the version is implicit. Contrast this with
  providing a version during deployment, because you could have downloaded
  more than one version.

- **sys_password**

  The repository sys user password. If not provided at the console, it will be
  prompted for.

Examples

Example 1
The following example undeploys the "oracle.sysman.db2" plug-in from the Oracle
Management Server.

```
undeploy_plugin_from_server -plugin="oracle.sysman.db2" -sys_password=knl_test7
```

Example 2
The following example prompts you for sys_password.

```
emcli undeploy_plugin_from_server -plugin="oracle.sysman.db2"
```

unregister_bipublisher

Unregisters a previously set up relationship between Enterprise Manager and a previously set up relationship (using setup_bipublisher). You can also use this verb to determine the status of the relationship between Enterprise Manager and BI Publisher if you do not specify the -force option.

Format

```
emcli unregister_bipublisher
    [-force]
```

Parameters

- **force**

  This option severs the relationship. The BI Publisher managed server is not stopped or uninstalled.

Examples

Example 1

```
emcli unregister_bipublisher

Error: The BI Publisher Web Application named
"https://somehost.somedomain.com:9704/xmlpserver" is registered. Use -force option
to overwrite this.
```

Example 2

```
emcli unregister_bipublisher -force

BI Publisher "https://somehost.somedomain.com:9704/xmlpserver" has been
unregistered for use with Enterprise Manager.
```

unsecure_agent

Unsecures a secured Management Agent. This verb requires operator privilege or full privilege on the Management Agent.

Format

```
emcli unsecure_agent
        -agent_name="agent_target_name"
        [-host_username ="agent_host_username" -host_pwd="agent_host_password"]
        [-credential_name ="credential_name"]
        [-credential_setname ="credential_setname_of_agent"]

[ ] indicates that the parameter is optional
```

Parameters

- **agent_name**

  Name of the Management Agent target.

- **host_username**

  User name of the OS user (on the host) who owns the Management Agent.

- **host_pwd**

  Password of the OS user (on the host) who owns the Management Agent.

- **credential_name**

  Name of the saved credential.

- **credential_setname**

  Name of the credential set of the Management Agent. Example: "HostCreds"

Examples

Example 1

```
emcli unsecure_agent -agent_name="agent.example.com:1234"
                            -host_username="test_user"
                            -host_pwd="test"
```

Example 2

```
emcli unsecure_agent -agent_name="agent.example.com:1234"
                            -credential_name="MyMachineCredential"
```

Example 3

```
emcli unsecure_agent -agent_name="agent.example.com:1234"
                            -credential_setname="HostCreds"
```

update_and_retry_step

Updates arguments of the failed step and retries it.

Format

```
emcli update_and_retry_step
      -stateguid=<state_guid>
      [-instance=<instance_guid>]
      [-exec=<execution_guid>]
      [-name=<execution_name>]
      [-owner=<execution_owner>]
      [-args="command1:value1;command2:value2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **stateguid**

  State GUID.

- **instance**

  GUID of the instance.

- **exec**

  GUID of the execution.

- **name**

  Name of the execution.

- **owner**

  Owner of the execution.

- **args**

  Arguments of the step to be updated during retry. The format of the arguments are name-value pairs. Name and value are separated by a colon (:), and each pair is separated by a semicolon (;). The arguments take scalar data and list data. The format of list data should be like [a,b,c].

  For the full list of arguments that can be updated, see the get_retry_arguments verb.

Examples

```
emcli update_and_retry_step -instance=16B15CB29C3F9E6CE040578C96093F61
-stateguid=51F762417C4943DEE040578C4E087168 -args="command:ls"
```

update_audit_settings

Updates the current audit settings in the repository and restarts the OMS.

Format

```
emcli update_audit_settings
      -audit_switch="ENABLE|DISABLE"
      -operations_to_enable="name_of_operations_to_enable"
      -operations_to_disable="name_of_operations_to_disable"
      -externalization_switch="ENABLE|DISABLE"
      -directory="directory_name"
      -file_prefix="file_prefix"
      -file_size="file_size"
      -data_retention_period="data_retention_period"
```

Parameters

- **audit_switch**

  Audit switch to enable auditing across Enterprise Manager.

- **operations_to_enable**

  Enables auditing for specified operations. To enable all operations, specify ALL. This parameter is invalid if auditing is disabled.

- **operations_to_disable**

  Disables auditing for specified operations. To disable all operations, specify ALL. This parameter is invalid if auditing is disabled.

- **externalization_switch**

  Enable the audit data export service. The default value is DISABLE.

- **directory**

  Database directory that should be configured with an OS directory where the export service archives the audit data files.

- **file_prefix**

  File prefix to be used by the export service to create the file name where audit data is to be written. The default value is em_audit.

- **file_size**

  Maximum value of each file size. The default value for this is 5000000 bytes.

- **data_retention_period**

  Maximum period the Enterprise Manager repository stores audit data. The default value is 365 days.

Examples

Example 1

The following example enables all operations except LOGIN and LOGOUT:

```
emcli update_audit_settings
      -audit_switch="ENABLE"
      -operations_to_enable="ALL"
      -operations_to_disable="LOGIN;LOGOUT"
```

Example 2

```
emcli update_audit_settings
    -externalization_switch="ENABLE"
    -directory="EM_DIR"
    -file_prefix="my_audit"
    -file_size="10000"
    -data_retention_period="60"
```

```
emcli update_audit_settings
    -externalization_switch="ENABLE"
    -directory="EM_DIR"
```

update_db_password

Updates the target database password change in the Enterprise Manager Credential sub-system and can change the password on the target database as well. This verb also propagates the collection or monitoring credentials to Enterprise Manager Management Agents.

Format

```
emcli update_db_password
      -target_name="tname"
      -user_name="user_name"
      [-target_type="ttype"]
      [-change_all_references="yes/no"]
      [-change_at_target="yes/no"]
      [-input_file="tag1:file_path1;tag2:file_path2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **target_name**

  Name of the target.

- **user_name**

  Name of the database user.

- **target_type**

  Type of target. The possible values for target type in this verb are -oracle_database and - rac_database. The default value for this parameter is oracle_database.

- **change_all_references**

  Specify if the password must be changed for all references in Enterprise Manager. Possible values are:

  – **yes** — Update all password references in Enterprise Manager for a DBSNMP user who has an old password that matches the new password.

  – **no** — Update the password for the currently logged in user.

  The default value of this option is Yes.

- **change_at_target**

  Specify whether the password must also be changed on the target. This is not supported for a SYS user.

  – **yes** — Change the password on the target database.

  – **no** — Update the password only on Enterprise Manager.

  The default value of this option is No.

- **input_file**

  Path of the file that has old and new passwords. Use this option to hide passwords displayed on the command line. You must accompany each path with a tag referenced in the password options.

When you execute this verb with the input_file option, you are prompted to enter the following values in non-echo mode:

-old_password
-new_password
-retype_new_password

Examples

```
emcli update_db_password
     -target_name=myDB
     -user_name=Admin1

emcli update_db_password
     -target_name=myDB
     -user_name=Admin1
     -change_at_target=yes
```

update_diagchecks

Updates diagnostic check scripts for targets.

Format

```
emcli update_diagchecks
      -target_name=<target_name_to_be_updated>
      -target_type=<target_type_to_be_updated>
      [-input_file=targetList:<complete_path_to_file>]
```

Parameters

- **target_name**

  Name of the target to be updated.

- **target_type**

  Type of the target to be updated.

- **input_file**

  Specify a file name that contains a list of targets, one per line in the following format:

  ```
  <targetType>:<targetName>
  ```

update_host_password

Updates the changed host password in the credential sub-system. For collection or monitoring credentials, the password change is optionalso propagated to the Enterprise Manager Management Agent.

Format

```
emcli update_host_password
      -target_name="tname"
      -user_name="user_name"
      [-change_all_references="yes/no"]
      [-input_file="tag1:file_path1;tag2:file_path2;..."]
```

[ ] indicates that the parameter is optional

> **Note:**   When you execute this verb, you are prompted to enter the following values in non-echo mode:
>
> -old_password
> -new_password
> -retype_new_password

Parameters

- **target_name**

  Name of the target.

- **user_name**

  Name of the database user.

- **change_all_references**

  Specifies if the password must be changed for all references in Enterprise Manager for the given user.

  Possible values are:

  – Yes — Updates all references in Enterprise Manager for this password.

  – No — Updates the password for the current logged-in user. This is the default.

- **input_file**

  File path that has old and new passwords. This hides passwords. You must accompany each path with a tag referenced in the password s.

Examples

Example 1
The following example asks the user to enter the values of the old and new passwords, then retype the new password to update the new password in Enterprise Manager for this target reference.

```
emcli update_host_password
      -target_name=myHost
      -user_name=Admin1
```

Example 2

The following example asks the user to enter the values of the old and new passwords, then retype the new password to update the new password in Enterprise Manager for all users' credentials referenced with the myHost target name and Admin1 user name.

```
emcli update_host_password
      -target_name=myHost
      -user_name=Admin1
      -change_all_references=yes
```

update_monitoring_creds_from_agent

Finds targets on the Agent, retrieves the monitoring credentials, and updates them in the repository.

Format

```
emcli update_monitoring_creds_from_agent
      [-emd_list=<emd_list>]
      [-update_all]

[ ] indicates that the parameter is optional
```

Parameters

- **emd_list**

  You must provide either this parameter or the update_all option.

- **update_all**

  You must provide either this parameter or the emd_list option.

Examples

Example 1
The following example finds all the targets monitored by host1.oracle.com:1832 and host2.us.oracle.com:1832 that have monitoring credentials on the Agent but not in the repository, and updates the monitoring credentials in the repository.

```
emcli update_monitoring_creds_from_agent
      -emd_list="host1.oracle.com:1832;host2.us.oracle.com:1832"
```

Example 2
The following example finds all the targets that have monitoring credentials on the Agents but not in the repository, and updates the monitoring credentials in the repository.

```
emcli update_monitoring_creds_from_agent
      -update_all
```

update_operation_plan

Updates the SiteGuard operation plan.

Format

```
emcli update_operation_plan
        [-name=<plan_name>]
        [-step_number=<step_number>]
        [-target_host=<host_name>]
        [-error_mode=<error_mode>]
        [-enabled=<true|false>]
        [-execution_mode=<Serial|Parallel>]
        [-move=<Up|Down>]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the operation plan.

- **step_number**

  Number of the step that should be updated.

- **target_host**

  Target host name. Specifying this updates all of the steps involving this target host.

See Also

emcli create_operation_plan

emcli get_operation_plan_details

Examples

```
emcli update_operation_plan -name="austin-switchover"
        -step_number="1"
        -error_mode="Continue"
        -enabled="true"
        -execution_mode="Serial"

emcli update_operation_plan -name="austin-switchover"
        -step_number="5"
        -move="Up"

emcli update_operation_plan -name="austin-switchover"
        -target_host="myhost.domain.com"
        -error_mode="Continue"
        -enabled="true"
```

update_password

Updates passwords or other credentials for a given target.

Format

```
emcli update_password
        -target_type="ttype"
        -target_name="tname"
        -credential_type="cred_type"
        -key_column="column_name:column_value"
        -non_key_column="col:oldvalue:newvalue;..."
        [-input_file="tag1:file_path1;tag2:file_path2;..."]

[ ] indicates that the parameter is optional
```

Parameters

- **target_type**

  Type of target.

- **target_name**

  Name of the target.

- **credential_type**

  Credential type to use. The type must be a base type, not a derived type. A derived type contains the XML tag <CredentialTypeRef> within its definition.

- **key_column**

  Name and value of the key column for the credential type. Usually, the key column represents the user name. To get the key column for a target type, you can execute following EM CLI verbs:

  emcli get_credential_type_info — Displays key columns for all target types.

  emcli get_credential_type_info -target_type=<target_type> — Displays key columns for a specific target type.

- **non_key_column**

  Name, old value, and new value of the non-key column(s) to modify. Usually, this is the name of the password column. Alternatively, a tag from the -input_file argument can be used so that the credential values are not seen on the command line. You can specify this parameter more than once.

- **input_file**

  Path of the file that has non_key_column argument(s). This option is used to hide passwords. You must accompany each path with a tag that is referenced in the non_key_column argument. You can specify this option more than once.

  You can obtain the list of columns and the credential types they belong to by using the emcli show_credential_type_info command.

Examples

Example 1

```
emcli update_password
```

```
        -target_type=oracle_database
        -target_name=myDB
        -credential_type=DBCreds
        -key_column="DBUserName:joe"
        -non_key_column="DBPassword:oldPass:newPass"
        -non_key_column="DBRole:normal:sysdba"
```

Example 2

In the following example, FILE1 is a tag used to refer to the contents of passwordFile. The contents of the password file is:

```
DBPassword:oldPass:newPass;DBRole:normal:sysdba
```

Note that this example has the same effect as Example 1.

```
emcli update_password
        -target_type=oracle_database
        -target_name=myDB
        -credential_type=DBCreds
        -key_column="DBUserName:joe"
        -non_key_column="FILE1"
        -input_file="FILE1:passwordFile"
```

update_procedure_input

Updates the configuration of a deployment procedure.

Format

```
emcli upate_procedure_input
        -name="name_of_procedure_configuration"
        [-input_file="file_path\file_name"]
        [-grants="users_and_access_levels"]
        [-schedule=
            start_time:yyyy/MM/dd HH:mm;
            tz:<java_timezone_ID>;
            grace_period:xxx;
        ]
        [-notification="procedure status"]

[ ] indicates that the parameter is optional
```

Parameters

- **name**

  Name of the configuration for the procedure.

- **input_file**

  Input property file for the deployment procedure. The file_path should point to a file containing the data property file.

- **grants**

  Specifies users and their corresponding access levels as a string of user:privilege pairs, each separated by a semi-colon (;). The user is an Enterprise Manager user name, and the privilege is either VIEW_JOB or FULL_JOB.

  See the example below.

- **schedule**

  Schedule for the deployment procedure. If not specified, the procedure is executed immediately.

  – **start_time** — When the procedure should start.

  – **tz** — Optional timezone ID.

  – **grace_period** — Optional grace period in minutes.

- **notification**

  Status of the procedure.

Example

```
emcli update_procedure_input
    -name=configProcedure
    -input_file=/home/data.properties -grants="user1:VIEW_JOB;user2:FULL_JOB"
    -schedule="start_time:2011/8/21 21:23;tz:America/New_York;grace_period:60"
    -notification="scheduled, action required, running"
```

update_siteguard_configuration

Updates the Site Guard configuration to add additional standby systems. One primary system can be associated with one or more standby systems.

> **Note:** If you update the site configuration, you must also update the operation plan, as described in update_monitoring_creds_from_agent.

Format

```
emcli update_siteguard_configuration
     [-primary_system_name=<primary_system_name>]
     [-standby_system_name=<standby_system_name>]
```

Parameters

- **primary_system**

  Name of the primary system.

- **standby_system**

  Name of the standby system. You can specify this parameter more than once.

See Also

create_siteguard_configuration

delete_siteguard_configuration

Examples

```
emcli update_siteguard_configuartion
     -primary_system_name="BISystem1"
     -standby_system_name="BISystem2"
```

update_siteguard_credential_association

Updates the credential association.

Format

```
emcli update_siteguard_credential_association
      [-system_name=<system_name>]
      [-target_name=<target_name>]
      [-credential_type=<credential_type>]
      [-credential_name=<credential_name>]
      [-use_preferred_credential=true|false]
      [-credential_owner=<credential_owner>]
```

Parameters

- **system_name**

  Name of the system.

- **target_name**

  Optional name of the target.

- **credential_type**

  Type of credential, which can be HostNormal, HostPrivileged, WLSAdmin, or DatabaseSysdba.

- **credential_name**

  Name of the credential.

- **use_preferred_credential**

  Use a preferred credential instead of the named credential. You need to specify credential_name if this option is false.

- **credential_owner**

  Owner of the credential.

See Also

delete_siteguard_credential_association

create_siteguard_credential_association

Examples

Example 1

```
emcli update_siteguard_credential_association
      -system_name="austin-system"
      -credential_type="HostNormal"
      -credential_name="HOST-SGCRED"
      -credential_owner="sysman"
```

Example 2

```
emcli update_siteguard_credential_association
      -system_name="utah-system"
      -credential_type="HostPrivileged"
```

```
-use_preferred_credential="true"
-credential_owner="sysman"
```

Example 3

```
emcli update_siteguard_credential_association
        -system_name="austin-system"
        -target_name="austin-database-instance"
        -credential_type="DatabaseSysdba"
        -credential_name="HOST-DBCRED"
        -credential_owner="sysman"
```

update_siteguard_script

Updates the path and the all_hosts flag associated with any script.

Format

```
emcli update_siteguard_script
      -script_id=<script_ID>
      [-path=<script_path>]
      [-credential_type=<type_of_credential>]
      [-all_hosts=true|false]
```

Parameters

- **script_id**

  ID associated with the script.

- **path**

  Optional path to the script.

- **credential_type**

  Type of credential, which can be either HostNormal or HostPrivileged.

- **all_hosts**

  Enables the script to run on all the hosts in the system. For example: true or false.

See Also

create_siteguard_script

get_siteguard_scripts

Examples

```
emcli update_siteguard_script
      -script_id="10"
      -path="/tmp/newprescript"
      -all_hosts="true"

emcli update_siteguard_script -script_id="16"
          -path="/tmp/script"
          -credential_type="HostPrivileged"
```

update_swlib_entity

Modifies an entity in the software library. A new revision of the entity is created by default. Changing only the description or attribute values does not create a new revision, and such changes will be visible across all existing revisions of the entity.

Format

```
emcli update_swlib_entity
        -entity_rev_id="entity_rev_id"
        [-desc="entity_desc"]
        [-attr="<attr_name>:<attr_value>"]
        [-prop="<prop_name>:<prop_value>"]
        [-secret_prop="<secret_prop_name>:<secret_prop_value>"]
        [-note="note_text"]
        [-use_latest_revision]

[ ] indicates that the parameter is optional
```

Parameters

- **entity_rev_id**

    Identifier of the entity revision. The software library home page exposes the identifier for folders and entities as a custom column (Internal ID) and is hidden by default.

- **desc**

    Description of the entity. The new description is visible to all existing revisions.

- **attr**

    An attribute and its value, separated by a colon (:). To specify values for multiple attributes, repeat this option. The new attribute value is visible to all existing revisions.

- **prop**

    Configuration property and its value, separated by a colon (:). To specify values for multiple attributes, repeat this option.

- **secret_prop**

    Configuration property and its secret value separated by a colon (:). It is recommended that the secret value not be specified on the command line. If omitted from the command line, the value is prompted for. To specify values for multiple properties, repeat this option.

- **note**

    Note on the entity. For multiple notes, repeat this option.

- **use_latest_revision**

    Indicates that the the latest revision of the entity should be updated instead of the revision identified by entity_rev_id.

Examples

The following example modifies the entity revision identified by entity_rev_id. The entity revision identifier value can be found from the Software Library home page. The

software library home page exposes the identifier for folders and entities as a custom column, which is hidden by default.

A new description is specified. Values for the entity attributes (PRODUCT, PRODUCT_VERSION and VENDOR) are specified. The value for the DEFAULT_ HOME configuration property is specified. A note on the entity is also specified.

A new revision is created for the modifications, but the specified entity revision (identified by entity_rev_id) remains unchanged. The identifier of the newly created entity is printed on the standard output.

```
entity_rev_id="oracle:defaultService:em:provisioning:1:cmp:COMP_Component:SUB_
Generic:B1B1880C6A8C62AAE040548C4D14:0.1"
        -entity_desc="myAcmeInstall description"
        -attr="PRODUCT:Acme"
        -attr="PRODUCT_VERSION:3.0"
        -attr="VENDOR:Acme Corp"
        -prop="DEFAULT_HOME:/u01/acme3/"
        -note="myAcmeInstall for test servers"
```

update_target_password

Updates the changed target password in the Enterprise Manager credential sub-system. For collection or monitoring credentials, the password change is also propagated to Enterprise Manager Management Agents.

Format

```
emcli update_target_password
      -target_type="ttype"
      -target_name="tname"
      -key_column="column_name:column_value"
      [-change_all_references="yes/no"]
      [-input_file="tag1:file_path1;tag2:file_path2;..."]

[ ] indicates that the parameter is optional
```

> **Note:** When you execute this verb, you are prompted to enter the following values in non-echo mode:
>
> -old_password
> -new_password
> -retype_new_password

Parameters

- **target_type**

  Type of target.

- **target_name**

  Name of the target.

- **key_column**

  Name and value of the key column for the credential type. The key column usually represents the user name.

  To obtain the key column for a target type, enter the following command:

  ```
  emcli get_credential_type_info -target_type=<target_type>"
  ```

  To obtain the key column for all target types, enter the following command:

  ```
  emcli get_credential_type_info
  ```

- **change_all_references**

  Specifies if the password must be changed for all references in Enterprise Manager for the given user.

  Possible values are:

  – Yes — Updates all references in Enterprise Manager for this password.

  – No — Updates the password for the current logged-in user. This is the default.

- **input_file**

  File path that has old and new passwords. This option hides passwords. You must accompany each path with a tag referenced in the password options. You can specify this option more than once.

Examples

Example 1
The following example asks the user to enter the values of the old and new passwords, then retype the new password to update the new password in Enterprise Manager for this target reference.

```
emcli update_target_password
      -target_type=host
      -target_name=myHost
      -key_column=HostUserName:Admin1
```

Example 2
The following example asks the user to enter the values of the old and new passwords, then retype the new password to update the new password in Enterprise Manager for all users' credentials referenced with the mydb target name and Admin1 user name.

```
emcli update_target_password
      -target_type=oracle_database
      -target_name=mydb
      -key_column=DBUserName:Admin1
      -change_all_references=yes
```

update_ticket_status

Updates the ticket status and last modified time stamp in Enterprise Manager from the external ticketing system based on the ticket_guid and connector_guid.

Format

```
emcli update_ticket_status
    -ticket_guid="ticket guid"
    -connector_guid="connector guid"
    -status="Incident status"
    -last_updated_date="last modified date"
    -date_format=
```

Parameters

- **ticket_guid**

  Ticket ID for which the status is modified.

- **connector_guid**

  Ticketing Connector ID.

- **status**

  Modified status of an incident ticket.

- **last_updated_date**

  Specifies the last modified date of an incident ticket.

- **date_format**

  Specify a date format followed in the Ticketing System, as in "MM/dd/yyyy hh:mm:ss" if the date field in Incident management is "10/13/2009 5:38:24 AM".

Example

The following example updates the ticket INC00000024 status as 'In Progress' in Enterprise Manager after the same ticket status was recently modified on the ticketing system.

```
emcli update_ticket_status
  -ticket_guid="INC21000024"
  -connector_guid="ccc1234"
  -status="2"
  -last_updated_date="05/28/2011 3:14:56PM"
  -date_format="MM/dd/yyyy hh:mm:ss"
```

upgrade_database

Upgrades a database.

Format

```
emcli upgrade_database
  -dbTargetName="target_to_be_upgraded"
  -dbTargetType="oracle_database|rac_database"
  -newOracleHome="directory_full_path"
  -hostCreds="named_credentials"
  -sysdbaCreds="named_credentials"
  [-precheck="YES|NO|ONLY"
  [-ignoreWarnings]
  [-diagnosticDest="diagnostic_destination"]
  [-disableArchiveLogMode]
  [-recompileInvalidObjects]
  [[-restoreSettingsOnly] | [-backupLocation="backup_location_full_path"]]
  [-listeners=<name:port[:NEW]>
  [-scriptsFromSoftwareLibrary "scripts_from_software_library"]
  [-beforeUpgradeCustomScript="custom_SQL_file_name"]
  -continueOnScriptError
  [-afterUpgradeCustomScript="Custom_SQL_file_name_upgrade"]
  [-noBlackout]

[ ] indicates that the parameter is optional
```

Parameters

- **dbTargetName**

  Enterprise Manager target name of the database to be upgraded. Versions 10.2.0.4 and above are supported for upgrade.

- **dbTargetType**

  Target type of the database — oracle_database for a single instance database, or rac_database for a cluster database.

- **newOracleHome**

  New Oracle Home directory full path. Upgrade to 11g Release 2 and later is supported. Does not support a database downgrade.

- **hostCreds**

  Named host credentials of the user who owns the Oracle Home installation. Should have necessary privileges on the database files to be upgraded.

- **sysdbaCreds**

  Named database credentials having SYSDBA privileges on the database to be upgraded.

- **precheck**

  Option to run prerequisite checks during the upgrade job. Valid values are:

  YES — Run prerequisite checks and proceed to the database upgrade if there are no errors during prerequisite checks.

  NO — Proceed to the database upgrade directly. Do not run prerequisite checks.

  ONLY — Run prerequisite checks only. Do not upgrade the database.

■ **ignoreWarnings**

Ignores any warnings during prerequisite checking and proceeds with the upgrade. Used only when pre-check is set to YES, otherwise ignored. Does not ignore errors.

■ **diagnosticDest**

Full directory path for Oracle trace and diagnostic files for the upgraded database. By default, ORACLE_BASE is used as the location.

■ **disableArchiveLogMode**

Disable archive logging during the database upgrade.

■ **recompileInvalidObjects**

The upgrade process may invalidate the objects in the database. You can choose to recompile invalid objects at the end of the upgrade. This increases the upgrade time, but minimizes subsequent latencies caused by on-demand automatic recompilation at run time.

■ **restoreSettingsOnly**

Reverts only the configuration changes made during the upgrade if upgrade fails. You can restore the database outside the upgrade using your custom restore strategy. Choose this option if you already have a custom backup and restore strategy for this database. In case of an upgrade failure, this setting will be used.

■ **backupLocation**

Full directory path to back up the database. Performs a full backup of the database. A script will be created to restore the database. All files are placed in the specified backup location. Reverts all the changes made during the upgrade if the upgrade fails.

■ **listeners**

Comma-separated list of the listener name and port (name1:port1,name2:port2) to register the upgraded database. Specify at least one listener in the case of a single-instance database target. These listeners should be configured in the new Oracle home or TNS_ADMIN location. Additionally, you can choose to create a new listener in the new Oracle home by specifying :NEW (name1:port1:NEW).

■ **scriptsFromSoftwareLibrary**

Specify the custom scripts from the software library components. The parameters 'beforeUpgradeCustomScript' and 'afterUpgradeCustomScript' are interpreted as entity URNs of the components that contain the scripts.

■ **beforeUpgradeCustomScript**

Full file path of the custom SQL script to be run before the database upgrade.

■ **continueOnScripteError**

Ignores a non-zero exit code when executing a custom SQL script and continues the upgrade job.

■ **afterUpgradeCustomScript**

Full file path of the custom SQL script to be run after the successful database upgrade.

■ **noBlackout**

Suppresses a blackout of the database target. A blackout suspends monitoring of the database target from Enterprise Manager, which is the default behavior during a database upgrade.

Examples

```
emcli upgrade_database
   -dbTargetName=test1 -dbTargetType=oracle_database
   -newOracleHome=/u01/app/oracle/product/11.2.0/dbhome_2 -hostCreds=HOST_CREDS
   -sysdbaCreds=SYSDBA_CREDS -precheck=YES -ignoreWarnings -disableArchiveLogMode
   -beforeUpgradeCustomScript=/home/user1/sqlfiles/script1.sql
   -continueOnScriptError
   -afterUpgradeCustomScript=/home/user1/sqlfiles/script2.sql
   -diagnosticDest=/u01/app/oracle
   -recompileInvalidObjects -noBlackout
```

upload_ats_test_databank_file

Uploads a databank file for the specified ATS test.

Format

```
emcli upload_ats_test_databank_file
      -name=target name
      -type=target type
      -testname=test name
      -testtype=test type
      -databankAlias=databank alias
      -input_file:databank=databank file
```

Parameters

- **name**

  Name of the target.

- **type**

  Name of the target type.

- **testname**

  Name of the test.

- **testtype**

  Type of test.

- **databankAlias**

  Databank alias.

- **input_file**

  Databank file.

Examples

```
emcli upload_ats_test_databank_file
      -name="Service Name"
      -type="generic_service"
      -testname="Test Name"
      -testtype="OATS"
      -databankAlias="alias1"
      -input_file:databank="databankFile.csv"
```

upload_patches

Uploads patches to the software library.

Format

```
emcli upload_patches
      -from_host="host_name"
      -patch_files="metadata_file_path;ZIP_file_path"
      [-cred_name="name" -cred_owner="owner"]
```

```
[ ] indicates that the parameter is optional
```

Parameters

- **from_host**

  Host from which to get the given patch files.

- **patch_files**

  List of patch file paths. A metadata file and a zip file must be provided.

- **cred_name**

  Named credential used to authenticate the host from which the given patch files are to be uploaded. If you do not provide this option, the normal preferred credential of the host from which the given patch files are to be uploaded will be used by default.

- **cred_owner**

  Owner of the named credentials used to authenticate the host from which the given patch files are to be uploaded.

Examples

```
emcli upload_patches -patch_files="/scratch/p13741363_112310_Linux-x86-64_
M.xml;/scratch/p13741363_112310_Linux-x86-64.zip" -from_host=h1.us.oracle.com
```

```
emcli upload_patches -patch_files="/scratch/p13741363_112310_Linux-x86-64_
M.xml;/scratch/p13741363_112310_Linux-x86-64.zip" -from_host=h1.us.oracle.com
-cred_name=AIMECRED -cred_owner=SYSMAN
```

See Also

create_patch_plan
delete_patches
describe_patch_plan_input
get_connection_mode
get_patch_plan_data
list_aru_languages
list_aru_platforms
list_aru_products
list_aru_releases
list_patch_plans
search_patches
set_connection_mode
set_patch_plan_data
show_patch_plan

submit_patch_plan

Also see "Patching Using EM CLI" in the *Enterprise Manager Lifecycle Management Administrator's Guide*:

http://docs.oracle.com/cd/E24628_01/em.121/e27046/emcli.htm#BABDEGHB

upload_swlib_entity_files

Uploads one or more files to an entity revision in the software library.

Format

```
emcli upload_swlib_entity_files
      -entity_rev_id="entity_rev_id"
      -file="<abs_file_path>[;<new_file_name>]"
      -host="hostname"
      [-credential_set_name="setname"] | [-credential_name="name"
        -credential_owner="owner"]
      [-upload_storage="<storage_location_name>;<storage_type>"]
      [-use_latest_revision]


[ ] indicates that the parameter is optional
```

Parameters

- **entity_rev_id**

  Identifier of the entity revision. The software library home page exposes the identifier for folders and entities as a custom column (Internal ID), and is hidden by default.

- **file**

  Absolute path of the file to be uploaded. The file name stored in the software library on upload defaults to the name of the file being uploaded. You can optionally specify a different file name, separated by a semicolon (;).

- **host**

  Target name of the host where the files are available.

- **credential_set_name**

  Set name of the preferred credential stored in the repository for the host target, which can be one of:

  HostCredsNormal — Default unprivileged credential set

  HostCredsPriv — Privileged credential set

- **credential_name**

  Name of a named credential stored in the repository. You must specify this option along with the credential_owner option.

- **credential_owner**

  Owner of a named credential stored in the repository. You must specify this option along with the credential_name option.

- **upload_storage**

  Destination storage location and type for the upload, separated by a semicolon (;). The location specified must be in 'active' status. This defaults to the storage type and location of the first upload location configured for the software library.

  The storage type can be one of:

  OmsShared — OMS shared file system

OmsAgent — OMS Agent file system

- **use_latest_revision**

  Flag indicating that the upload should occur to the latest revision of the entity instead of the revision identified by entity_rev_id.

Examples

Example 1

The following example uploads the file '/u01/acme_downloads/file1.zip' to the entity revision identified. The file present on the host 'fs1.us.acme.com' should be accessible using the preferred credential set for the 'HostCredsNormal' credential set for the user logged in to EMCLI. The host must be a managed host target in Enterprise Manager, and the Agent on this host must be up and running.

```
emcli upload_swlib_entity_files
      -entity_rev_id="oracle:defaultService:em:provisioning:1:
        cmp:COMP_Component:SUB_Generic:
        B1B1880C6A8C62AAE040548C42832D14:0.1"
      -file="/u01/acme_downloads/file1.zip"
      -host="fs1.us.acme.com"
      -credential_set_name="HostCredsNormal"
```

Example 2

The following example uploads the files to the specified entity revision. The file present on the host 'fs1.us.acme.com' should be accessible using the credential named 'MyAcmeCreds' owned by 'ACME_USER'. File '/u01/acme_downloads/file1.zip' after upload will be associated with the entity revision as 'newfile1.zip'. A new revision will be created from the latest revision of the entity.

```
emcli upload_swlib_entity_files
      -entity_rev_id="oracle:defaultService:em:provisioning:1:
        cmp:COMP_Component:SUB_Generic:
        B1B1880C6A8C62AAE040548C42832D14:0.1"
      -file="/u01/acme_downloads/file1.zip;newfile1.zip"
      -file="/u01/acme_downloads/file2.zip"
      -host="fs1.us.acme.com"
      -credential_name="MyAcmeCreds"
      -credential_owner="ACME_USER"
      -use_latest_revision
```

verify_updates

Checks for archives that are missing in the software library, and prints steps to download them and re-import them to the software library.

Format

```
emcli verify_updates
```

Parameters

None.

version

Lists EM CLI verb versions or the EM CLI client version.

Format

```
emcli version
    [-verb_name=<verb_name_filter>]
    [-exact_match]
    [-noheader]
    [-script | -format=
                [name:"pretty|script|csv"];
                [column_separator:"column_sep_string"];
                [row_separator:"row_sep_string"];
    ]

[ ] indicates that the parameter is optional
```

Parameters

- **verb_name**

  Verb name filter. Selects matching EM CLI verb names. When you specify this , an output table shows the version for each verb whose name matches <verb_name_filter>. The EM CLI client version is displayed when you do not specify this option.

  Verb filters use regular expression pattern matching unless you specify -exact_match. A zero-length filter matches everything.

  > **Note:** For Unix csh, use single quotes around a filter value containing '$' .

- **exact_match**

  Uses exact matching for filters.

- **noheader**

  Displays tabular information without column headers.

- **script**

  This is equivalent to `-format="name:script"`.

- **format**

  Format specification (default is `-format="name:pretty"`).

  - `format="name:pretty"` prints the output table in a readable format not intended to be parsed by scripts.

  - `format="name:script"` sets the default column separator to a tab and the default row separator to a newline. The column and row separator strings can be specified to change these defaults.

  - `format="name:csv"` sets the column separator to a comma and the row separator to a newline.

  - `format=column_separator:"column_sep_string"` column-separates the verb output by <column_sep_string>. Rows are separated by the newline character.

     – row_separator:"row_sep_string" row-separates the verb output by
        <row_sep_string>. Rows are separated by the tab character.

Output Columns

Verb, Version (when -verb_name is specified)

Examples

Example 1
The following example shows the version for all verbs:

```
emcli version -verb_name=
```

Example 2
The following example shows the version for all verbs with names that exactly match
the string "sync" :

```
emcli version -verb_name=sync -exact_match
```

Example 3
The following example shows the version for all verbs with names starting with "log:"

```
emcli version -verb_name="^log"
```

Example 4
The following example shows the version for all verbs with names that end with "in:"

```
emcli version -verb_name="in$"
```

Example 5
The following example shows the version for all verbs with names that contain a
substring matching "elp" or with names that begin with "ver" or "lo", contains "i", and
ends with "n:"

```
emcli version -verb_name="elp|^(ver|lo).*i.*n$"
```

Example 6
The following example shows the version for all verbs with names that exactly match
the string "setup." Alternatively, you could use the -exact_match .

```
emcli version -verb_name="^setup$"
```

view_redundancy_group

Shows the present configuration of the redundancy group.

Format

```
emcli view_redundancy_group
     -redundancyGroupName="redGrpName"
```

Parameters

- **redundancyGroupName**

  You must specify a single redundancy group name. The target name should be the same as present in the repository, and it should be of target type= "generic_redundancy_group".

Examples

The following example shows the details for the 'redGrp1' Redundancy Group.

```
emcli view_redundancy_group -redundancyGroupName='redGrp1'
```

3

Error Code Reference

This chapter documents errors and associated codes returned by EM CLI. You can use EM CLI return codes to manage the control flow in a workflow/scripting environment. EM CLI return codes for Verb errors are positive integers. A Verb returns either 0 (successful execution) or an error number.

The following sections provide reference tables for these types of errors:

- EM CLI infrastructure
- OMS connection
- File-fed option
- Built-in verb

3.1 EM CLI Infrastructure Errors

Any execution of the EM CLI client could result in the following errors.

Table 3–1   Infrastructure Errors

Error Code	Description
242	A Verb has encountered a problem with a dependency specific to the implementation of the Verb (INSIDE of its abstraction barrier) unrelated to the Verb's semantics.
248	Configuration files are corrupt or inaccessible.
253	The command name is not recognized.
254	Internal system error.

3.2 OMS Connection Errors

Verbs that execute at the OMS return these error codes as indicated in the listing for each applicable verb.

Table 3–2   OMS Connection Errors

Error Code	Description
243	License has not been accepted by the current user.
249	Cannot connect to the OMS.
250	Wrong credentials for log in to the OMS.

3.3 File-fed Option Errors

Verbs that allow for file-fed options (rather than options where the values are explicitly defined on the command line) can return the following error codes.

Table 3–3   File-Fed Option Errors

Error Code	Description
244	Cannot find an option value file.
245	Cannot read in an option value file.
246	An option value file is too big.

3.4 Built-in Verb Errors

The following error codes are returned by each verb (not including EM CLI infrastructure errors that apply to all verbs).

Table 3–4   Built-In Verb Errors

Verb	Error Code
add_beacon	0—Beacon added successfully.
	129— Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	173—Beacon does not exist.
	201—Beacon is already in the monitoring beacons list.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
add_group_to_mpa	2—I/O error occurred while writing to the MPA file.
	3—The specified MP already exists in the MPA.
	4—The group name is empty or not specified.
	223—The supplied options are syntactically incorrect.
add_mp_to_mpa	1—File does not exist, is unreadable, or an I/O error occurred.
	2—I/O error occurred while writing to the MPA file.
	3—The specified MP already exists in the MPA.
	4—The target-type definition file cannot be parsed.
	5—The MPA filename is not between 1 and 255 characters.
	6 —A file of a particular file type is required for another file.
	223—The supplied options are syntactically incorrect.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
add_target	1—The supplied target type does not exist. Unable to retrieve target metadata from the specified host's Management Agent.
	2—Host does not exist.
	3—Agent does not exist.
	4—Group does not exist.
	5—No monitoring credentials set found for target in the repository.
	6—Target instance already exists in the repository.
	7—The supplied target properties are incomplete.
	8—One or more of the supplied target properties are invalid.
	15—Target deletion in progress.
	20—Unable to connect to the specified host's Agent.
	21—Unable to save the target instance to the specified host's Agent.
	22—Cannot add more than one Agent target for a single Agent URL.
	23—Unable to add an instance of an Agent target without a URL.
	219—Insufficient privileges to add the target to the group.
	223—Unable to parse command line correctly. Invalid argument value.
	File-Fed Option Errors—The errors associated with file-fed options.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
apply_privilege_delegation_setting	0—Setting successfully applied.
	2—Setting does not exist.
	3—All or some of the targets are invalid.
	129—Syntax error. The displayed message indicates which argument is syntactically incorrect.
apply_template_tests	1—Error processing input XML file.
	4—Insufficient privileges for apply template.
	6—Target does not exist.
	7—Incompatible template and target types during apply.
	8—Test(s) specified for overwriteExisting do not exist in the template.
	9—Key test(s) specified as disabled for apply.
	10—Stepgroup contains a step that does not exist in the file.
	11—Some text property in file does not conform to valid syntax.
	12—Some text property contains variable but variable value is missing.
	13—Some transaction property/threshold/collection setting does not conform to required restrictions.
	50—Generic error.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
argfile	Possible return error codes consist of the following list plus all of the errors returned by the Verb specified in the command line file for execution.
	244—The file does not exist.
	245—There is a problem reading in the file or it does not exist.
	246—The file ends inside a quoted token.
	247—The argfile options are specified incorrectly.
assign_test_to_target	0—Test assigned to target type successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	190—Test or target type invalid.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
change_service_system_assoc	0—Service system changed successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	171—System <system> does not exist.
	172—Key component does not exist.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
clear_credential	1—Target type does not exist.
	2—Target does not exist.
	3—Credential set does not exist.
	4—Insufficient privileges.
	5—Credential column does not exist.
create_aggregate_service	1—Target does not exist.
	2—Target exists.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
create_blackout	1—Blackout X already exists.
	2—Only Super Administrators are allowed to add a new reason (use get_ blackout_reasons).
	3—Agent targets cannot be directly blacked out.
	217—The blackout end_time cannot be in the past.
	The dates specified will never cause this blackout to take effect.
	The difference between the end_time and the start_time must be equal to the duration.
	The difference between the repeat interval and the duration must be at least X minutes.
	The duration must be -1 (for indefinite blackouts) or positive.
	The duration must be at least X minutes.
	219—Current user does not have OPERATOR privilege over all blackout targets.
	220—Target X does not exist.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
create_group	1—Group X already exists.
	2—Cannot add target X to typed group of base type Y.
	218—Group X is currently in the process of being deleted.
	219—Current user does not have privilege X over all member targets.
	220—Member target X does not exist.
	223—Unable to parse command line correctly.
	Invalid argument value.
	Group type is invalid.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
create_privilege_ delegation_setting	0—Setting successfully created.
	129—Syntax error. The displayed message indicates which argument is syntactically incorrect.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
create_red_group	0—Redundancy Group "<red_group_name>" created successfully.
	1—Redundancy Group "<red_group_name>" of target type <red_group_type> already exists.
	2—Cannot add target "<member_target_type>" to typed group of base type "<red_group_type>".
	3—Time Zone Region <timezone_region> does not exist.
	4—Redundancy Group Type "<red_group_type>" is invalid.
	218—Redundancy Group "<red_group_name>:<red_group_type>" is currently in the process of being deleted.
	220—Target "<member_target_name>:<member_target_type>" does not exist.
	223—Redundancy Group name "<red_group_name>" is not valid. It may contain only alphanumeric characters, multi-byte characters, a space, "-", "_", ".", ":", and have a maximum length of 256 characters.
	223—User name "<owner>" is not valid. It must begin with an alphabetic character, contain only alphanumeric characters, underscores (\"_\"), or periods (\".\"), and have a maximum length of 256 characters.
	223—Invalid value for parameter "add_targets": "<add_targets>". Reason: "<add_targets>" is not a name-value pair.
	223—Member Targets not of same type.
	223—"<generic_redundancy_group>" does not support member of type "<member_target_type>" .
create_role	1—Role by same name already exists.
	2—User with same name as role already exists.
	4—Privilege is invalid or nonexistent.
	5—Target specified in one of the privileges is invalid.
	6—The Super Administrator privilege cannot be granted to a role.
	7—Role does not exist.
	8—Group specified in one of the privileges is invalid.
	9—Job in privilege is invalid or nonexistent.
	10—Creating a role that you are assigning to the new role.
	11—The specified user does not exist.
	219—User is unauthorized to perform this action.
	223—Unable to parse command line correctly.
	Invalid argument value.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
create_service	0—Web application created successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	130—Missing key components.
	151—Test validation failed.
	171—System <system> does not exist.
	172—Key component does not exist.
	173—Beacon does not exist.
	181—No key tests defined.
	182—No key beacons defined.
	200—Service <target_name> already exists.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
create_system	0—System "<system_name:system_type>" created successfully.
	110—System "<system_name:system_type>" already exists.
	120—Member target "<member_target_name>:<member_target_type>" does not exist.
	122—Type "<system_type>" is not a valid System type.
	123—Time Zone Region "<timezone_region>" does not exist.
	130—Type meta version "<type_meta_ver>" is invalid.
	223—System name "<system_name>" is not valid. It must begin with an alphabetic char, contain only alphanumeric chars or any of "- _.:", and have a maximum length of 256 chars.
	223—Type meta version "<type_meta_ver>" is invalid. It must contain only numeric and "." characters, and have a maximum length of 8 chars.
	223—Timezone_region cannot be null or blank.
	223—Invalid value for parameter "add_members": "<add_members>". Reason: "<add_members>" is not a name-value pair.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
create_user	1—Target specified in one of the privileges is invalid.
	2—Group specified in one of the privileges is invalid.
	3—Job specified in one of the privileges is invalid.
	4—One of the specified privileges is invalid.
	5—Such user already exists.
	6—One or more roles to be granted to the new user does not exist.
	7—A role with the same name as the new user already exists.
	218—A delete is pending against this user until all blackouts and jobs submitted by this user are stopped.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly:
	Invalid argument value.
	User name is somehow invalid.
	Supplied password does not have the proper format. Example: Password left empty.
	File-Fed Option Errors—The errors associated with file-fed options.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_blackout	1—Blackout X created by user Y does not exist.
	2—Cannot delete a blackout that has not ended or was not stopped.
	219—You (X) do not have the SUPER_USER privilege needed to stop, delete, or modify blackout Y created by user Z.
	Only the blackout owner can stop, delete, or modify the blackout.
	Current user does not have OPERATOR privilege over all blackout targets.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_group	1—Group X does not exist.
	218—Group X is currently in the process of being deleted.
	219—Current user does not have sufficient privileges to perform this action.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_job	1—Specified job is invalid or non-existent.
	219—User has insufficient privileges to perform this operation.
	218—Some executions are not stopped when delete happens.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
delete_metric_promotion	0—SUCCESS
	223—SYNTAX_ERRNUM: Input is malformed.
	255—VERB_FAILED_ERRNUM: Back-end validation fails.
delete_privilege_delegation_settings	0—Setting successfully deleted.
	2—All or some of the names are invalid.
	129—Syntax error. The displayed message indicates which argument is syntactically incorrect.
delete_role	1—Role does not exist.
	219—User is unauthorized to perform this action.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_system	0—System "<system_name:system_type>" deleted successfully.
	121—System "<system_name:system_type>" does not exist.
	122—Type "<system_type>" is not a valid System type.
	219—Current user does not have sufficient privileges to perform this action.
	223—System name "<system_name>" is not valid. It must begin with an alphabetic character, contain only alphanumeric characters or any of "- _.:", and have a maximum length of 256 chars.
delete_target	15—Target deletion in progress.
	219—Insufficient privileges to delete specified target.
	220—Target does not exist.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
delete_test	0—Test deleted successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	174—Test does not exist.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
delete_user	1—Cannot delete the repository owner.
	2—Specified user does not exist.
	3—Cannot delete the current user.
	218—A delete is pending against this user until all blackouts and jobs submitted by this user are stopped.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
disable_audit	223—Syntax Error.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
disable_test	0—Test disabled successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	174—Test does not exist
	203—Test already disabled.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
enable_audit	223—Syntax Error.
enable_test	0—Test enabled successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	174—Test does not exist
	202—Test already enabled.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
execute_hostcmd	0—Command execution succeeded for all targets.
	2—Command execution failed for one or more targets. Detailed errors will be displayed for each failed target.
	3—Invalid or unknown targets in the targets list.
	4—Preferred credentials are missing for one or more targets.
	5—Invalid credential set name.
	223—Unable to parse the command line properly.
execute_sql	0—Command execution succeeded for all targets.
	2—Command execution failed for one or more targets. Detailed errors will be displayed for each failed target.
	3—Invalid or unknown targets in the targets list.
	4—Preferred credentials are missing for one or more targets.
	5—Invalid credential set name.
	223—Unable to parse the command line properly.
export_template	223—Unable to parse command line correctly, or an exception was thrown during SQL handling.
	245—There is a problem writing to the file.
extract_template_tests	2—Error serializing XML output.
	3—Insufficient privileges for extract template.
	5—Template does not exist in repository.
	50—Generic error.
get_aggregate_service_info	1—Target does not exist.
	2—Target exists.
get_aggregate_service_ members	1—Target does not exist.
	2—Target exists.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
get_blackout_details	1—Blackout X created by user Y does not exist.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_blackout_reasons	OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_blackout_targets	1—Host X does not exist.
	223—Unable to parse command line correctly.
	220—Target X does not exist.
get_blackouts	1—Host X does not exist.
	220—Target X does not exist.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_group_members	1—Group X does not exist.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_groups	Other than the confirmation message, the get_groups verb only generates syntax errors. The SQL invoked by get_groups does not throw any exception.
	0—All groups (TargetName, TargetType) in the repository are displayed.
	223—Syntax Error: Argument -script cannot be specified with a value.
	223—Syntax Error: -format argument "name" value must match one of these strings: "script|pretty|csv".
	223—Syntax Error: Invalid value for parameter "format": "name:<format_name>;column_separator=<column_separator_char>". Reason: "column_separator=column_separator_char" is not a name-value pair.
	223—Syntax Error: -format argument contains an unrecognized key name <key_name>
get_jobs	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
get_system_members	121—System "<system_name:system_type>" does not exist.
get_targets	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
grant_privs	2—User does not exist.
	3—Invalid privilege.
	4—Invalid target privilege.
	5—Invalid globally unique identifier (GUID).
	6—One or more targets are not groups.
	7—Specified job does not exist.
	8—Privilege grant failed.
grant_roles	2—User does not exist.
	7—Role does not exist.
help	1—There is no help available.
	223—Unable to parse the command line correctly.
import_template	21—Occurs if one of the templates has an OMS version specified in it that does not match the version of the OMS you are importing it into, and there are no other errors.
	22—Occurs if one of the template files cannot be parsed, and there are no other errors.
	99—More than one of the templates to be imported had errors during processing.
	223—Unable to parse command line correctly, or an exception was thrown during SQL handling.
	245—There is a problem reading in the file, or it does not exist.
login	0—Verb success exit value.
	1—Cannot establish an OMS connection storage area, or a corrupt area already exists.
	2—A connection with the OMS cannot be established.
	3—The login with the credentials provided failed at the OMS.
	4— The Enterprise Manager license was not accepted by the current user.
	5—The user is already logged in Enterprise Manager.
	223—Command syntax error Verb exit value.
	241—Custom attribute error handling.
	255—Error code for browser-related errors.
logout	0—Verb success exit value.
	1—Cannot establish an OMS connection storage area, or a corrupt area already exists.
	2—A connection with the OMS cannot be established.
	3—The login with the credentials provided failed at the OMS.
	4— The Enterprise Manager license was not accepted by the current user.
	249—OMS connection error verb exit value.
	255—Error code for browser-related errors.
modify_aggregate_service	1—Target does not exist.
	2—Target exists.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
modify_group	1—Group X does not exist.
	2—Cannot add target X to typed group of base type Y.
	3—Group X contains itself as a sub-group at some level.
	219—Current user does not have sufficient privileges to perform this action:
	Current user does not have privilege X over all member targets. Current user does not have sufficient privileges on target X to add it to the group.
	220—Target X does not exist.
	223—Unable to parse command line correctly. Group type is invalid.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
modify_red_group	0—Redundancy Group ""<red_group_name>" modified successfully.
	1—Redundancy Group ""<red_group_name>:<red_group_type>" does not exist.
	2—Cannot add target "<member_target_type>" to typed group of base type "<red_group_type>".
	4—Redundancy Group Type "<red_group_type>" is invalid.
	218—Redundancy Group "<red_group_name>:<red_group_type>" is currently in the process of being deleted.
	220—Target "<member_target_name>:<member_target_type>" does not exist.
	223—Redundancy Group name "<red_group_name>" is not valid. It may contain only alphanumeric characters, multi-byte characters, a space, "-", "_", ".", ":", and have a maximum length of 256 characters.
	223—User name "<owner>" is not valid. It must begin with an alphabetic character, contain only alphanumeric characters, underscores (\"_\"), or periods (\".\"), and have a maximum length of 256 characters.
	223—Invalid value for parameter "add_targets": "<add_targets>". Reason: "<add_targets>" is not a name-value pair.
	223—Member Targets not of same type.
	223—"Generic redundancy group" does not support member of type "<member_target_type>" .

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
modify_role	4—Privilege is invalid or nonexistent.
	5—Target specified in one of the privileges is invalid.
	6—The Super Administrator privilege cannot be granted to a role.
	7—Role does not exist.
	8—Group specified in one of the privileges is invalid.
	9—Job in privilege is invalid or nonexistent.
	10—Cannot have a circular chain of role grants.
	11—The specified user does not exist.
	219—User is unauthorized to perform this action.
	223—Unable to parse command line correctly. Invalid argument value.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
modify_system	0—System "<system_name:system_type>" modified successfully.
	101—System <system_name:system_type> contains itself as a sub-system at some level.
	120—Member target "<member_target_name>:<member_target_type>" does not exist.
	121—System "<system_name:system_type>" does not exist.
	122—Type "<system_type>" is not a valid System type.
	219—Current user does not have sufficient privileges on target <member_target_name> to add it to the system.
	219—Current user does not have sufficient privileges to perform this action.
	223—Invalid value for parameter "add_members": "<add_members>". Reason: "<add_members>" is not a name-value pair.
modify_target	8—One or more of the supplied target properties are invalid.
	15—Target deletion in progress.
	219—Insufficient privileges to modify target.
	220—Target does not exist.
	223—Unable to parse command line correctly.
	File-Fed Option Errors—The errors associated with file-fed options.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
modify_user	1—Target specified in one of the privileges is invalid.
	2—Group specified in one of the privileges is invalid.
	3—Job specified in one of the privileges is invalid.
	4—One of the specified privileges is invalid.
	5—Specified user does not exist.
	6—One or more roles to be granted to the new user does not exist.
	218—A delete is pending against this user until all blackouts and jobs submitted by this user are stopped.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly: Invalid argument value or user name is somehow invalid.
	File-Fed Option Errors—The errors associated with file-fed options.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
provision	1—An Internal error occurred. Could not get an Instance of the Assignment Manager. Exception occurred when getting URN from path.
	2—Could not provision. Exception occurred either in getting editable ProvisioningAssigment object, or during call to Initiate Provisioning.
	3—Could not get one or more URNs. Returned if any of imageUrn, bootServerUrn, stageServerUrn, networkProfileUrn, targetUrn retrieved is null.
	4—Could not create assignment state. Failed to create an AssignmentState object.
	5—Could not set assignment properties. Failed to set the assignment properties in the assignment state object.
	Since this verb uses the FileArgRemoteVerb, the following errors are also possible:
	■ This Verb posts Verb.SYNTAX_ERRNUM if a specified option/file mapping on the command line is not properly formatted.
	■ This Verb posts Verb.LOGIN_SYSTEM_ERRNUM if it cannot log in to the OMS.
	■ This Verb posts Verb.OMS_CONNECTION_SYSTEM_ ERRNUM if it cannot connect to the OMS.
	■ This Verb posts Verb.CONFIGURATION_SYSTEM_ ERRNUM if the configuration files are corrupt or inaccessible.
	■ This Verb posts Verb.MISSING_FILE_SYSTEM_ERRNUM if it cannot find an option value file.
	■ This Verb posts Verb.FILE_READ_SYSTEM_ERRNUM if it cannot read in an option value file.
	■ This Verb posts Verb.FILE_SYNTAX_SYSTEM_ERRNUM.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
relocate_targets	0—Moved all targets from Source Agent to Destination Agent.
	1—Target relocation has failed. The following errors are possible:
	■ SQL exception when relocating targets : <Database-specific error message>.
	■ Communication exception when relocating targets: < communication exception message >.
	■ Verb usage error:

	``
	emcli relocate_targets
	    -src_agent=<source agent target name>
	    -dest_agent=<dest agent target name>
	     {-target_name=<name of the target to be
	relocated>
	    - target_type=<type of the target to be
	relocated>}
	| {-input_file=dupTargets:<complete path to file>}
	    {-force=yes}; "
	``

	■ Errors relocating targets from Source Agent to Destination Agent:
	< error message >
	< error message >
	■ Exception in parsing targets from the command line argument <message>.
remove_beacon	0—Beacon removed successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	173—Beacon does not exist.
	225—Beacon not in monitoring beacons list.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
remove_service_system_assoc	0—System removed from service successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	180—System does not exist.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
retry_job	1—Cannot restart job of a non-restartable type.
	2—Specified job execution does not exist or has not failed.
	3—The specified job execution has already been restarted and failed on restart.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
revoke_roles	2—User does not exist.
	7—Role does not exist.
revoke_privs	2—User does not exist.
	3—Invalid privilege.
	4—One or more targets are invalid.
	5—Invalid globally unique identifier (GUID) privilege.
	6—One or more targets are not groups.
	7—Specified job does not exist.
	8—Privilege grant failed.
set_availability	0—Availability set successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	180—No system defined.
	181—No key tests defined.
	182—No key beacons defined.
	230—Insufficient privileges.
	231—Availability not changed.
	255—Back-end error. Verb failed.
set_credential	1—Target type does not exist.
	2—Target (of given target type) does not exist.
	3—Credential set does not exist.
	4—Insufficient privileges.
	5—Credential column does not exist.
	6—Credential column number mismatch.
set_key_beacons_tests	0—Key beacons and tests set successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	135—Must specify at least one key beacon and test.
	170—Service does not exist.
	173—Beacon does not exist.
	175—Beacon not in list of monitoring beacons.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
set_metric_promotion	0—SUCCESS
	223—SYNTAX_ERRNUM: Input is malformed.
	255—VERB_FAILED_ERRNUM: Back-end validation fails.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
set_properties	0—Properties set successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	132—Invalid property.
	133—Invalid property value.
	170—Service does not exist.
	173—Beacon does not exist.
	175—Beacon not in list of monitoring beacons.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
setup	1—The Verb cannot establish a configuration area, or a corrupt area already exists.
	2—A connection with the OMS cannot be established.
	3—The login with the provided credentials fails at the OMS.
	4—The supplied "url" option is malformed or is not http/https.
	5—The configuration directory is not local as determined by the user in non-trustall HTTPS mode.
	6—The Verb cannot collect the user password safely.
	7—License is not been accepted by the user.
	223—Unable to parse command line correctly.
stop_blackout	1—Blackout X created by user Y does not exist.
	2—The blackout has already ended or stopped.
	3—Agent-side blackouts cannot be edited or stopped.
	218—The start of the blackout is currently being processed.
	The blackout is already pending stop.
	The last set of edits to the blackout have not yet been committed.
	219—You (X) do not have the Super Administrator privilege needed to stop, delete, or modify blackout Y created by user Z.
	Only the blackout owner can stop, delete, or modify the blackout.
	Current user does not have OPERATOR privilege over all blackout targets.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
stop_job	1—Specified job is invalid or non-existent.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
submit_job	1—Supplied job type is invalid or non-existent.
	2—Job with the same name already exists.
	3—One or more specified targets are invalid.
	4—Missing job parameter.
	5—Invalid job parameters, possibly including the security parameters such as "pwd".
	217—Specified job schedule is invalid.
	219—User has insufficient privileges to perform this operation.
	223—Unable to parse command line correctly.
	Invalid argument value.
	File-Fed Option Errors—The errors associated with file-fed options.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
subscribeto_rule	1—Rule with name X and owner Y does not exist.
	2—EM user X does not exist.
	3—EM user X has no email addresses set up (see console tab Preferences->General).
	4—Outgoing Mail (SMTP) Server not set up (see console tab Setup->Notification Methods).
	219—You (X) do not have the SUPER_USER or MANAGE_ ANY_ USER privilege needed to add email addresses for user Y.
	You (X) do not have the SUPER_USER or MANAGE_ANY_ USER privilege needed to subscribe Y to the rule owned by Z.
	223—Unable to parse command line correctly.
	Invalid argument value.
	OMS Connection Errors—The errors associated with connecting to the executing OMS.
sync	1—The Verb cannot establish a configuration area or a corrupt area already exists.
	2—A connection with the OMS cannot be established.
	3—The login with the provided credentials fails at the OMS.
	4—The license has not been accepted by the current user.
	223—Unable to parse the command line correctly.
sync_beacon	0—Beacon synced successfully.
	129—Syntax Error. The displayed message indicates which argument is syntactically incorrect.
	170—Service does not exist.
	173—Beacon does not exist.
	175—Beacon not in list of monitoring beacons.
	230—Insufficient privileges.
	255—Back-end error. Verb failed.
update_audit_settings	223—Syntax error, which could be an invalid directory name or invalid audit settings.

Table 3–4   (Cont.) Built-In Verb Errors

Verb	Error Code
update_db_password	1—Invalid target.
	2—Invalid key value parameter.
	3—Invalid old password.
	4—Invalid privilege.
	223—Syntax error.
update_host_password	1—Invalid target.
	2—Invalid key value parameter.
	3—Invalid old password.
	4—Invalid privilege.
	223—Syntax error.
update_password	4—Target (of given target type) does not exist.
	5—Credential type does not exist for given target.
	6—Key value (that is, user name) does not exist.
	7—Non-operator cannot change credentials.
	8—Wrong value for old password.
	9—Old and new passwords match.
	10—No such non_key_column name.

Index

T

U

V