

☐ This version of the document is in draft state • Publish

CLI API Example for exporting, importing, and deleting different objects using CSV files (v00.17.25)

☐ Like • 2
 ☐ Comment • 0
 ☐

☐ Document created by [Eric Beasley](#)  on Sep 8, 2016 • Last modified by [Eric Beasley](#)  on Jan 5, 2017

☐ Version 8

Overview

The export, import, delete using CSV files scripts in this post, currently version 00.17.25, dated 2017-01-04, are intended to allow operations on an existing R80 or R80.10 Check Point management server (SMS or MDM) from bash on the management server or a management server able authenticate and reach the target management server.

These scripts show examples of:

- an export of objects and services with full and standard json output, and export of hosts, networks, groups, groups-with-exclusion, address-ranges, dns-domains, and group members to csv output [future services dump to csv is pending further research]
- an export of hosts, networks, groups, groups-with-exclusion, address-ranges, dns-domains, and group members to csv output [future services dump to csv is pending further research]
- an import of hosts, networks, groups, groups-with-exclusion, address-ranges, dns-domains, and group members from csv output generated by the export to csv operations above.
- a script to delete groups-with-exclusion, groups, address-ranges, dns-domains, networks, hosts, using csv files created by an object name export to csv for the respective items deleted. **NOTE: DANGER!, DANGER!, DANGER! Use at own risk with extreme care!**

For immediate questions, hit me up at ericb@checkpoint.com

Description

This post includes a set of four (4) script packages, which can be used independently combined. All script files end with .sh for shell and are intended for Check Point bash implementation on R80 and R80.10 or later. Scripts in the packages have specific purposes and some scripts call sub-scripts for extensive repeated operations. The packages also include specific expected default directory folders that are not created by the script action.

The packages are:

- Export Objects : [cli_api_export_objects_ericb_v00x17x25.7z](#)
- Export Specific Objects : [cli_api_export_specific_objects_ericb_v00x17x25.7z](#)

- Import Objects : [cli_api_import_objects_eribc_v00x17x25.7z](#)
- Delete Objects : [cli_api_delete_objects_eribc_v00x17x25.7z](#)

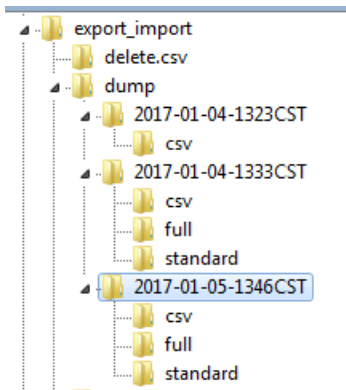
General Information:

<yyyy-mm-dd-hhmm-tz> is a date time group (DTG) generated at time of execution and used for the full operation of the respective script, providing consistent information for a specific script run. Example: 2017-01-05-1346CST for January 5, 2017, at 13:46 hrs CST.

Output Generated by Scripts:

Output from the scripts is directed a sub-folder (default is a dump folder with DTG sub-folder, e.g. ./dump/<yyyy-mm-dd-hhmm-tz>) and further placed in a sub-folder based on script: csv, full, standard, import, delete.

Example:



NOTE: Current CSV output includes additional files used in the process that are raw data, sorted raw data, csv header, and the original data with header.

Export Objects

The Export Objects package provides the capability to export objects and services with full and standard json output, and export of hosts, networks, groups, groups-with-exclusion, address-ranges, dns-domains, and group members to csv output [future services dump to csv is pending further research]. It is composed to two (2) key scripts handling the export of json and/or csv data, with a third script for specific csv export of object names only. The scripts allows for CLI parameter definition of the csv output file location.

The scripts are:

- [cli_api_export_objects_v00x17.sh](#)

Executing this script generates the full dump of all found objects to json, both standard and full output, and to csv.

This script can take some time to operate and provides feedback about what operation is taking place. The csv generation of the group members is quite verbose showing each checked group and how many members are in that group.

- JSON output is generated at standard and full details level for:
 - objects: address-ranges, dns-domains, groups, group members, group-with-exclusions, hosts, networks
 - services: application-site-categories, application-site-groups, application-sites, service-groups, services-dce-rpc, services-other, services-rpc, services-sctp, services-tcp, services-udp
- CSV output is generated for:
 - objects: address-ranges, dns-domains, groups, group-with-exclusions, group members, hosts, networks
- Subscripts used by [cli_api_export_objects_v00x17.sh](#)

- [cli_api_export_objects_actions_v00x17.sh](#) - generates JSON output
- [cli_api_export_objects_actions_to_csv_v00x17.sh](#) - generates CSV output
- Default location for the files generated is:
 - JSON Full Details : `./dump/<yyyy-mm-dd-hhmm-tz>/full/`
 - JSON Standard Details : `./dump/<yyyy-mm-dd-hhmm-tz>/standard/`
 - CSV : `./dump/<yyyy-mm-dd-hhmm-tz>/csv/`
- CSV output files generated by the operation:
 - hosts : `hosts_full_csv.csv`
 - networks : `networks_full_csv.csv`
 - groups : `groups_full_csv.csv`
 - group-with-exclusions : `groups-with-exclusion_full_csv.csv`
 - group members : `group-members_full_csv.csv`
 - address-ranges : `address-ranges_full_csv.csv`
 - dns-domains : `dns-domains_full_csv.csv`
 - group members : `group-members_full_csv.csv`
- Executing this script generates the full dump of all found objects to csv. This script can take some time to operate and provides feedback about what operation is taking place. The csv generation of the group members is quite verbose showing each checked group and how many members are in that group.
 - CSV output is generated for:
 - objects: address-ranges, dns-domains, groups, group-with-exclusions, group members, hosts, networks
 - Default location for the files generated is:
 - CSV : `./dump/<yyyy-mm-dd-hhmm-tz>/csv/`
 - CSV output files generated by the operation:
 - hosts : `hosts_full_csv.csv`
 - networks : `networks_full_csv.csv`
 - groups : `groups_full_csv.csv`
 - group-with-exclusions : `groups-with-exclusion_full_csv.csv`
 - group members : `group-members_full_csv.csv`
 - address-ranges : `address-ranges_full_csv.csv`
 - dns-domains : `dns-domains_full_csv.csv`
 - group members : `group-members_full_csv.csv`
- [cli_api_export_object-names_to_csv_v00x17.sh](#)

Executing this script generates a names only dump of all found objects to csv. This output can be useful for other actions like the delete operation. This script can take some time to operate and provides feedback about what operation is taking place.

 - CSV output is generated for:
 - objects: address-ranges, dns-domains, groups, group-with-exclusions, hosts, networks
 - Default location for the files generated is:
 - CSV : `./dump/<yyyy-mm-dd-hhmm-tz>/csv/`
 - CSV output files generated by the operation:
 - hosts : `hosts_name_csv.csv`
 - networks : `networks_name_csv.csv`
 - groups : `groups_full_name.csv`
 - group-with-exclusions : `groups-with-exclusion_name_csv.csv`

- address-ranges : address-ranges_name_csv.csv
- dns-domains : dns-domains_name_csv.csv
- group members : group-members_name_csv.csv

Export Specific Objects

The Export Specific Objects package provides the capability to export script specific objects like--hosts, networks, groups, groups-with-exclusion, address-ranges, dns-domains, and group members--to csv output [future services dump to csv is pending further research]. It is composed to seven (7) scripts handling the export of csv data. The scripts allows for CLI parameter definition of the csv output file location.

The scripts are:

- [Common to all Scripts](#)
 - Default location for the files generated is:
 - CSV : ./dump/<yyyy-mm-dd-hhmm-tz>/csv/
 - CSV output files generated by the operation:
 - hosts : hosts_full_csv.csv
 - networks : networks_full_csv.csv
 - groups : groups_full_csv.csv
 - group-with-exclusions : groups-with-exclusion_full_csv.csv
 - group members : group-members_full_csv.csv
 - address-ranges : address-ranges_full_csv.csv
 - dns-domains : dns-domains_full_csv.csv
 - group members : group-members_full_csv.csv
- [cli_api_export_object_address-ranges_to_csv_v00x17.sh](#)

Executing this script generates an address-ranges only dump of all found group objects to csv. This script can take some time to operate and provides feedback about what operation is taking place.

 - CSV output is generated for:
 - objects: address-ranges
- [cli_api_export_object_dns-domains_to_csv_v00x17.sh](#)

Executing this script generates a dns-domains only dump of all found group objects to csv. This script can take some time to operate and provides feedback about what operation is taking place.

 - CSV output is generated for:
 - objects: dns-domains
- [cli_api_export_object_groups_to_csv_v00x17.sh](#)

Executing this script generates a groups only dump of all found group objects to csv. This script can take some time to operate and provides feedback about what operation is taking place.

 - CSV output is generated for:
 - objects: groups
- [cli_api_export_object_groups-with-exclusions_to_csv_v00x17.sh](#)

Executing this script generates a group-with-exclusions only dump of all found group objects to csv. This script can take some time to operate and provides feedback about what operation is taking place.

 - CSV output is generated for:
 - objects: group-with-exclusions
- [cli_api_export_object_hosts_to_csv_v00x17.sh](#)

Executing this script generates a hosts only dump of all found group objects to csv. This script can take some time to operate and provides feedback about what operation is taking place.

- CSV output is generated for:
 - objects: hosts
- [cli_api_export_object_networks_to_csv_v00x17.sh](#)
 Executing this script generates a networks only dump of all found group objects to csv. This script can take some time to operate and provides feedback about what operation is taking place.
 - CSV output is generated for:
 - objects: networks
- [cli_api_export_object_group-members_to_csv_v00x17.sh](#)
 Executing this script generates a group member only dump of all found group objects to csv. This script can take some time to operate and provides feedback about what operation is taking place. The csv generation of the group members is quite verbose showing each checked group and how many members are in that group.
 - CSV output is generated for:
 - objects: group members

Import Objects

The Import Objects package provides the capability to import objects using csv export of hosts, networks, groups, groups-with-exclusion, address-ranges, dns-domains, and group members [future services import from csv is pending further research]. It is composed to one (1) key scripts handling the import of csv data and a directory folder ("import.csv") which is the default source location for csv files to import. The scripts allows for CLI parameter definition of the csv import file location.

The scripts are:

- [cli_api_import_objects_from_csv_v00x17.sh](#)
 Executing this script operates the import for the full content all found csv files conforming to the file naming for the export operation. This script can take some time to operate and provides feedback about what operation is taking place.
 - JSON output is generated at full details level for the mgmt_cli add command execution.
 - Default folder location for the expected csv files is "./import.csv"
 - CSV input files expected for operation:
 - hosts : hosts_full_csv.csv
 - networks : networks_full_csv.csv
 - groups : groups_full_csv.csv
 - group-with-exclusions : groups-with-exclusion_full_csv.csv
 - group members : group-members_full_csv.csv
 - address-ranges : address-ranges_full_csv.csv
 - dns-domains : dns-domains_full_csv.csv
 - group members : group-members_full_csv.csv

Delete Objects

The Delete Objects package provides the capability to delete objects using csv export of hosts, networks, groups, groups-with-exclusion, address-ranges, dns-domains, and group members [future services dump to csv is pending further research]. It is composed to two (2) key scripts handling the deletion using csv data and generation of the object name csv file, and a directory folder ("delete.csv") which is the default source location for csv files to import. The scripts allows for CLI parameter definition of the csv delete file location.

NOTE: DANGER!, DANGER!, DANGER! Use at own risk with extreme care!

NOTE: It is possible that some objects intended for the delete are not released by management based on processing, so may still be present after running the script. Check the JSON results output for information.

WHY: testing operations on the scripts or having a flub with data (e.g. the change in color names from R80 to R80.10 API) may require hard clean-up. So this is how I can undo a mess. But it harbors danger!

The scripts are:

- [cli_api_delete_objects_using_csv_v00x17.sh](#)
 Executing this script operates the delete for the full content all found csv files conforming to the file naming for the export operation. This script can take some time to operate and provides feedback about what operation is taking place.
 - JSON output is generated at full details level for the mgmt_cli add command execution.
 - Default folder location for the expected csv files is `"/delete.csv"`
 - CSV input files expected for operation:
 - `hosts : hosts_name_csv.csv`
 - `networks : networks_name_csv.csv`
 - `groups : groups_full_name.csv`
 - `group-with-exclusions : groups-with-exclusion_name_csv.csv`
 - `address-ranges : address-ranges_name_csv.csv`
 - `dns-domains : dns-domains_name_csv.csv`
 - `group members : group-members_name_csv.csv`
- [cli_api_export_object-names_to_csv_v00x17.sh](#)
 Executing this script generates a names only dump of all found objects to csv. This output can be useful for other actions like the delete operation. This script can take some time to operate and provides feedback about what operation is taking place.
 - CSV output is generated for:
 - `objects: address-ranges, dns-domains, groups, group-with-exclusions, hosts, networks`
 - Default location for the files generated is:
 - `CSV : ./dump/<yyyy-mm-dd-hhmm-tz>/csv/`
 - CSV output files generated by the operation:
 - `hosts : hosts_name_csv.csv`
 - `networks : networks_name_csv.csv`
 - `groups : groups_full_name.csv`
 - `group-with-exclusions : groups-with-exclusion_name_csv.csv`
 - `address-ranges : address-ranges_name_csv.csv`
 - `dns-domains : dns-domains_name_csv.csv`
 - `group members : group-members_name_csv.csv`

Instructions

To utilize the scripts, download the scripts from this repository post, extract the script files and directory folders [import and delete actions], then upload those files and directory folders to a working target folder location (e.g. `/var/tmp/api-scripts`) on the target management server where the scripts will execute from. Once uploaded to a working folder the relevant scripts are executed like any other bash script. If executing directly from the folder where the script is located use `"./<script>.sh"` for execution. If script modifications are made outside of Check Point Linux, it is recommended to first run `"dos2unix <script>.sh"` to ensure compatibility with bash shell.

Each script accepts command line parameters to control important inputs that have some defined defaults.

If the "-p <password>" parameter is not used, the user is prompted for the console user/administrators password, as in this example:

```

:0]# ./cli_api_export_objects_v00x17.sh
Date Time Group : 2017-01-05-2225CST

mgmt_cli Login?

Login to mgmt_cli as administrator and save to session file : id.txt

Password: █

```

If the "-r" or "--root" parameter is used then the above prompt should be skipped as in standard mgmt_cli execution.

Command Line Parameters

The scripts all (except actions sub-scripts) can take Command Line Parameters (CLI parameters). To get a dump of the active CLI parameters for a specific script run it with "--help" or "-?". Example:

```

:0]# ./cli_api_export_objects_v00x17.sh --help

cli_api_export_objects_v00x17 [-?][[-r]][-u <admin_name>] [-p <password>]] [-m
<server_IP>] [-d <domain>] [-s <session_file_filepath>] [-o <output_path>] [-i <
import_path>]

Script Version: 00.17.25 Date: 2017-01-04

Standard Command Line Parameters:

Show Help                -? | --help

Authenticate as root     -r | --root
Set Console User Name    -u <admin_name> | --user <admin_name> |
                        -u=<admin_name> | --user=<admin_name>
Set Console User password -p <password> | --password <password> |
                        -p=<password> | --password=<password>
Set Management Server IP -m <server_IP> | --management <server_IP> |
                        -m=<server_IP> | --management=<server_IP>
Set Management Domain    -d <domain> | --domain <domain> |
                        -d=<domain> | --domain=<domain>
Set session file path    -s <session_file_filepath> |
                        -session-file <session_file_filepath> |
                        -s=<session_file_filepath> |
                        -session-file=<session_file_filepath>
Set output file path     -o <output_path> | --output <output_path> |
                        -o=<output_path> | --output=<output_path>
Set import file path     -i <import_path> | --import-path <import_path> |
                        -i=<import_path> | --import-path=<import_path>
Set delete file path     -k <delete_path> | --delete-path <delete_path> |
                        -k=<delete_path> | --delete-path=<delete_path>

session_file_filepath = fully qualified file path for session file
output_path = fully qualified file path for output file
import_path = fully qualified folder path for import files
delete_path = fully qualified folder path for delete files

Example:

]# cli_api_export_objects_v00x17 -u fooAdmin -p voodoo -m 192.168.1.1 -d foovil
le -s "/var/tmp/id.txt" -o "/var/tmp/script_dump.txt" -k "/var/tmp/delete/"

:0]#

```

Command line parameters support multiple input formats as displayed, and can be mixed and matched as needed.

This is the standard help output for cli_api_export_objects_v00x17.sh script, which is the standard baseline for all scripts

in this package release:

```

]# ./cli_api_export_objects_v00x17.sh --help

cli_api_export_objects_v00x17 [-?]|[[-r]|[-u <admin_name>] [-p <password>]] [-m
<server_IP>] [-d <domain>] [-s <session_file_filepath>] [-o <output_path>] [-i
<import_path>]

```

Script Version: 00.17.25 Date: 2017-01-04

Standard Command Line Parameters:

Show Help	-? --help
Authenticate as root	-r --root
Set Console User Name	-u <admin_name> --user <admin_name> -u=<admin_name> --user=<admin_name>
Set Console User password	-p <password> --password <password> -p=<password> --password=<password>
Set Management Server IP	-m <server_IP> --management <server_IP> -m=<server_IP> --management=<server_IP>
Set Management Domain	-d <domain> --domain <domain> -d=<domain> --domain=<domain>
Set session file path	-s <session_file_filepath> -session-file <session_file_filepath> -s=<session_file_filepath> -session-file=<session_file_filepath>
Set output file path	-o <output_path> --output <output_path> -o=<output_path> --output=<output_path>
Set import file path	-i <import_path> --import-path <import_path> -i=<import_path> --import-path=<import_path>
Set delete file path	-k <delete_path> --delete-path <delete_path> -k=<delete_path> --delete-path=<delete_path>

session_file_filepath = fully qualified file path for session file
 output_path = fully qualified file path for output file
 import_path = fully qualified folder path for import files
 delete_path = fully qualified folder path for delete files

Example:

```

]# cli_api_export_objects_v00x17 -u fooAdmin -p voodoo -m 192.168.1.1 -d fooville -s
"/var/tmp/id.txt" -o "/var/tmp/script_dump.txt" -k "/var/tmp/delete/"

```

Standard Command Line Parameters:

Parameter Purpose	Parameter value and options	Default Value	Description
Show Help	-? --help	n/a	Show help for script
Authenticate as root	-r --root	n/a	Instead of using administrator user name and password operate as root
Set Console User Name	-u <admin_name> --user <admin_name> -u=<admin_name> --user=<admin_name>	administrator	Set the username of console user/administrator executing the script. <admin_name> username for console/administrator, e.g. admin
Set Console User password	-p <password> --password <password> -p=<password> --password= <password>	n/a	Set the password to be used for console user/administrator authentication. If not used the default operation will prompt for the console user/administrator password. <password> password to use for console user/administrator. NOTE: Entry is visible when used.
Set Management Server IP	-m <server_IP> --management <server_IP> -m=<server_IP> --management= <server_IP>	localhost	Set the IP address of the management server to use for this operation. <server_IP> is the TCP/IP address of the target management server, e.g. 10.10.100.66
Set Management Domain	-d <domain> --domain <domain> -d=<domain> --domain=<domain>	not set	Set the management domain to use for this operation on a Multi-Domain Management Server <domain> is the domain to use for the operation, e.g. fooville
Set session file path	-s <session_file_filepath> -session-file <session_file_filepath> -s= <session_file_filepath> -session-file= <session_file_filepath>	./id.txt	Set the full path and file name to the session ID file <session_file_filepath> full path to the session ID file, e.g. /var/tmp/id.txt
Set output file path	-o <output_path> --output <output_path> -o=<output_path> --output= <output_path>	./dump/<yyyy-mm-dd-hhmm-tz>	Set the path for output files generated by the script. <output_path> path (no following "/"), e.g. "/var/tmp/script"
Set import file path	-i <import_path> --import-path <import_path> -i=<import_path> --import-path=	./import.csv	Set the path for input files required by the script. <import_path> path (no following "/"), e.g. "/var/tmp/script/input"

	<import_path>		
Set delete file path	-k <delete_path> --delete-path <delete_path> -k=<delete_path> --delete-path=<delete_path>	./delete.csv	Set the path for input files required by the script to identify what to delete. <delete_path> path (no following "/"), e.g. "./var/tmp/script/input"

NOTE: I just noticed I have a semantic error in script version 00.17.25, with the output_path, import_path, and delete_path example in the CLI output with "-?" or "--help" being wrong, since I'm not expecting a fully qualified path with file name just the path sans following "/"). I'll be correcting that soon.

Configuration Parameters in the Script *[this section needs more work]*

NOTE: This predicates some scripting ability and capability to use a text editor. I recommend using the dos2unix command on any updated scripts once uploaded to the target management server host to ensure compatibility.

These script examples attempt to provide some detail tailoring and configuration via variables set for the specific script. Some of these configuration values are influenced by the Command Line Parameters that can be passed to the script. This version does not make the approach overly generic (e.g. name of exported CSV file is hardcode in the import), and future versions of this example set may clearly abstract and configure command line input variables.

Key values to configure for:

Export, Import, and Delete scripts

Variable	Definition
APICLIadmin	SmartConsole administrator name to use for operations
APICLIsessionfile	filename and path to mgmt_cli session ID file generated by login and used for all subsequent

Export Scripts

Variable	Definition
APICLIpatheroot	root of path for output files
APICLIpathbase	base path for output files, generally uses \$APICLIpatheroot and for operations time delineation can utilize the \$DATE variable
APICLIfileoutputpre	General prefix for the output file, prefixes the filename in the full output file path
APICLIfileoutputext	File extension for operational output file, default is .txt

<i>APICLIfileoutputsuffix</i>	File suffix for the operational output file, default is \$DATE. <i>\$APICLIfileoutputtext</i> so generally<date_time_group>.txt
<i>APICLIJSONfileoutputtext</i>	File extension for mgmt_cli json output file, default is .json NOTE: this is not used in this example
<i>APICLIJSONfileoutputsuffix</i>	File suffix for the mgmt_cli json output file, default is \$DATE. <i>\$APICLIJSONfileoutputtext</i> so generally<date_time_group>.json NOTE: this is not used in this example
<i>APICLICSVfileoutputtext</i>	File extension for generated CSV file, default is .csv
<i>APICLICSVfileoutputsuffix</i>	File suffix for the operational output file, default is \$DATE. <i>\$APICLICSVfileoutputtext</i> so generally<date_time_group>.csv NOTE: this was purposely done for the work utilizing this example, which stipulates a defined state of CSV output to export based on the time of execution. For those wanting a generic approach, the value can be set to be more static and not include the \$DATE value element.
<i>APICLIOobjectLimit</i>	This is the maximum number of groups to export, providing the limit value for the mgmt_cli show groups command to populate the array of groups to export members from.
<i>APICLIoutput</i>	full file path to operational output file for later review of actions

Import Scripts

Header 1	Header 2
<i>APICLIfileoutputpre</i>	General prefix for the output file, prefixes the filename in the full output file path
<i>APICLIfileoutputtext</i>	File extension for mgmt_cli json output file, default is .json
<i>APICLIfileoutputsuffix</i>	File suffix for the mgmt_cli json output file, default is \$DATE. <i>\$APICLIfileoutputtext</i> so generally<date_time_group>.json
<i>OutputPathRoot</i>	root of path for output files
<i>OutputPathBase</i>	base path for output files, generally uses <i>\$OutputPathRoot</i> and for operations time delineation can utilize the \$DATE variable
<i>CSVImportType</i>	mgmt_cli type for import operation, in this example it is group
<i>CSVImportPathRoot</i>	This is the path root for the location of the CSV file to import, in the example it is a sub-directory relative to the location of the script
<i>CSVImportPathFile</i>	This is the file name of the CSV file to import, in this case hard-coded based the CSV output generated by the export operation. NOTE: this was purposely done for the work utilizing this example, which stipulates a defined state of CSV output to import. For those wanting a generic approach, the value can be set to be more static and not include the \$DATE value element.
<i>CSVImportPath</i>	This is the path to the CSV file to import based on the <i>\$CSVImportPathRoot</i> and <i>\$CSVImportPathFile</i> variables.
<i>OutputPath</i>	full file path to operational output file for later review of actions

Delete Scripts

Header 1	Header 2
<i>APICLIfileoutputpre</i>	General prefix for the output file, prefixes the filename in the full output file path
<i>APICLIfileoutputtext</i>	File extension for mgmt_cli json output file, default is .json
<i>APICLIfileoutputsufix</i>	File suffix for the mgmt_cli json output file, default is \$DATE. <i>\$APICLIfileoutputtext</i> so generally<date_time_group>.json
<i>OutputPathRoot</i>	root of path for output files
<i>OutputPathBase</i>	base path for output files, generally uses <i>\$OutputPathRoot</i> and for operations time delineation can utilize the \$DATE variable
<i>CSVImportType</i>	mgmt_cli type for import operation, in this example it is group
<i>CSVImportPathRoot</i>	This is the path root for the location of the CSV file to import, in the example it is a sub-directory relative to the location of the script
<i>CSVImportPathFile</i>	<p>This is the file name of the CSV file to import, in this case hard-coded based the CSV output generated by the export operation.</p> <p>NOTE: this was purposely done for the work utilizing this example, which stipulates a defined state of CSV output to import. For those wanting a generic approach, the value can be set to be more static and not include the \$DATE value element.</p>
<i>CSVImportPath</i>	This is the path to the CSV file to import based on the <i>\$CSVImportPathRoot</i> and <i>\$CSVImportPathFile</i> variables.
<i>OutputPath</i>	full file path to operational output file for later review of actions

Modification of the script sections to suit personal preference and requirements is strongly encouraged via the copy-paste operation.

Modification of the script sections to suit personal preference and requirements is strongly encouraged via the copy-paste operation.

I may be updating these later, with some harmonization of common variables required and some abstraction options via command line parameters.

Why and What for...

These scripts were developed to address a pressing need in my own basement cloud laboratory, after some issues cropped up with my migrated management server, which has an original data base starting from R70 and migrated, upgraded, imported to Multi-Domain Management, and now exported from Multi-Domain Management, which has left the system a bit wonky and questionable. By creating scripts to handle the output of objects from my existing management server, I can then use the CSV data to import to a clean, new installation, where I can start fresh, with all my objects, but probably none of the baggage or garbage from almost 9 years of lab/home use operations. It is an excellent learning opportunity and mentors like Uri Bialik help with this very much.

However, these scripts can also help with some other operations that may be necessary, probably requiring some

tweaks, but the example can help a bunch for starting out, operations like:

- Duplicating group members after group import for laboratory environments, when building a from-scratch test environment, but wanting to use familiar objects
- Duplicating group members after group import to a different Domain in Multi-Domain Management, when not wanting to use global objects. This will require some adjustments to handle the Domain specification variables, but should not be rocket magic to make happen.
- Operations in Professional Services for either recovering from an exported baseline, or assisting with pre-migration testing operations, and rebuild operations
- Having a backup of objects on hand to utilize for bare-metal rebuild where a import or restore is not plausible or advisable.

Future Improvements and Extensions:

- **Correct CLI parameter help**
- Implementing and improving error handling, currently omitted due to time constraint and for actual mgmt_cli operations a concept for approach
- Export, Import, and [yes also] Delete of Services, but very much focused on those created by the user, not the native Services delivered in the installation
- Configuration of exported fields to CSV for more control at import
- Interactive selection of what items to export, import, or delete depending on script
- Port to Windows Power Shell scripting, once I figure out how to handled date-time-group values and jq in Windows Power Shell.

Code Version

Code version 0.17.25


Tested on version


R80, API version 1.0
R80.10 EA, API version 1.? [2016-12 EA package]


Change Log


Code version	Key Changes
0.17.25	Updated scripts to include comprehensive Command Line parameter handling (CLI parameters) Added specific scripts for explicit object export to csv. Added delete objects package for clean-up operations Refined operation of scripts to leverage sub routines for repeated operations with extensive parameterization to simplify adding more objects and services Solved the way to pass variable to JQ element in export operation Think I've solved the MDS JQ location problem with older scripts. Providing packaged sets for export, export of specific objects, import, delete, and template shell version 0.5.0


ATTACHMENTS

 cli_api_export_objects_ericeb_v00x17x25.7z.zip
6.1 KB

 cli_api_import_objects_ericeb_v00x17x25.7z.zip
4.3 KB


 cli_api_delete_objects_ericeb_v00x17x25.7z.zip
4.8 KB

 cli_api_export_specific_objects_ericeb_v00x17x25.7z.zip
5.7 KB

 api_mgmt_cli_shell_template_with_cmd_line_parameters.template.v00.05.00.sh
16.8 KB

OUTCOMES

Helpful(1)

Visibility:  [Code Library](#) • 405 Views

Last modified on Jan 5, 2017 8:37 PM






Tags: [export](#) [csv](#) [objects](#) [export-import](#) [import](#) [Edit tags](#)

Categories: [Object Management](#) [Multi Domain](#) [Edit categories](#)






0 Comments

Add a comment

Related Content

-  CLI API Example for exporting group members to a CSV file for later import
-  Check Point API Template base for CLI operations on Gaia
-  CLI API Example json to CSV Export for later use in batch add API commands
-  CP_R80_Gaia_InstallationAndUpgradeGuide.pdf
-  Python tool for exporting/importing a policy package or parts of it

Recommended Content

-  Tags - Simple but powerful. How to use them?
-  ICAP Protocol
-  VMware vRealize Orchestrator (VRO) - Checkpoint package
-  Connecting an R80 smart event to an r77.30 multidomain
-  Using a-synchronous commands (e.g. publish, install-policy and run-script)

