How To Use cp_merge to Manipulate Security Policies

19 December 2010
Important Information

Latest Software
We recommend that you install the most recent software release to stay up-to-date with the latest functional improvements, stability fixes, security enhancements and protection against new and evolving attacks.

Latest Documentation
The latest version of this document is at:
http://supportcontent.checkpoint.com/documentation_download?ID=11783
For additional technical information, visit the Check Point Support Center (http://supportcenter.checkpoint.com).

Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12/16/2010</td>
<td>First release of this document</td>
</tr>
</tbody>
</table>

Feedback
Check Point is engaged in a continuous effort to improve its documentation.

Please help us by sending your comments (mailto:cp_techpub_feedback@checkpoint.com?subject=Feedback on How To Use cp_merge to Manipulate Security Policies ).
Before You Start

Related Documentation and Assumed Knowledge

- Rxx Command Line Interface (CLI) Reference Guide.

Impact on the Environment and Warnings

To apply `cp_merge` tool in a SmartCenter can modify your policy package files and your objects database files affecting directly to your security policy, hence to your security level.

**WARNING:** modifying your policy package with the `cp_merge` tool must be preceded by a backup to avoid risks and potential data loss.

Notes:

- Check Point Management GUI clients may interfere with the `cp_merge` utility and prevent it from changing the repository.
- The import operation fails if the policy uses objects that were deleted after the policy was exported.
- Policy package names are case-sensitive.
- The policy package import is performed one policy package at a time.
- Firewall-1 object names are case sensitive.
- If the name is the same as an existing object the new object cannot be merged.
The cp_merge Command

Check Point Database Merge Tool Usage

```
cp_merge
  merge_objects [-s <db_server>] [-u <user> | -c <cert_file>] [-p <password>] -d <input dir> [-t]
  export_policy [-s <db_server>] [-u <user> | -c <cert_file>] [-p <password>] [-n <pkg name> | -i <policy_name>] [-f <output file>] -d <output dir> [-r]
  delete_policy [-s <db_server>] [-u <user> | -c <cert_file>] [-p <password>] [-n <pkg name>]
  list_policy [-s <db_server>] [-u <user> | -c <cert_file>] [-p <password>] [-n <pkg name>]
```

The cp_merge Command – Help for Detailed Usage
The `cp_merge` Command

Run `cp_merge --help` for detailed usage

- specify database server IP/name
- path to certificate file
- database administrator user name
- user's password
- specify working directory
- print this summary

Objects Merge options:
- test mode – does not save

Policy Export options:
- package name policy package to export
- policy name export policy package which <policy name> belongs to.

Policy Export options:
- package name policy package to export
- policy name export policy package which <policy name> belongs to.
- output file name (default: <policy name>.pol)

Policy Import options:
- input file name specify input file name
- override existing policy if found
- rename policy to <policy name> when importing

Policy Restore options:
- input file name specify input file name
- override existing policy if found
- rename policy to <policy name> when importing

Note: Restore will work only when run locally on management server.

Policy Delete options:
- policy name policy to delete

Delimited Policy Import/Export options:
- export policy

Delimited Policy Import/Export options:
- import new import a new policy
- import override imported policy will replace current
- import append imported policy's rules will be appended to current
- policy name policy to export to/from
- file name file to export to/from
- security | nat | all types of policy to operate on

Note: security policy file is file_name.sec, Net policy file is file_name.net.

[@SMCP211#]
Using cp_merge

In This Section

Listing Security Policies 7
Exporting Security Policies 7
Importing Security Policies 8
Merging Objects 9
Manipulating Policies 10

Listing Security Policies

The cp_merge tool can list all policies included in the database using list_policy option.

Usage: cp_merge list_policy

Options: server and login information

Results: a list with the policy names

Example: Listing policies:

Listing files in that directory will show one file per policy:

Exporting Security Policies

The cp_merge tool can export a security policy in text format.

Usage: cp_merge export_policy

Options: server and login information, input and output information

Results: a security policy in text format

Examples: Exporting all policies: (without parameters, cp_merge export_policy will export all security policies found in the database)

Listing files in that directory will show one file per policy:
Files containing the exported policy are text based:

```plaintext
[Expert@SMCR71]# more Political.pol
{
    :version {
        :major_release_version (9)
        :minor_release_version (7)
        :major_schema_version (6)
        :minor_schema_version (7)
        :cp_version_build (976000027)
    }
    :policies_collection (Political)
    :AdminInfo {
        :ClassName (policies_collection)
        :table (policies_collections)
        :LastModified {
            :Time ("Mon Jun 14 15:08:03 2018")
            :last_modified_utc (1276520083)
            :By (admin)
            :FROM (CPPOE1)
        }
        :icon ("Applications/policy_package")
        :Wiznum (-1)
    }
    :comments ()
    :installable_targets ()
}
```

Exporting only one security policy:

```plaintext
[Expert@SMCR71]# cp_merge export_policy -u admin -p Vpml23 -l Standard -f Standard exported.pol -d c/home/admin
Enter Server name (ENTER for 'localhost'):
Successfully exported policy collection 'Standard'.
```

**Note:** By adding the `-r` parameter you can remove original policy from repository.

---

**Importing Security Policies**

The `cp_merge` tool can import a security policy in text format.

**Usage:**

`cp_merge import_policy`

**Options:**

- server and login information, input (source file with the: `-f` parameter) and output (policy result with the: `-n` parameter) information.

**Results:**

a new security policy into our policy packages

**Examples:** Import policy errors:

```plaintext
[Expert@SMCR71]# cp_merge import_policy -s localhost -u admin -p Vpml2345 -n Political -d .
  -f Political.pol
ERROR: Policy collection 'Political' already exists. Policy import is aborted.
ERROR: Failed to import policy from file 'Political.pol'
[Expert@SMCR71]# cp_merge import_policy -s localhost -u admin -p Vpml2345 -n Political10 -d .
  -f Political.pol
Policy collection 'Political10' is renamed to 'Political'.
ERROR: Failed to update the policies collection object 'Political10' in the database. Probably the database is locked by another client (e.g. SmartDashboard).
ERROR: Failed to import policy from file 'Political1.pol'
```

---

**Using cp_merge**  Page 8
Merging Objects

The `cp_merge` tool can be used to merge objects from a given file into the SmartCenter database.

**Usage:**

```
cp_merge merge_objects
```

**Options:**

- server and login information, input file with objects to merge

**Results:**

- a new objects database including merged objects

**Examples:**

Merging objects failed:

```
[Expert@SMR711]# cp_merge merge_objects
Enter server name (ENTER for 'localhost'):
Enter User Name: admin
Enter User Password:
Failed to open file 'objects_5.0.0': No such file or directory
Error while reading file 'objects_5.0.0' verify that the file exists and is in a set format.
```

Merging objects successfully:

```
[Expert@SMR711]# cp_merge merge_objects -s localhost -u admin -p Upnl2345 -d . -t
```

Checks and warnings:

```
now checking resources
now checking services
now checking times
now checking network_objects
now checking servers
now checking ldap
now checking opsec
now checking products
now checking qos
now checking encryption
now checking communities
now checking methods
now checking software_gw_types
now checking svn
now checking ce_properties
now checking customers
now checking accounting_schemes
now checking secun même
now checking credentials_manager
now checking desktop_profiles
now checking web_authority_must_rules
now checking web_authority_allow_rules
now checking web_authority_effect_rules
```
Manipulating Policies

The cp_merge tool can be used to import/export policies.

Usage: cp_merge delimited_policy

Options: server and login information, input and output information

(you can export a policy, import a policy into a new one, or even import a policy overriding/appending rules)

Results: imports or exports a policy to CSV from CPMI

Note: Rule names are not supported. Works on either NAT or security rules or both

Examples: Appending security and NAT rules:

```
cp_merge delimited_policy <- Command
-s localhost <- SmartCenter Server
-u admin -p Vpn12345 <- user and password
-l Politica2 <- Policy to import
-f Politica2 <- File to import
-a import_append <- import and append rules
-k all <- security and nat rules
```

```[Export@SMCR711] cp_merge delimited_policy: s localhost: u admin: p Vpn12345: l Politica2: f Politica2: a import_append: k all]
The security policy was successfully imported into the SmartCenter Server database.
The NAT policy was successfully imported into the SmartCenter Server database.
```
**Manipulating Policies**

**Using cp_merge**

**Page 11**

**Importing a policy overriding the rules:**

`cp_merge` <- Command
`delimited_policy` <- Option for managing security policies
`-s localhost` <- SmartCenter Server
`-u admin -p Vpn12345` <- user and password
`-l Politica2` <- Policy to import
`-f Politica2` <- File to import
`-a import_override` <- imported rules will override existing ones.
`-k all` <- security and nat rules

```
[Expert@Smcr71]# cp_merge delimited_policy -s localhost -u admin -p Vpn12345 -l Politica2 -f Politica2 -a import_override -k all
The security policy was successfully imported into the SmartCenter Server database.
The NAT policy was successfully imported into the SmartCenter Server database.
```

**Exporting security rules from a policy package to a file:**

```
[Expert@Smcr71]# cp_merge delimited_policy -s localhost -u admin -p Vpn12345 -l Politica2 -f Politica2 -a export_security -k security
The security policy was successfully exported to the "file1.sec" file.
```

**Checking output file:**
# ExpertCpModule

```bash
$ more file1.sec
```

- Total of 122 security rules were exported
- from policy <file>_Cluster_20180723_ID
- by user <admin>
- on server <cpmodule>
- on Mon Aug 9 10:42:38 2010 (GMT)

- Use "!" to negate an object in the Source, Destination and Service fields, e.g. !LocalMachine
- Use "0" for "Users Access" in the Source field, e.g. All Users@Any
- Use "->" for "Service With Resource" in the Service field, e.g. ftp->ftp_resource
- Use "~" to mark "New Line" in the Comment field, e.g. First line~Second line

Valid values for the "Rule Type" field are:
- (1) Rule - Normal rule
- (2) DisabledRule - Disabled normal rule
- (3) SectionTitle - Title of a rules section

```bash
```

```
# RULE TYPE : SOURCE : DESTINATION : VPN : SERVICE : ACTION : TRACK : INSTALL ON :
# TIME : COMMENT :

SectionTitle: Drop rule for blacklist IPs:
Rule: Reg_Block_IPs; Any; Any; drop; None; Any; Any; 
Rule: Any; Reg_Block_IPs; Any; Any; drop; None; Any; Any; 
```
Completing the Procedure

To finish the security policies manipulation you must review manually the security and NAT rules to verify that your policy is still in line with your company security policy.
Verifying

Check CLI outputs and open SmartDashboard and check that you have the desired policies and network objects.

Then review all the objects and policies merged, verify and install the policy and check that all is OK.