Command Line Interface (CLI)

Version NGX R61

For additional technical information about Check Point products, consult Check Point’s SecureKnowledge at

http://support.checkpoint.com/kb/

See the latest version of this document in the User Center at

http://www.checkpoint.com/support/technical/documents

February 2006
© 2003 - 2006 Check Point Software Technologies Ltd.

All rights reserved. This product and related documentation are protected by copyright and distributed under licensing restricting their use, copying, distribution, and decapsulation. No part of this product or related documentation may be reproduced in any form or by any means without prior written authorization of Check Point. While every precaution has been taken in the preparation of this book, Check Point assumes no responsibility for errors or omissions. This publication and features described herein are subject to change without notice.

RESTRICTED RIGHTS LEGEND:
Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 and FAR 52.227-19.

TRADEMARKS:
Check Point Software Technologies Ltd. All rights reserved.
Check Point, Application Intelligence, Check Point Express, the Check Point logo, AlertVigil, ClusterXL, Cooperative Enforcement, ConnectControl, Connectra, CoSa, Coordinated Security Alliance, FireWall-1, FireWall-1 GX, FireWall-1 SecureServer, FloodGate-1, Hacker ID, iMSecure, INSPECT, INSPECT XL, Integrity, InterSpect, IQ Engine, Open Security Extension, OPSEC, Policy Lifecycle Management, Provider-1, Safety@Home, SecureAccess, SecureClient, SecureConnect, SecurityClient, SecurityServer, SecureRemote, SecuServe, SecureUpdate, SecureXL, SiteManager-1, SmartCenter, SmartCenter Pro, Smarter Security, SmartDashboard, SmartDefense, SmartLSM, SmartMap, SmartUpdate, SmartView, SmartView Monitor, SmartView Reporter, SmartView Status, SmartViewTracker, SoftWare, SSL Network Extender, TrueVector, UAM, User-to-Address Mapping, UserAuthority, VPN-1, VPN-1 Accerator Card, VPN-1 Edge, VPN-1 SecureClient, VPN-1 SecureServer, VN-1 VX, Web Intelligence, ZoneAlarm, Zone Alarm Pro, Zone Labs, and the Zone Labs logo, are trademarks or registered trademarks of Check Point Software Technologies Ltd. or its affiliates. All other product names mentioned herein are trademarks or registered trademarks of their respective owners. The products described in this document are protected by U.S. Patent No. 5,006,668, 5,835,726 and 6,496,935 and may be protected by other U.S. Patents, foreign patents, or pending applications.

PARTIES:
Entrust is a registered trademark of Entrust Technologies, Inc. in the United States and other countries. Entrust’s logos and Entrust product and service names are also trademarks of Entrust Technologies, Inc. Entrust Technologies Limited is a wholly owned subsidiary of Entrust Technologies, Inc. FireWall-1 and SecuRemote incorporate certificate management technology from Entrust.

Vestign is a trademark of Vestign Inc.

The following statements refer to those portions of the software copyrighted by the University of Michigan. Portions of the software copyright © 1992-1996 Regents of the University of Michigan. All rights reserved. Redistribution and use in source and binary forms are permitted provided that this notice is preserved and that due credit is given to the University of Michigan at Ann Arbor. The name of the University may not be used to endorse or promote products derived from this software without specific prior written permission. This software is provided "as is" without warranty of any kind, express or implied.

Copyright © Sax Software (terminal emulation only).

The following statements refer to those portions of the software copyrighted by Carnegie Mellon University.

Copyright 1997 by Carnegie Mellon University. All Rights Reserved.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Gnu must not be used in advertising or publicity pertaining to distribution of the software without specific, written permission. Gnu disclaims all warranties with regard to this software, including all implied warranties of merchantability and fitness, in no event shall Gnu be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortuous action, arising out of or in connection with the use or performance of this software.

The following statements refer to those portions of the software copyrighted by The Open Group.

The software is provided "as is", without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose and noninfringement, in no event shall the open group be liable for any claim, damages or other liabilities, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the software or use or other dealings in the software.

The following statements refer to those portions of the software copyrighted by The OpenSSL Project. This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/). This software is provided by the openssl project "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the OpenSSL project or its contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

Check Point Software Technologies Ltd.

U.S. Headquarters: 800 Bridge Parkway, Redwood City, CA 94065, Tel: (650) 628-2000 Fax: (650) 654-4233, info@CheckPoint.com
International Headquarters: 3A Jabotinsky Street, Ramat Gan, 52520, Israel, Tel: 972-3-753 4555 Fax: 972-3-575 9256, http://www.checkpoint.com
Table Of Contents

CLI - Getting Started
  Introduction  9
  General Information  9
    Debugging SmartConsole Clients  9

SmartCenter Commands
  cpca_client 11
    cpca_client create_cert 11
    cpca_client revoke_cert 11
    cpca_client set_mgmt_tools 12
  cpconfig 13
  cplic 13
    cplic check 14
    cplic db_add 14
    cplic db_print 15
    cplic db_rm 16
    cplic del 16
    cplic del <object name> 17
    cplic get 18
    cplic put 18
    cplic put <object name> ... 20
    cplic print 21
    cplic upgrade 21
  cp_merge 23
    cp_merge delete_policy 23
    cp_merge export_policy 24
    cp_merge import_policy|restore_policy 25
    cp_merge list_policy 26
  cppkg 27
    cppkg add 27
    cppkg delete 28
    cppkg get 29
    cppkg getroot 29
    cppkg print 30
cppkg setroot 31
cpridrestart 32
cpridstart 32
cpridstop 32
cprinstall 32
cprinstall boot 33
cprinstall cprestart 33
cprinstall cpstart 33
cprinstall cpstop 34
cprinstall get 34
cprinstall install 35
cprinstall stop 37
cprinstall uninstall 37
cprinstall upgrade 38
cprinstall verify 39
cprinstall verify_upgrade 40
cpstart 41
cpstat 41
cpstop 42
cpwd_admin 43
cpwd_admin start 43
cpwd_admin stop 44
cpwd_admin list 44
cpwd_admin exist 45
cpwd_admin kill 45
cpwd_admin config 45
dbedit 47
dbver 50
dbver create 50
dbver export 50
dbver import 51
dbver print 51
dbver print_all 52
dynamic_objects 52
fw 53
fw ctl 53
fw expdate 55
fw fetch 56
fw fetchlogs 57
fw kill 58
fw lea_notify 58
fw lichosts 59
fw log 59
fw logswitch 62
fw mergefiles 64
fw lslogs 65
fw putkey 67
fw repairlog 68
fw sam 69
fwm 74
fwm dbimport 74
fwm dbexport 76
fwm dbload 78
fw hastat 79
fwm ikecrypt 79
fwm load 80
fwm lock_admin 81
fwm logexport 81
fwm sic_reset 83
fwm unload <targets> 84
fwm ver 84
GeneratorApp 85
inet_alert 86
ldapcmd 89
ldapcompare 90
ldapconvert 91
ldapmodify 94
ldapsearch 96
log_export 97
queryDB_util 100
rs_db_tool 102
sam_alert 103

FireWall-1 Commands
comp_init_policy 105
fw 106
fw isp_link 106
fw monitor 106
fw tab 115
fw stat 116
fw ver 117
svr_webupload_config 118

VPN-1 Commands

**VPN 119**
- vpn accel 119
- vpn compreset 121
- vpn compstat 122
- vpn crl_zap 122
- vpn crlview 122
- vpn debug 123
- vpn drv 124
- vpn export_p12 125
- vpn macutil 125
- vpn nssm_toplogy 126
- vpn overlap_encdom 126
- vpn sw_topology 127
- vpn ver 128
- vpn tu 128

SmartView Monitor Commands

**RTM 131**
- rtm debug 131
- rtm drv 131
- rtm monitor 132
- rtm monitor 135
- rtm rtmd 136
- rtm stat 137
- rtm ver 137
- rtmstart 137
- rtmstop 137

SecureClient Commands

**SCC 139**
- scc connect 139
- scc connectnowait 140
- scc disconnect 140
Table of Contents

scc erasecreds 140
scc listprofiles 140
scc numprofiles 141
scc restartsc 141
scc passcert 141
scc setmode <mode> 141
scc setpolicy 142
scc sp 142
scc startsc 142
scc status 142
scc stopsc 142
scc suppressdialogs 143
scc userpass 143
scc ver 143

ClusterXL Commands

cphaconf 145
cphaprobx 146
cphastart 147
cphastop 147
CLI - Getting Started

In This Chapter

Introduction

General Information

Introduction

This guide documents the Command Line Interface (CLI) commands across different Check Point Products and features. The commands are documented according to the product for which they are used. Commands which are common to more than one product are documented in the SmartCenter chapter.

Within each product chapter, the commands are arranged alphabetically.

For Provider-1/SiteManager-1 CLI commands, see Chapter Twelve of the Provider-1/SiteManager-1 User Guide.

General Information

Debugging SmartConsole Clients

It is possible to obtain debugging information on any of the SmartConsole clients by running these clients in a debug mode. You can save the debug information in a default text file, or you can specify another file in which this information should be saved.

Usage: `<fwpolicy.exe> -d -o <Debug-Output-File-Name.txt>`
Syntax:

<table>
<thead>
<tr>
<th>parameter</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>enter the debug mode. If -o is omitted, debug information is saved into a file with the default name: <code>&lt;ROLE_STR&gt;_debug_output.txt</code>.</td>
</tr>
<tr>
<td>-o</td>
<td>This optional parameter, followed by a file name indicates in which text file debug information should be saved.</td>
</tr>
</tbody>
</table>
SmartCenter Commands

### cpca_client

**Description**
This command and all its derivatives are used to execute operations on the ICA.

**Usage**
cpca_client

### cpca_client create_cert

**Description**
This command prompts the ICA to issue a SIC certificate for the SmartCenter server.

**Usage**
cpca_client [-d] create_cert [-p <ca_port>] -n "CN=<common name>" -f <PKCS12 filename>

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>Debug flag</td>
</tr>
<tr>
<td>-p &lt;ca_port&gt;</td>
<td>Specifies the port which is used to connect to the CA (if the CA was not run from the default port 18209)</td>
</tr>
<tr>
<td>-n &quot;CN=&lt;common name&gt;&quot;</td>
<td>sets the CN</td>
</tr>
<tr>
<td>-f &lt;PKCS12 filename&gt;</td>
<td>specifies the file name where the certificate and keys are saved.</td>
</tr>
</tbody>
</table>

### cpca_client revoke_cert

**Description**
This command is used to revoke a certificate issued by the ICA.

**Usage**
cpca_client [-d] revoke_cert [-p <ca_port>] -n "CN=<common name>"


**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>debug flag</td>
</tr>
<tr>
<td>-p &lt;ca_port&gt;</td>
<td>specifies the port which is used to connect to the CA (if the CA was not run from the default port 18209)</td>
</tr>
<tr>
<td>-n &quot;CN=&lt;common name&gt;&quot;</td>
<td>sets the CN</td>
</tr>
</tbody>
</table>

**cpca_client set_mgmt_tools**

**Description**

This command is used to invoke or terminate the ICA Management Tool.

**Usage**

```
cpca_client [-d] set_mgmt_tools on|off [-p <ca_port>] [-no_ssl] [-a|-u "administrator|user DN" -a|-u "administrator|user DN" ... ]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>debug flag</td>
</tr>
<tr>
<td>set_mgmt_tools on</td>
<td>on - Start the ICA Management tool</td>
</tr>
<tr>
<td>-off</td>
<td>off - Stop the ICA Management tool</td>
</tr>
<tr>
<td>-p &lt;ca_port&gt;</td>
<td>Specifies the port which is used to connect to the CA (if the appropriate service was not run from the default port 18265)</td>
</tr>
<tr>
<td>-no_ssl</td>
<td>Configures the server to use clear http rather than https.</td>
</tr>
<tr>
<td>-a</td>
<td>-u &quot;administrator</td>
</tr>
</tbody>
</table>

**Comments**

Note the following:

1. If the command is ran without -a or -u the list of the permitted users and administrators isn’t changed. The server can be stopped or started with the previously defined permitted users and administrators.

2. If two consecutive start operations are initiated the ICA Management Tool will not respond, unless you change the ssl mode. Once the ssl mode has been modified, the Server can be stopped and restarted.
**Description**  
This command is used to run a Command Line version of the Check Point Configuration Tool. This tool is used to configure/reconfigure a VPN-1Pro installation. The configuration options shown depend on the installed configuration and products. Amongst others, these options include:

- Licenses - modify the necessary Check Point licenses
- Administrators - modify the administrators authorized to connect to the SmartCenter Server via the SmartConsole
- GUI Clients - modify the list of GUI Client machines from which the administrators are authorized to connect to a SmartCenter Server
- Certificate Authority - install the Certificate Authority on the SmartCenter Server in a first-time installation
- Key Hit Session - enter a random seed to be used for cryptographic purposes.
- Secure Internal Communication - set up trust between the module on which this command is being run and the SmartCenter Server
- Fingerprint - display the fingerprint which will be used on first-time launch to verify the identity of the SmartCenter Server being accessed by the SmartConsole. This fingerprint is a text string derived from the SmartCenter Server's certificate.

**Usage**  
cpconfig

**Further Info.**  
See the *Getting Started* Guide and the *SmartCenter* Guide.

**Description**  
This command and all its derivatives relate to the subject of Check Point license management. All `cplic` commands are located in `$CPDIR/bin`. License Management is divided into three types of commands:

- **Local Licensing Commands** are executed on local machines.
- **Remote Licensing Commands** are commands which affect remote machines and are executed on the SmartCenter Server.
- **License Repository Commands** are executed on the SmartCenter Server

**Usage**  
cplic
**cpli check**

**Description**  
Use this command to check whether the license on the local machine will allow a given feature to be used.

**Usage**  

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p &lt;product name&gt;</td>
<td>The product for which license information is requested. For example fw1, netsso.</td>
</tr>
<tr>
<td>-v &lt;product version&gt;</td>
<td>The product version for which license information is requested. For example 4.1, 5.0</td>
</tr>
<tr>
<td>-c count</td>
<td>Count the licenses connected to this feature</td>
</tr>
<tr>
<td>-t &lt;date&gt;</td>
<td>Check license status on future date. Use the format ddmmmyyyy. A given feature may be valid on a given date on one license, but invalid in another.</td>
</tr>
<tr>
<td>-r routers</td>
<td>Check how many routers are allowed. The feature option is not needed.</td>
</tr>
<tr>
<td>-S SRusers</td>
<td>Check how many SecuRemote users are allowed. The feature option is not needed</td>
</tr>
<tr>
<td>&lt;feature&gt;</td>
<td>The &lt;feature&gt; for which license information is requested.</td>
</tr>
</tbody>
</table>

**cpli db_add**

**Description**  
The cpli db_add command is used to add one or more licenses to the license repository on the SmartCenter Server. When local license are added to the license repository, they are automatically attached to its intended Check Point Gateway, central licenses need to undergo the attachment process.

**Usage**  
cpli db_add < -l license-file | host expiration-date signature SKU/features >
**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-l license-file</td>
<td>adds the license(s) from license-file. The following options are NOT needed:</td>
</tr>
<tr>
<td></td>
<td>Host Expiration-Date Signature</td>
</tr>
<tr>
<td></td>
<td>SKU/feature</td>
</tr>
</tbody>
</table>

**Comments**

This command is a License Repository command, it can only be executed on the SmartCenter Server.

Copy/paste the following parameters from the license received from the User Center. More than one license can be added.

- **host** - the target hostname or IP address
- **expiration date** - The license expiration date.
- **signature** - The License signature string. For example:
  
  aa6uwknDe-CE6CRTjhv-zipoVNSnm-z98N7Ck3m (Case sensitive. The hyphens are optional)

<table>
<thead>
<tr>
<th>SKU/features</th>
<th>The SKU of the license summarizes the features included in the license. For example:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPSUITE-EVAL-3DES-vNG</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

If the file 192.168.5.11.lic contains one or more licenses, the command:

```
cplic db_add -l 192.168.5.11.lic
```

will produce output similar to the following:

```
Adding license to database ...
Operation Done
```

---

**cplic db_print**

**Description**

The `cplic db_print` command displays the details of Check Point licenses stored in the license repository on the SmartCenter Server.

**Usage**

```
cplic db_print <object name | -all> [-n noheader] [-x print signatures] [-t type] [-a attached]
```
**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object name</td>
<td>Print only the licenses attached to Object name. Object name is the name of the Check Point Gateway object, as defined in SmartDashboard.</td>
</tr>
<tr>
<td>-all</td>
<td>Print all the licenses in the license repository</td>
</tr>
<tr>
<td>-noheader</td>
<td>Print licenses with no header.</td>
</tr>
<tr>
<td>-x</td>
<td>Print licenses with their signature</td>
</tr>
<tr>
<td>-t or -type</td>
<td>Print licenses with their type: Central or Local.</td>
</tr>
<tr>
<td>-a or -attached</td>
<td>Show which object the license is attached to. Useful if the -all option is specified.</td>
</tr>
</tbody>
</table>

**Comments**

This command is a License Repository command, it can only be executed on the SmartCenter Server.

---

### cplic db_rm

**Description**
The `cplic db_rm` command removes a license from the license repository on the SmartCenter Server. It can be executed ONLY after the license was detached using the `cplic del` command. Once the license has been removed from the repository, it can no longer be used.

**Usage**

`cplic db_rm <signature>`

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signature</td>
<td>The signature string within the license.</td>
</tr>
</tbody>
</table>

**Example**

`cplic db_rm 2f540abb-d3bcb001-7e54513e-kfyigpwn`

**Comments**

This command is a License Repository command, it can only be executed on the SmartCenter Server.

---

### cplic del

**Description**

Use this command to delete a single Check Point license on a host, including unwanted evaluation, expired, and other licenses. This command is used for both local and remote machines.

---

16 CLI Guide
cplic del <object name>  

**Usage**  
```bash  
cplic del [-F <output file>] <signature> <object name>  
```

**Syntax**
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-F &lt;output file&gt;</td>
<td>Send the output to &lt;output file&gt; instead of the screen.</td>
</tr>
<tr>
<td>&lt;signature&gt;</td>
<td>The signature string within the license.</td>
</tr>
</tbody>
</table>

**Description**  
Use this command to detach a Central license from a Check Point Gateway. When this command is executed, the License Repository is automatically updated. The Central license remains in the repository as an unattached license. This command can be executed only on a SmartCenter Server.

**Usage**  
```bash  
cplic del <Object name> [-F outputfile] [-ip dynamic ip] <Signature>  
```

**Syntax**
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object name</td>
<td>The name of the Check Point Gateway object, as defined in SmartDashboard.</td>
</tr>
<tr>
<td>-F outputfile</td>
<td>Divert the output to outputfile rather than to the screen.</td>
</tr>
</tbody>
</table>
| -ip dynamic ip    | Delete the license on the Check Point Gateway with the specified IP address. This parameter is used for deleting a license on a DAIP Check Point Gateway  
  **Note** - If this parameter is used, then `object name` must be a DAIP Module. |
| Signature         | The signature string within the license.                                   |

**Comments**  
This is a Remote Licensing Command which affects remote machines that is executed on the SmartCenter Server.
**cplic get**

**Description**
The `cplic get` command retrieves all licenses from a Check Point Gateway (or from all Check Point Gateways) into the license repository on the SmartCenter Server. Do this to synchronize the repository with the Check Point Gateway(s). When the command is run, all local changes will be updated.

**Usage**
```
cplic get <ipaddr | hostname | -all> [-v41]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipaddr</td>
<td>The IP address of the Check Point Gateway from which licenses are to be retrieved.</td>
</tr>
<tr>
<td>hostname</td>
<td>The name of the Check Point Gateway object (as defined in SmartDashboard) from which licenses are to be retrieved.</td>
</tr>
<tr>
<td>-all</td>
<td>Retrieve licenses from all Check Point Gateways in the managed network.</td>
</tr>
<tr>
<td>-v41</td>
<td>Retrieve version 4.1 licenses from the NF Check Point Gateway. Used to upgrade version 4.1 licenses.</td>
</tr>
</tbody>
</table>

**Example**
If the Check Point Gateway with the object name `caruso` contains four Local licenses, and the license repository contains two other Local licenses, the command: `cplic get caruso` produces output similar to the following:
```
Get retrieved 4 licenses.
Get removed 2 licenses.
```

**Comments**
This is a Remote Licensing Command which affects remote machines that is executed on the SmartCenter Server.

**cplic put**

**Description**
The `cplic put` command is used to install one or more Local licenses on a local machine.

**Usage**
```
cplic put [-o overwrite] [-c check-only] [-s select] [-F <output file>]
[-P Pre-boot] [-k kernel-only] <-l license-file | host expiration date signature SKU/feature>
```

18 CLI Guide
Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-overwrite (or -o)</td>
<td>On a SmartCenter Server this will erase all existing licenses and replace them with the new license(s). On a Check Point Gateway this will erase only Local licenses but not Central licenses, that are installed remotely.</td>
</tr>
<tr>
<td>-check-only (or -c)</td>
<td>Verify the license. Checks if the IP of the license matches the machine, and if the signature is valid</td>
</tr>
<tr>
<td>select (or -s)</td>
<td>Select only the Local licenses whose IP address matches the IP address of the machine.</td>
</tr>
<tr>
<td>-F outputfile</td>
<td>Outputs the result of the command to the designated file rather than to the screen.</td>
</tr>
<tr>
<td>-Preboot (or -p)</td>
<td>Use this option after upgrading to VPN-1/FireWall-1 NG FP2 and before rebooting the machine. Use of this option will prevent certain error messages.</td>
</tr>
<tr>
<td>-kernel-only (or -k)</td>
<td>Push the current valid licenses to the kernel. For Support use only.</td>
</tr>
<tr>
<td>-l license-file</td>
<td>Installs the license(s) in license-file, which can be a multi-license file. The following options are NOT needed: host expiration-date signature SKU/features</td>
</tr>
</tbody>
</table>

Comments

Copy and paste the following parameters from the license received from the User Center.

- **host** - One of the following:
  - **All platforms** - The IP address of the external interface (in dot notation); last part cannot be 0 or 255.
  - **Sun OS4 and Solaris2** - The response to the `hostid` command (beginning with 0x).
  - **HP-UX** - The response to the `uname -i` command (beginning with 0d).
  - **AIX** - The response to the `uname -i` command (beginning with 0d), or the response to the `uname -m` command (beginning and ending with 00).
  - **expiration date** - The license expiration date. Can be never
cplc put <object name> ...

- **signature** - The License signature string. For example:
  aa6uwknDc-CE6Crtjhv-zipoVWSnm-z98N7Ck3m (Case sensitive. The hyphens are optional)

- **SKU/features** - A string listing the SKU and the Certificate Key of the license. The SKU of the license summarizes the features included in the license. For example: CPMP-EVAL-1-3DES-NG CK0123456789ab

**Example**
cplc put -l 215.153.142.130.lic produces output similar to the following:

<table>
<thead>
<tr>
<th>Host</th>
<th>Expiration</th>
<th>SKU</th>
</tr>
</thead>
<tbody>
<tr>
<td>215.153.142.130</td>
<td>26Dec2001</td>
<td>CPMP-EVAL-1-3DES-NG CK0123456789ab</td>
</tr>
</tbody>
</table>

**cplc put <object name> ...**

**Description**
Use the cplc put command to attach one or more central or local license remotely. When this command is executed, the License Repository is also updated.

**Usage**
cplc put <object name> [-ip dynamic ip] [-F outputfile] -l license-file | host expiration-date signature SKU/features >

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object name</td>
<td>The name of the Check Point Gateway object, as defined in SmartDashboard.</td>
</tr>
<tr>
<td>-ip dynamic ip</td>
<td>Install the license on the Check Point Gateway with the specified IP address. This parameter is used for installing a license on a DAIP Check Point Gateway. <strong>NOTE:</strong> If this parameter is used, then object name must be a DAIP Check Point Gateway.</td>
</tr>
<tr>
<td>-F outputfile</td>
<td>Divert the output to outputfile rather than to the screen.</td>
</tr>
<tr>
<td>-l license-file</td>
<td>Installs the license(s) from license-file. The following options are <strong>NOT</strong> needed: Host Expiration-Date Signature SKU/features</td>
</tr>
</tbody>
</table>

**Comments**
This is a Remote Licensing Command which affects remote machines that is executed on the SmartCenter Server.
This is a Copy and paste the following parameters from the license received from the User Center. More than one license can be attached

- **host** - the target hostname or IP address
- **expiration date** - The license expiration date. Can be never
- **signature** - The License signature string. For example:
  
  aa6uwknDc-CReSte0jv-zip0VWSnm-z98N7ck3m (Case sensitive. The hyphens are optional)
- **SKU/features** - A string listing the SKU and the Certificate Key of the license. The SKU of the license summarizes the features included in the license. For example: CPMP-EVAL-1-3DES-NG CX0123456789ab

---

**cplic print**

**Description**
The `cplic print` command (located in `$CPDIR/bin`) prints details of Check Point licenses on the local machine.

**Usage**

```
cplic print [-n noheader] [-x prints signatures] [-t type] [-F <outputfile>] [-p preatures]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-noheader</code> (or <code>-n</code>)</td>
<td>Print licenses with no header.</td>
</tr>
<tr>
<td><code>-x</code></td>
<td>Print licenses with their signature</td>
</tr>
<tr>
<td><code>-type</code> (or <code>-t</code>)</td>
<td>Prints licenses showing their type: Central or Local.</td>
</tr>
<tr>
<td><code>-F &lt;outputfile&gt;</code></td>
<td>Divert the output to <code>outputfile</code>.</td>
</tr>
<tr>
<td><code>-p reatures</code> (or <code>-p</code>)</td>
<td>Print licenses resolved to primitive features.</td>
</tr>
</tbody>
</table>

**Comments**
On a Check Point Gateway, this command will print all licenses that are installed on the local machine — both Local and Central licenses.

---

**cplic upgrade**

**Description**
Use the `cplic upgrade` command to upgrade licenses in the license repository using licenses in a license file obtained from the User Center.

**Usage**

```
cplic upgrade <–l inputfile>
```
The following example explains the procedure which needs to take place in order to upgrade the licenses in the license repository.

- Upgrade the SmartCenter Server to the latest version.
  Ensure that there is connectivity between the SmartCenter Server and the remote workstations with the version 4.1 products.
- Import all licenses into the License Repository. This can also be done after upgrading the products on the remote workstations to NG
- Run the command: `cplic get -all`. For example

```
Getting licenses from all modules ...
count:root(su) [-] # cplic get -all
golda: Retrieved 1 licenses. Detached 0 licenses. Removed 0 licenses. count: Retrieved 1 licenses. Detached 0 licenses. Removed 0 licenses.
```

- To see all the licenses in the repository, run the command:
  `cplic db_print -all -a`

```
count:root(su) [-] # cplic db_print -all -a
Retrieving license information from database ...
The following licenses appear in the database:
==================================================================
<table>
<thead>
<tr>
<th>Host</th>
<th>Expiration</th>
<th>Features</th>
<th>Check Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.8.11</td>
<td>Never</td>
<td>CPFW-FIG-25-41</td>
<td>CK-49C3A3CC7121 golda</td>
</tr>
<tr>
<td>192.168.5.11</td>
<td>26Nov2002</td>
<td>CPSUITE-EVAL-3DES-NG</td>
<td>CK-1234567890 count</td>
</tr>
</tbody>
</table>
```

- Upgrade the version 4.1 products on the remote Check Point Gateways.
- In the User Center (http://www.checkpoint.com/usercenter), view the licenses for the products that were upgraded from version 4.1 to NG and create new upgraded licenses.
Download a file containing the upgraded NG licenses. Only download licenses for the products that were upgraded from version 4.1 to NG.

If you did not import the version 4.1 licenses into the repository in step 4, import the version 4.1 licenses now using the command `cplic get -all -v41`

Run the license upgrade command: `cplic upgrade -l <inputfile>
- The licenses in the downloaded license file and in the license repository are compared.
- If the certificate keys and features match, the old licenses in the repository and in the remote workstations are updated with the new licenses.
- A report of the results of the license upgrade is printed.

In the following example, there are two NG licenses in the file. One does not match any license on a remote workstation, the other matches a version 4.1 license on a remote workstation that should be upgraded:

Comments: This is a Remote Licensing Command which affects remote machines that is executed on the SmartCenter Server.

Further Info. See the SmartUpdate chapter of the SmartCenter Guide.

---

cp_merge

Description: The cp_merge utility has two main functionalities
- Export and import of policy packages
- Merge of objects from a given file into SmartCenter database

Usage: `cp_merge help`

Syntax:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>help</td>
<td>Displays the usage for cp_merge.</td>
</tr>
</tbody>
</table>

---

cp_merge delete_policy

Description: This command provides the options of deleting an existing policy package. Note that the default policy can be deleted by delete action.

Usage: `cp_merge delete_policy [-s <db server>] [-u <user> | -c <certificate file>] [-p <password>] -n <package name>`
**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s &lt;db server&gt;</td>
<td>Specify the database server IP Address or DNS name.²</td>
</tr>
<tr>
<td>-u &lt;user&gt;</td>
<td>The administrator’s name.¹,²</td>
</tr>
<tr>
<td>-c &lt;certificate file&gt;</td>
<td>The path to the certificate file.¹</td>
</tr>
<tr>
<td>-p &lt;password&gt;</td>
<td>The administrator’s password.¹</td>
</tr>
<tr>
<td>-n &lt;policy package name&gt;</td>
<td>The policy package to export.²,³</td>
</tr>
</tbody>
</table>

**Comments**

Further considerations:

1. Either use certificate file or user and password
2. Optional

**Example**

Delete the policy package called standard.

```bash
cp_merge delete_policy -n Standard
```

---

**cp_merge export_policy**

**Description**

This command provides the options of leaving the policy package in the active repository, or deleting it as part of the export process. The default policy cannot be deleted during the export action.

**Usage**

```bash
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s &lt;db server&gt;</td>
<td>Specify the database server IP Address or DNS name.²</td>
</tr>
<tr>
<td>-u &lt;user&gt;</td>
<td>The database administrator’s name.¹</td>
</tr>
<tr>
<td>-c &lt;certificate file&gt;</td>
<td>The path to the certificate file.¹</td>
</tr>
<tr>
<td>-p &lt;password&gt;</td>
<td>The administrator’s password.¹</td>
</tr>
<tr>
<td>-n &lt;policy package name&gt;</td>
<td>The policy package to export.²,³</td>
</tr>
<tr>
<td>-l &lt;policy name&gt;</td>
<td>Export the policy package which encloses the policy name.²,³,⁴</td>
</tr>
</tbody>
</table>
SmartCenter Commands

### cp_merge import_policy|restore_policy

**Description**
This command provides the options to overwrite an existing policy package with the same name, or preventing overwriting when the same policy name already exists.

**Usage**
```
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d &lt;output directory&gt;</td>
<td>Specify the output directory.</td>
</tr>
<tr>
<td>-f &lt;outputfile&gt;</td>
<td>Specify the output file name (where the default file name is <code>&lt;policy name&gt;.pol</code>).</td>
</tr>
<tr>
<td>-r</td>
<td>Remove the original policy from the repository.</td>
</tr>
</tbody>
</table>

**Comments**
Further considerations:
1. Either use certificate file or user and password
2. Optional
3. If both -n and -l are omitted all policy packages are exported.
4. If both -n and -l are present -l is ignored.

**Example**
Export policy package Standard to file
```
```

---

**cp_merge import_policy|restore_policy**

**Description**
This command provides the options to overwrite an existing policy package with the same name, or preventing overwriting when the same policy name already exists.

**Usage**
```
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s &lt;db server&gt;</td>
<td>Specify the database server IP Address or DNS name.</td>
</tr>
<tr>
<td>-u &lt;user&gt;</td>
<td>The administrator’s name.</td>
</tr>
<tr>
<td>-c &lt;certificate file&gt;</td>
<td>The path to the certificate file.</td>
</tr>
<tr>
<td>-p &lt;password&gt;</td>
<td>The administrator’s password.</td>
</tr>
<tr>
<td>-n &lt;policy package name&gt;</td>
<td>Rename the policy package to <code>&lt;policy package name&gt;</code> when importing.</td>
</tr>
</tbody>
</table>

---

SmartCenter Commands 25
### cp_merge list_policy

**Usage**

```plaintext
cp_merge list_policy [-s <db server>] [-u <user> | -c <certificate file>] [-p <password>]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s &lt;db server&gt;</td>
<td>Specify the database server IP Address or DNS name.</td>
</tr>
<tr>
<td>-u &lt;user&gt;</td>
<td>The administrator's name.</td>
</tr>
<tr>
<td>-c &lt;certificate file&gt;</td>
<td>The path to the certificate file.</td>
</tr>
<tr>
<td>-p &lt;password&gt;</td>
<td>The administrator's password.</td>
</tr>
</tbody>
</table>

**Comments**

Further considerations:

1. Either use certificate file or user and password
2. Optional

**Example**

List all policy packages which reside in the specified repository:

```plaintext
cp_merge list_policy
```

---

### cp_merge import_policy

**Usage**

```plaintext
cp_merge import_policy -f Standard.pol -n StandardCopy
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f &lt;inputfile&gt;</td>
<td>Specify the input file name.</td>
</tr>
<tr>
<td>-n &lt;inputfile&gt;</td>
<td>Override an existing policy if found.</td>
</tr>
</tbody>
</table>

**Comments**

Further considerations:

1. Either use certificate file or user and password
2. Optional

**Example**

Import the policy package saved in file `Standard.pol` into the repository and rename it to `StandardCopy`.

```plaintext
cp_merge import_policy -f Standard.pol -n StandardCopy
```
cppkg add

cp_merge list -s localhost

cppkg

Description This command is used to manage the product repository. It is always executed on the SmartCenter Server.

cppkg add

Description The cppkg add command is used to add a product package to the Product Repository. Only SmarUpdate packages can be added to the Product Repository.

Products can be added to the Repository as described in the following procedures, by importing a file downloaded from the Download Center web site at http://www.checkpoint.com/techsupport/downloads/downloads.html. The package file can be added to the Repository directly from the CD or from a local or network drive.

Usage cppkg add <package-full-path | CD drive>

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>package-full-path</td>
<td>If the package to be added to the repository is on a local disk or network drive, type the full path to the package.</td>
</tr>
<tr>
<td>CD drive</td>
<td>If the package to be added to the repository is on a CD:</td>
</tr>
<tr>
<td></td>
<td>For Windows machines type the CD drive letter, e.g.</td>
</tr>
<tr>
<td></td>
<td>d:\</td>
</tr>
<tr>
<td></td>
<td>For UNIX machines, type the CD root path, e.g.</td>
</tr>
<tr>
<td></td>
<td>/caruso/image/CPsuite-NG/FP2</td>
</tr>
<tr>
<td></td>
<td>You will be asked to specify the product and appropriate Operating System (OS).</td>
</tr>
</tbody>
</table>

Comments cppkg add does not overwrite existing packages. To overwrite existing packages, you must first delete existing packages.

Example [d:\winnt\fw1\ng\bin]cppkg add 1:\CPsuite-NG_FP2\n
Enter package name:
cppkg delete

----------------------
(1) SVNfoundation
(2) firewall
(3) floodgate
(4) rtm

(e) Exit
Enter you choice : 1
Enter package OS :
----------------------
(1) win32
(2) solaris
(3) linux
(4) hpux
(5) ipso
(6) aix

(e) Exit
Enter your choice : 1
You choose to add 'SVNfoundation' for 'win32' OS. Is this correct? [y/n] : y
Adding package from CD ...
Package added to repository.

cppkg delete

**Description**  The command is used to delete a product package from the repository. To delete a product package you must specify a number of options. To see the format of the options and to view the contents of the Product Repository, use the cppkg print command.

**Usage**  cppkg delete [vendor product version os [sp]]
cppkg get

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vendor</td>
<td>Package vendor (e.g. checkpoint).</td>
</tr>
<tr>
<td>product</td>
<td>Package name Options are: SVNfoundation, firewall, floodgate.</td>
</tr>
<tr>
<td>version</td>
<td>Package version (e.g. NG).</td>
</tr>
<tr>
<td>sp</td>
<td>Package service pack (e.g. fcs for NG R54 initial release, FP1, FP2 etc.) This parameter is optional. Its default is fcs.</td>
</tr>
</tbody>
</table>

Comments

It is not possible to undo the cppkg del command.

Example

[d: \winnt\fw1\ng\bin]cppkg del

Getting information from package repository. Please wait...

Select package:

-----------------------
(1) checkpoint SVNfoundation NG win32 FCS_FP1
(2) checkpoint SNVfoundation NG win32 FP1
(e) Exit

Enter your choice : 2

You choose to delete 'checkpoint SVNfoundation NG win32 FP1' Is this correct? [y/n] : y

Package removed from repository.

cppkg get

Description

This command synchronizes the Package Repository database with the content of the actual package repository under $SUROOT.

Usage

cppkg get

cppkg getroot

Description

The command is used to find out the location of the Product Repository. The default Product Repository location on Windows machines is C:\SUroot. On UNIX it is /var/SUroot.
cppkg print

Usage

cppkg getroot

Example

# cppkg getroot
Current repository root is set to : /var/suroot/

cppkg print

Description
The command is used to list the contents of the Product Repository.

Use cppkg print to see the product and OS strings required to install a product package using the cprinstall command, or to delete a package using the cppkg delete command.

Usage

cppkg print
cppkg setroot

**Description**

The command is used to create a new repository root directory location, and to move existing product packages into the new repository.

The default Product Repository location is created when the SmartCenter Server is installed. On Windows machines the default location is `C:\SUroot` and on UNIX it is `/var/SUroot`. Use this command to change the default location.

When changing repository root directory:

- The contents of the old repository is copied into the new repository.
- The `$SUROOT` environment variable gets the value of the new root path.
- A product package in the new location will be overwritten by a package in the old location, if the packages are the same (that is, they have the same ID strings).

The repository root directory should have at least 200 Mbyte of free disk space.

**Usage**

```
cppkg setroot <repository-root-directory-full-path>
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>repository-root-directory-full-path</td>
<td>The desired location for the Product Repository.</td>
</tr>
</tbody>
</table>

**Comments**

It is important to reboot the SmartCenter Server after performing this command, in order to set the new `$SUROOT` environment variable.

**Example**

```
# cppkg setroot /var/new_suroot
Repository root is set to : /var/new_suroot/
```

Note: When changing repository root directory:

![cppkg setroot example](d:\winnt\fw1\ng\bin)cppkg print

Getting information from package repository. Please wait ...

<table>
<thead>
<tr>
<th>Vendor</th>
<th>Product</th>
<th>Version</th>
<th>OS</th>
<th>SP</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkpoint</td>
<td>SVNfoundation NG</td>
<td>win32</td>
<td>FCS_FP1</td>
<td>SVN foundation</td>
<td>NG Feature Pack 1 for 4.1 upgrade</td>
</tr>
<tr>
<td>checkpoint</td>
<td>SVNfoundation NG</td>
<td>win32</td>
<td>FP1</td>
<td>SVN foundation</td>
<td>Feature Pack 1 for NG upgrade</td>
</tr>
</tbody>
</table>
1. Old repository content will be copied into the new repository.
2. A package in the new location will be overwritten by a package in the old location, if the packages have the same name.

Change the current repository root? [y/n]: y
The new repository directory does not exist. Create it? [y/n]: y
Repository root was set to: /var/new_root
Notice: To complete the setting of your directory, reboot the machine!

---

cpridrestart

**Description**  
Stops and starts the Check Point Remote installation Daemon (cprid). This is the daemon that is used for remote upgrade and installation of products. It is part of the SVN Foundation. In Windows it is a service.

**Usage**  
cpridrestart

---

cpridstart

**Description**  
Start the Check Point Remote installation Daemon (cprid). This is the service that allows for the remote upgrade and installation of products. It is part of the SVN Foundation. In Windows it is a service.

**Usage**  
cpridstart

---

cpridstop

**Description**  
Stop the Check Point Remote installation Daemon (cprid). This is the service that allows for the remote upgrade and installation of products. It is part of the SVN Foundation. In Windows it is a service.

**Usage**  
cpridstop

---

cprinstall

**Description**  
Use cprinstall commands to perform remote installation of product packages, and associated operations.

On the SmartCenter Server, cprinstall commands require licenses for SmartUpdate.

On the remote Check Point Gateways the following are required:
Trust must be established between the SmartCenter Server and the Check Point Gateway.

- cpd must run.
- cprid remote installation daemon must run. cprid is available on VPN-1/FireWall-1 4.1 SP2 and higher, and as part of SVN Foundation for NG and higher.

### cprinstall boot

**Description**
The command is used to boot the remote computer.

**Usage**
cprinstall boot <object name>

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object name</td>
<td>Object name of the Check Point Gateway defined in SmartDashboard.</td>
</tr>
</tbody>
</table>

**Example**

```
# cprinstall boot harlin
```

### cprinstall cprestart

**Description**
This command enables cprestart to be run remotely.

All products on the Check Point Gateway must be of the same version of NG.

**Usage**
cprinstall cprestart <object name>

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object name</td>
<td>Object name of the Check Point Gateway defined in SmartDashboard.</td>
</tr>
</tbody>
</table>

### cprinstall cpstart

**Description**
This command enables cpstart to be run remotely.

All products on the Check Point Gateway must be of the same version of NG.

**Usage**
cprinstall cpstart <object name>
cprinstall cpstop

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object name</td>
<td>Object name of the Check Point Gateway defined in SmartDashboard.</td>
</tr>
</tbody>
</table>

**Description**

This command enables *cpstop* to be run remotely.

All products on the Check Point Gateway must be of the same version of NG.

**Usage**

`cprinstall cpstop <-proc | -nopolicy> <object name>`

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object name</td>
<td>Object name of the Check Point Gateway defined in SmartDashboard.</td>
</tr>
<tr>
<td>-proc</td>
<td>Kills Check Point daemons and Security Servers while maintaining the active Security Policy running in the kernel. Rules with generic allow/reject/drop rules, based on services continue to work.</td>
</tr>
<tr>
<td>-nopolicy</td>
<td></td>
</tr>
</tbody>
</table>

**cprinstall get**

**Description**

The `cprinstall get` command is used to obtain details of the products and the Operating System installed on the specified Check Point Gateway, and to update the database.

**Usage**

`cprinstall get <Object name>`

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object name</td>
<td>Object name of the Check Point Gateway defined in SmartDashboard.</td>
</tr>
</tbody>
</table>
cprinstall install

Example

[c:\winnt\fw1\5.0\bin]cprinstall get fred

Getting information from fred...

Operating system    Version    SP
----------------------

solaris             5.7       fcs

Vendor    Product    Version    SP
----------------------

CheckPoint    VPN-1 Pro    NG    fcs
CheckPoint    SVNfoundation    NG    fcs

Description

The cprinstall install command is used to install Check Point products on remote Check Point Gateways. To install a product package you must specify a number of options. Use the cppkg print command and copy the required options.

Usage

cprinstall install [-boot] <Object name> <vendor> <product> <version> [sp]

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-boot</td>
<td>Boot the remote computer after installing the package. Only boot after ALL products have the same version, either NG or NG FP1. Boot will be cancelled in certain scenarios. See the Release Notes for details.</td>
</tr>
<tr>
<td>Object name</td>
<td>Object name of the Check Point Gateway defined in SmartDashboard.</td>
</tr>
<tr>
<td>vendor</td>
<td>Package vendor (e.g. checkpoint)</td>
</tr>
</tbody>
</table>
Before transferring any files, this command runs the cprinstall verify command to verify that the Operating System is appropriate and that the product is compatible with previously installed products.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>product</td>
<td>Package name&lt;br&gt;Options are: SVNfoundation, firewall, floodgate.</td>
</tr>
<tr>
<td>version</td>
<td>Package version (e.g. NG FP2)</td>
</tr>
<tr>
<td>sp</td>
<td>Package service pack (e.g. fcs for NG FP2 initial release, FP1 for NG Feature Pack 1.)</td>
</tr>
</tbody>
</table>
Example

```bash
# cprinstall install -boot fred checkpoint firewall NG FP1

Installing firewall NG FP1 on fred...
Info : Testing Check Point Gateway
Info : Test completed successfully.
Info : Transferring Package to Check Point Gateway
Info : Installing package on Check Point Gateway
Info : Product was successfully applied.
Info : Rebooting the Check Point Gateway
Info : Checking boot status
Info : Reboot completed successfully.
Info : Checking Check Point Gateway
Info : Operation completed successfully.
```

cprinstall stop

**Description**

This command is used to stop the operation of other cprinstall commands. In particular, this command stops the remote installation of a product - even during transfer of files, file extraction, and pre-installation verification. The operation can be stopped at any time up to the actual installation.

cprinstall stop can be run from one command prompt to stop a running operation at another command prompt.

**Usage**

`cprinstall stop <Object name>`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object name</td>
<td>Object name of the Check Point Gateway, defined in SmartDashboard.</td>
</tr>
</tbody>
</table>

**Example**

```
[c:\winnt\fw1\5.0\bin]  cprinstall stop Check Point Gateway01
Info : Stop request sent
```

cprinstall uninstall

**Description**

The cprinstall uninstall command is used to uninstall products on remote Check Point Gateways. To uninstall a product package you must specify a number of options. Use the cppkg print command and copy the required options.

**Usage**

`cprinstall uninstall [-boot] <Object name> <vendor> <product> <version> [sp]`
**cprinstall upgrade**

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-boot</td>
<td>Boot the remote computer after installing the package. Only boot after ALL products have the same version, either NG or NG FP1. Boot will be cancelled in certain scenarios. See the Release Notes for details.</td>
</tr>
<tr>
<td>Object name</td>
<td>Object name of the Check Point Gateway defined in SmartDashboard.</td>
</tr>
<tr>
<td>vendor</td>
<td>Package vendor (e.g. checkpoint)</td>
</tr>
<tr>
<td>product</td>
<td>Package name Options are: SVNfoundation, firewall, floodgate.</td>
</tr>
<tr>
<td>version</td>
<td>Package version (e.g. NG FP2)</td>
</tr>
<tr>
<td>sp</td>
<td>Package service pack (e.g. fcs for NG FP2 initial release, FP1 for NG Feature Pack 1.)</td>
</tr>
</tbody>
</table>

**Comments**

*Before* uninstalling any files, this command runs the `cprinstall verify` command to verify that the Operating System is appropriate and that the product is installed.

*After* uninstalling, retrieve the Check Point Gateway data by running `cprinstall get`.

**Example**

```
# cprinstall uninstall fred checkpoint firewall NG FP1

Uninstalling firewall NG FP1 from fred...
Info : Removing package from Check Point Gateway
Info : Product was successfully applied.
Operation Success. Please get network object data to complete the operation.
```

---

**cprinstall upgrade**

**Description**

Use the `cprinstall upgrade` command to upgrade all products on a Check Point Gateway to the latest version.

All products on the Check Point Gateway must be of the same version of NG.
Usage

cprinstall upgrade [-boot] <object name>

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-boot</td>
<td>Boot the remote Check Point Gateway after completing the remote installation.</td>
</tr>
<tr>
<td>object name</td>
<td>Object name of the Check Point Gateway, defined in SmartDashboard.</td>
</tr>
</tbody>
</table>

Comments

When cprinstall upgrade is run, the command first verifies which products are installed on the Check Point Gateway, and that there is a matching product package in the Product Repository with the same OS, and then installs the product package on the remote Check Point Gateway.

cprinstall verify

Description

The cprinstall verify command is used to verify:

- If a specific product can be installed on the remote Check Point Gateway.
- That the Operating System and currently installed products are appropriate for the package.
- That there is enough disk space to install the product.
- That there is a CPRID connection.

Usage

cprinstall verify <Object name> <vendor> <product> <version> [sp]

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object name</td>
<td>Object name of the Check Point Gateway defined in SmartDashboard.</td>
</tr>
<tr>
<td>vendor</td>
<td>Package vendor (e.g. checkpoint).</td>
</tr>
<tr>
<td>product</td>
<td>Package name Options are: SVNFoundation, firewall, floodgate.</td>
</tr>
<tr>
<td>version</td>
<td>Package version (e.g. NG).</td>
</tr>
<tr>
<td>sp</td>
<td>Package service pack (e.g. fcs for NG with Application Intelligence initial release, FP1, FP2 etc.) This parameter is optional. Its default is fcs.</td>
</tr>
</tbody>
</table>
cprinstall verify_upgrade

Example
The following examples show a successful and a failed verify operation:

Verify succeeds:

```
cprinstall verify harlin checkpoint SVNfoundation NG_FP4
```

Verifying installation of SVNfoundation NG FP4 on harlin...
Info : Testing Check Point Gateway.
Info : Test completed successfully.
Info : Installation Verified, The product can be installed.

Verify fails:

```
cprinstall verify harlin checkpoint SVNfoundation NG FCS_FP4
```

Verifying installation of SVNfoundation NG FCS_FP4 on harlin...
Info : Testing Check Point Gateway
Info : SVN Foundation NG is already installed on 192.168.5.134
Operation Success.Product cannot be installed, did not pass dependency check.

cprinstall verify_upgrade

Description
Use the cprinstall verify_upgrade command to verify the success of the upgrade of all products on a Check Point Gateway to the latest version, before performing the upgrade. This command is automatically performed by the cprinstall upgrade command.

All products on the Check Point Gateway must be of the same version of NG.

Usage
cprinstall verify_upgrade <object name>

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object name</td>
<td>Object name of the Check Point Gateway, defined in SmartDashboard.</td>
</tr>
</tbody>
</table>

Comments
When the command is run, the command verifies which products are installed on the Check Point Gateway, and that there is a matching product package in the Product Repository with the same OS.
**cpstart**

**Description**  This command is used to start all Check Point processes and applications running on a machine.

**Usage**  

cpstart

**Comments**  This command cannot be used to start cpriD. cprid is invoked when the machine is booted and it runs independently.

**cpstat**

**Description**  cpstat displays the status of Check Point applications, either on the local machine or on another machine, in various formats.

**Usage**  

cpstat [-h host][-p port][-f flavour][-d] application_flag

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-h host</td>
<td>A resolvable hostname, or a dot-notation address (for example,192.168.33.23). The default is localhost.</td>
</tr>
<tr>
<td>-p port</td>
<td>Port number of the AMON server. The default is the standard AMON port (18192)</td>
</tr>
<tr>
<td>-f flavour</td>
<td>The flavor of the output (as appears in the configuration file). The default is to use the first flavor found in configuration file.</td>
</tr>
<tr>
<td>-d</td>
<td>debug flag</td>
</tr>
</tbody>
</table>

**Application_flag**  One of:

- fwm — FireWall-1
- vpn — VPN-1
- fg — FloodGate-1
- ha — ClusterXL (High Availability)
- os — SVN Foundation and OS Status
- mg — for SmartCenter

Where the flavors are:

- fwm — "fw", with flavours: "default", "all", "policy",
  "performance", "hmem", "hmem", "inspect", "cookies", "chains",
  "fragments", "totals", "ufp_caching", "http_stat", "ftp_stat",
  "telnet_stat", "rlogin_stat", "ufp_stat", "smtp_stat"

SmartCenter Commands  41
cpinstall verify_upgrade

- vpn — "product", "general", "IKE", "ipsec", "fwz", "accelerator", "all"
- fg — "all"
- mg — "default"
- os — "default", "routing"
- ha — "default", "all"

Example

```
> cpstat fw
Policy name: Standard
Install time: Wed Nov 1 15:25:03 2000
Interface table
|Name|Dir|Total *|Accept**|Deny|Log|
-----------------------------------------
|hme0|in |739041*|738990**|51 *|7**|
|hme0|out|463525*|463525**| 0 *|0**|
|*******|1202566|1202515*|51**|7**|
```

cpstop

**Description**  This command is used to terminate all Check Point processes and applications, running on a machine.

**Usage**  
```
cpstop
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-fwflag -proc</td>
<td>Kills Check Point daemons and Security Servers while maintaining the active Security Policy running in the kernel. Rules with generic allow/reject/drop rules, based on services continue to work.</td>
</tr>
<tr>
<td>-fwflag -default</td>
<td>Kills Check Point daemons and Security Servers. The active Security Policy running in the kernel is replaced with the default filter.</td>
</tr>
</tbody>
</table>
Comments

This command cannot be used to terminate cpriid. cpriid is invoked when the machine is booted and it runs independently.

cpwd_admin

Description

cpwd (also known as WatchDog) is a process that invokes and monitors critical processes such as Check Point daemons on the local machine, and attempts to restart them if they fail. Among the processes monitored by Watchdog are cpd, fwd, fwm. cpwd is part of the SVN Foundation.

fwd does not work in a Management Only machine. To work with fwd in a Management Only machine add -n (for example, fwd -n).

cpwd writes monitoring information to the $CPDIR/log/cpwd.elg log file. In addition, monitoring information is written to the console on UNIX platforms, and to the Windows Event Viewer.

The cpwd_admin utility is used to show the status of processes, and to configure cpwd.

Usage

cpwd_admin start -name <process name> -path <"full path"> -command <"executable name">
**cpwd_admin stop**

**Description**  
Stop a process which is being monitored by cpwd.

**Usage**  
```
cpwd_admin stop -name <process name> [-path "full path"] -command "executable name"]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-name &lt;process name&gt;</td>
<td>A name for the process to be watched by WatchDog.</td>
</tr>
<tr>
<td>-path &quot;full path&quot;</td>
<td>Optional: the full path to the executable (including the executable name) that is used to stop the process.</td>
</tr>
<tr>
<td>-command &quot;executable name &amp; arguments&quot;</td>
<td>Optional: the name of the executable file mentioned in -path</td>
</tr>
</tbody>
</table>

**Comments**  
If -path and -command are not stipulated, cpwd will abruptly terminate the process.

**Example**  
stop the FWM process using fw kill.
```
cpwd_admin stop -name FWM -path "$FWDIR/bin/fw" -command "fw kill fwm"
```

**cpwd_admin list**

**Description**  
This command is used to print a status of the selected processes being monitored by cpwd.

**Usage**  
```
cpwd_admin list
```

**Output**  
The status report output includes the following information:

- **APP** — Application. The name of the process.
- **PID** — Process Identification Number.
- **STAT** — Whether the process Exists (E) or has been Terminated (T).
- **#START** — How many times the process has been started since cpwd took control of the process.
- **START TIME** — The last time the process was run.
- **COMMAND** — The command that cpwd used to start the process.

For example:
**cpwd_admin exist**

**Description**  
This command is used to check whether cpwd is alive.

**Usage**  
cpwd_admin exist

**cpwd_admin kill**

**Description**  
This command is used to kill cpwd.

**Usage**  
cpwd_admin kill

**cpwd_admin config**

**Description**  
This command is used to set cpwd configuration parameters. When parameters are changed, these changes will not take affect until cpwd has been stopped and restarted.

**Usage**  
cpwd_admin config -p

cpwd_admin config -a <value=data value=data...>

cpwd_admin config -d <value value...>

cpwd_admin config -r

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>config -p</td>
<td>Shows the cpwd parameters added using the config -a option.</td>
</tr>
<tr>
<td>config -a</td>
<td>Add one or more monitoring parameters to the cpwd configuration.</td>
</tr>
<tr>
<td>config -d</td>
<td>Delete one or more parameters from the cpwd configuration.</td>
</tr>
<tr>
<td>config -r</td>
<td>Restore the default cpwd parameters.</td>
</tr>
</tbody>
</table>

Where the values are as follows:

```bash
#cpwd_admin list
APP  PID  STAT  #START    START_TIME                COMMAND
```
The following example shows two configuration parameters being changed:

- **timeout** to 120 seconds, and **no_limit** to 10.
config -a and cpwd_admin config -d have no effect if cpwd is running. They will affect cpwd the next time it is run.

**dbedit**

**Description**  
This command is used by administrators to edit the objects file on the SmartCenter Server. From version NG, there is an objects file on the Module and a new file, objects_5_0.C on the SmartCenter Server. A new objects.C file is created on the Module (based on the objects_5_0.C on the SmartCenter Server) whenever a Policy is installed. Editing the objects.C file on the Module is no longer required or desirable, since it will be overwritten the next time a Policy is installed.

**Usage**  

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s server</td>
<td>The SmartCenter Server on which the objects_5_0.C file to be edited is located. If this is not specified in the command line, then the user will be prompted for it. If the server is not localhost, the user will be required to authenticate.</td>
</tr>
<tr>
<td>-u user</td>
<td>The user’s name (the name used for the SmartConsole) or the full path to the certificate file.</td>
</tr>
<tr>
<td>-c certificate</td>
<td>The user’s password (the password used for the SmartConsole).</td>
</tr>
<tr>
<td>-p password</td>
<td>The user’s password (the password used for the SmartConsole).</td>
</tr>
<tr>
<td>Argument</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-f filename</td>
<td>The name of the file containing the commands. If <code>filename</code> is not given, then the user will be prompted for commands.</td>
</tr>
<tr>
<td>-r db-open-reason</td>
<td>A non-mandatory flag used to open the database with a string that states the reason. This reason will be attached to audit logs on database operations.</td>
</tr>
<tr>
<td>-help</td>
<td>Print usage and short explanation.</td>
</tr>
</tbody>
</table>

**dbedit commands:**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>create [object_type] [object_name]</td>
<td>Create an object with its default values. The create command may use an extended (or “owned”) object. Changes are committed to the database only by an update or quit command.</td>
</tr>
<tr>
<td>modify [table_name] [object_name] [field_name] [value]</td>
<td>Modify fields of an object which is: • stored in the database (the command will lock the object in such case). • newly created by <code>dbedit</code> Extended Formats for owned objects can be used: For example, [field_name] = Field_A:Field_B</td>
</tr>
<tr>
<td>update [table_name] [object_name]</td>
<td>Update the database with the object. This command will check the object validity and will issue an error message if appropriate.</td>
</tr>
<tr>
<td>delete [table_name] [object_name]</td>
<td>Delete an object from the database and from the client implicit database.</td>
</tr>
<tr>
<td>addelement [table_name] [object_name] [field_name] [value]</td>
<td>Add an element (of type string) to a multiple field.</td>
</tr>
</tbody>
</table>
Example

Replace the owned object with a new null object, where NULL is a reserved word specifying a null object:

```plaintext
modify network_objects my_obj firewall_setting NULL
```

Example

Extended Format

`firewall_properties` owns the object `floodgate_preferences`.

`floodgate_preferences` has a Boolean attribute `turn_on_logging`, which will be set to `true`.

```plaintext
modify properties firewall_properties
floodgate_preferences:turn_on_logging true
```

`comments` is a field of the owned object contained in the ordered container. The 0 value indicates the first element in the container (zero based index).

```plaintext
modify network_objects my_networkObj interfaces:0:comments my_comment
```

Replace the owned object with a new one with its default values.

```plaintext
modify network_objects my_net_obj interfaces:0:security
interface_security
```
dbver

**Description**  
The `dbver` utility is used to export and import different revisions of the database. The properties of the revisions (last time created, administrator responsible for, etc) can be reviewed. The utility can be found in `$FWDIR/bin`.

**Usage**  
- `export <version_numbers> <delete | keep>`
- `import <exported_version_in_server>`
- `create <version_name> <version_comment>`
- `delete <version_numbers>`
- `print <version_file_path>`
- `print_all`

---

**dbver create**

**Description**  
Create a revision from the current state of `$fwdir/conf`, including current objects, rule bases, etc.

**Usage**  
- `create <version_name> <version_comment>`

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>version_name</code></td>
<td>the name of the revision</td>
</tr>
<tr>
<td><code>version_comment</code></td>
<td>append a comment to the revision</td>
</tr>
</tbody>
</table>

---

**dbver export**

**Description**  
Archive the revision as an archive file in the revisions repository: `$fwdir/conf/db_versions/export`.

**Usage**  
- `export <version_numbers> <delete | keep>`
### dbver import

**Description:** Add an exported revision to the repository a version from `$fwdir/conf/db_versions/export`. Give filename of revision as input.

**Usage:**

```
import <exported_version_in_server>
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>exported_version_in_server</code></td>
<td>The file name of the exported version.</td>
</tr>
</tbody>
</table>

### dbver print

**Description:** Print the properties of the revision.

**Usage:**

```
print <version_file_path>
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>version_file_path</code></td>
<td>The full name and path on the local machine of the revision.</td>
</tr>
</tbody>
</table>

**Output**

```
dbver> print c:\rwright_2002-04-01_160810.tar.gz

Version Id: 1
Version Date: Mon Apr  1 16:08:10 2002
```
dbver print_all

Version Name: save
Created by Administrator: jbrown
Major Version: NG
Minor Version: FP2

dbver print_all

Description  Print the properties of all revisions to be found on the server side:
/fwdir/conf/db_versions

Usage  

dynamic_objects

Description  dynamic_objects specifies an IP address to which the dynamic object will be
resolved on this machine.
This command cannot be executed when the VPN-1Pro Module is running.

Usage  dynamic_objects -o <object_name> [-r [fromIP toIP] ...] [-s] [-a]
[-d] [-l] [-n <object_name>] [-c]

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-o &lt;object_name&gt;</td>
<td>The Object Name.</td>
</tr>
<tr>
<td>-r [fromIP toIP] ...</td>
<td>address ranges — one or more “from IP address to IP address” pairs</td>
</tr>
<tr>
<td>-a [fromIP toIP] ...</td>
<td>add ranges to object</td>
</tr>
<tr>
<td>-d [fromIP toIP] ...</td>
<td>delete range from object</td>
</tr>
<tr>
<td>-l</td>
<td>list dynamic objects</td>
</tr>
<tr>
<td>-n &lt;object_name&gt;</td>
<td>create new object (if VPN-1 Pro Module is not running)</td>
</tr>
<tr>
<td>-c</td>
<td>compare the objects in the dynamic objects file and in objects.c.</td>
</tr>
<tr>
<td>-do object_name</td>
<td>delete object</td>
</tr>
</tbody>
</table>

Example  Create a new dynamic object named “bigserver” and add to it the IP address
range 190.160.1.1-190.160.1.40: dynamic_objects -n bigserver -r
190.160.1.1 190.160.1.40 -a
fw

**Description**  The *fw* commands are used for working with various aspects of FireWall-1. All *fw* commands are executed on the FireWall-1 enforcement module.

Typing *fw* at the command prompt sends a list of available *fw* commands to the standard output.

**Usage**  

```
fw
```

fw ctl

**Description**  The *fw ctl* command controls the FireWall-1 kernel module.

**Usage**  

```
fw ctl <install|uninstall>
fw ctl ip_forwarding [never|always|default]
fw ctl debug [-x] [-m <module>] [+|-] <options | all | 0>
fw ctl debug -buf [buffer size]
fw ctl kdebug
fw ctl pstat [-h][-k][-s][-n][-l]
fw ctl iflist
fw ctl arp [-n]
fw ctl block <on|off>
fw ctl chain
fw ctl conn
```
## fw ctl

### Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Install&gt;</td>
<td>• Uninstall — tells the operating system to stop passing packets to FireWall-1, and unloads the Security Policy. The networks behind it become unprotected.</td>
</tr>
<tr>
<td>Uninstall&gt;</td>
<td>• Install — tells the operating system to start passing packets to FireWall-1. The command <code>fw ctl install</code> runs automatically when <code>cpstart</code> is performed.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: If you run <code>fw ctl uninstall</code> followed by <code>fw ctl install</code>, the Security Policy is not restored.</td>
</tr>
<tr>
<td>debug</td>
<td>Generate debug messages to a buffer.</td>
</tr>
<tr>
<td></td>
<td>`fw ctl debug [-m module] [+</td>
</tr>
<tr>
<td></td>
<td>Sets or resets debug flags for the requested module (default is fw).</td>
</tr>
<tr>
<td></td>
<td>• If + is used, the specified flags are set, and the rest remain as they were.</td>
</tr>
<tr>
<td></td>
<td>• If - is used, the specified flags are reset, and the rest remain as they were.</td>
</tr>
<tr>
<td></td>
<td>• If neither + nor - are used, the specified flags are set and the rest are reset.</td>
</tr>
<tr>
<td></td>
<td><code>fw ctl debug 0</code></td>
</tr>
<tr>
<td></td>
<td>Returns all flags in all modules to their default values, releases the debug buffer (if there was one).</td>
</tr>
<tr>
<td>debug -buf</td>
<td>Allocates a buffer of size kilobytes (default 128) and starts collecting messages there.</td>
</tr>
<tr>
<td>[buffer size]</td>
<td></td>
</tr>
<tr>
<td>debug -h</td>
<td>Print a list of modules and flags.</td>
</tr>
<tr>
<td>debug -x</td>
<td>Do not use.</td>
</tr>
<tr>
<td>kdebug</td>
<td>Reads the debug buffer and obtains the debug messages. If there is no debug buffer, the command will fail. If -t is used, the command will read the buffer every second and print the messages, until <code>Ctrl-C</code> is pressed. Otherwise, it will read the current buffer contents and end. If -T is added the time will be printed in microseconds.</td>
</tr>
</tbody>
</table>
SmartCenter Commands

**fw expdate**

**Description**
This command is used to modify the expiration date of all users and administrators.

**Usage**

```
fw expdate dd-mmm-1976
```
fw fetch

**Comments**  The date can be modified using a filter.

**Example**  `fw expdate 02-mar-2003 -f 01-mar-2003`

---

**fw fetch**

**Description**  This command fetches the Inspection Code from the specified host and installs it to the kernel.

**Usage**  `fw fetch [-n] [-f <filename>] [-c] [-i] master1 [master2] ...`

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-n</code></td>
<td>Fetch the Security Policy from the SmartCenter Server to the local state directory, and install the Policy only if the fetched Policy is different from the Policy already installed.</td>
</tr>
<tr>
<td><code>-f &lt;filename&gt;</code></td>
<td>Fetch the Security Policy from the SmartCenter Server listed in <code>&lt;filename&gt;</code>. If filename is not specified, the list in <code>conf/masters</code> is used.</td>
</tr>
<tr>
<td><code>-c</code></td>
<td>Cluster mode, get policy from one of the cluster members, from the Check Point High Availability (CPHA) kernel list</td>
</tr>
<tr>
<td><code>-i</code></td>
<td>Ignore SIC information (for example, SIC name) in the database and use the information in <code>conf/masters</code>. This option is used when a Security Policy is fetched for the first time by a DAIP Module from a SmartCenter Server with a changed SIC name.</td>
</tr>
<tr>
<td><code>master1</code></td>
<td>Execute command on the designated master. The name of the SmartCenter Server from which to fetch the Policy. You may specify a list of one or more SmartCenter Servers, such as <code>master1 master2</code> which will be searched in the order listed. If no targets is not specified, or if targets is inaccessible, the Policy is fetched from localhost.</td>
</tr>
</tbody>
</table>
fw fetchlogs

Description  fw fetchlogs fetches Log Files from a remote machine. You can use the fw fetchlogs command to transfer Log Files to the machine on which the fw fetchlogs command is executed. The Log Files are read from and written to the directory $FWDIR/log.

Usage  fw fetchlogs [-f file name] ... module

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f filename</td>
<td>The Log Files to be transferred. The file name can include wildcards. In Solaris, any file containing wildcards should be enclosed in quotes. The default parameter is *.log. Related pointer files will automatically be fetched.</td>
</tr>
<tr>
<td>module</td>
<td>The name of the remote machine from where you transfer the Log Files.</td>
</tr>
</tbody>
</table>

Comments  The files transferred by the fw fetchlogs command are MOVED from the source machine to the target machine. This means that they are deleted from the source machine once they have been successfully copied.

Fetching Current Log Data  The active Log File (fw.log) cannot be fetched. If you want to fetch the most recent log data, proceed as follows:

- Run \ to close the currently active Log File and open a new one.
- Run fw lslogs to see the newly-generated log file name.
- Run fw fetchlogs -f filename to transfer the file to the machine on which the fw fetchlogs command is executed. The file is now available for viewing in the SmartView Tracker.

After a file has been fetched, it is renamed. The Module name and the original Log File name are concatenated to create a new file name. The new file name consists of the module name and the original file name separated by two (underscore) _ _ characters.

Example  The following command: fw fetchlogs -f 2001-12-31_123414.log module3

fetches the Log File 2001-12-31_123414.log from Module3.
After the file has been fetched, the Log File is renamed:

```
module3_ _2001-12-31_123414.log
```

Further Info.  See the SmartCenter Guide

---

**fw kill**

**Description**  This command prompts the kernel to shut down all FireWall-1 daemon processes. The command is located in the `$FWDIR/bin` directory on the SmartCenter Server or Module machine.

The FireWall-1 daemons and Security Servers write their pids to files in the `$FWDIR/tmp` directory upon startup. These files are named `$FWDIR/tmp/daemon_name.pid`. For example, the file containing the pid of the FireWall-1 snmp daemon is `$FWDIR/tmp/snmpd.pid`.

**Usage**  `fw kill [-t sig_no] proc-name`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-t sig_no</code></td>
<td>This Unix only command specifies that if the file <code>$FWDIR/tmp/proc-name.pid</code> exists, send signal <code>sig_no</code> to the pid given in the file. If no signal is specified, signal 15 (sigterm or the terminate command) is sent.</td>
</tr>
<tr>
<td><code>proc-name</code></td>
<td>Prompt the kernel to shut down specified FireWall-1 daemon processes.</td>
</tr>
</tbody>
</table>

**Comments**  In Windows, only the default syntax is supported: `fw kill proc_name`. If the `-t` option is used it is ignored.

---

**fw lea_notify**

**Description**  This command should be run from the SmartCenter Server. It sends a LEA_COL_LOGS event to all connected lea clients, see the LEA Specification documentation. It should be used after new log files have been imported (manually or automatically) to the `$FWDIR/log` directory in order to avoid the scheduled update which takes 30 minutes.

**Usage**  `fw lea_notify`
fw lichosts

**Description**  This command prints a list of hosts protected by VPN-1Pro products. The list of hosts is in the file $fwdir/database/fwd.h

**Usage**  
fw lichosts [-x] [-l]

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-x</td>
<td>Use hexadecimal format.</td>
</tr>
<tr>
<td>-l</td>
<td>Use long format.</td>
</tr>
</tbody>
</table>

fw log

**Description**  fw log displays the content of Log files.

**Usage**  
fw log [-f [-t]] [-n] [-l] [-o] [-c action] [-h host] [-s starttime] [-e endtime] [-b starttime endtime] [-u unification_scheme_file] [-m unification_mode(initial|semi|raw)] [-a] [-k (alert_name|all)] [-g] [logfile]

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f [-t]</td>
<td>After reaching the end of the currently displayed file, do not exit (the default behavior), but continue to monitor the Log file indefinitely and display it while it is being written. The -t parameter indicates that the display is to begin at the end of the file, in other words, the display will initially be empty and only new records added later will be displayed. -t must come with a -f flag. These flags are relevant only for active files.</td>
</tr>
<tr>
<td>-n</td>
<td>Do not perform DNS resolution of the IP addresses in the Log file (the default behavior). This option significantly speeds up the processing.</td>
</tr>
<tr>
<td>-l</td>
<td>Display both the date and the time for each log record (the default is to show the date only once above the relevant records, and then specify the time per log record).</td>
</tr>
</tbody>
</table>
**Argument** | **Description**  
---|---  
-o | Show detailed log chains (all the log segments a log record consists of)  
-c action | Display only events whose action is action, that is, accept, drop, reject, authorize, deauthorize, encrypt and decrypt. Control actions are always displayed.  
-h host | Display only log whose origin is the specified IP address or name.  
-s starttime | Display only events that were logged after the specified time (see format below). `starttime` may be a date, a time, or both. If date is omitted, then today's date is assumed.  
-e endtime | Display only events that were logged before the specified time (see format below). `endtime` may be a date, a time, or both.  
-b starttime endtime | Display only events that were logged between the specified start and end times (see format below), each of which may be a date, a time, or both. If date is omitted, then today's date is assumed. The start and end times are expected after the flag.  
-u unification_scheme_file | Unification scheme file name.
Where the full date and time format is: **MMM DD, YYYY HH:MM:SS**. For example: May 26, 1999 14:20:00

It is possible to specify date only in the format **MMM DD, YYYY**, or time only, in the format: **HH:MM:SS**, where time only is specified, the current date is assumed.

**Example**

```
fw log
fw log | more
fw log -c reject
fw log -s "May 26, 1999"
fw log -f -s 16:00:00
```
fw logswitch

Output

\[ \text{<date>} \ <time> \ <action> \ <origin> \ <interface \ dir \ and \ name> \ [\text{alert}] \ [\text{field \ name: \ field \ value;}] \ldots \]

Each output line consists of a single log record, whose fields appear in the format shown above.

Example

14:56:39 reject jam.checkpoint.com >daemon alert src: veredr.checkpoint.com; dst: jam.checkpoint.com; user: a; rule: 0; reason: Client Encryption: Access denied - wrong user name or password; scheme: IKE; reject_category: Authentication error; product: VPN-1 Pro

14:57:49 authcrypt jam.checkpoint.com >daemon src: veredr.checkpoint.com; user: a; rule: 0; reason: Client Encryption: Authenticated by Internal Password; scheme: IKE; methods: AES-256, IKE, SHA1; product: VPN-1 Pro;

14:57:49 keyinst jam.checkpoint.com >daemon src: veredr.checkpoint.com; peer gateway: veredr.checkpoint.com; scheme: IKE; IKE: Main Mode completion.; CookieI: 32f09ca38aeaf4a3; CookieR: 73b91d59b378958c; msgid: 47ad4a8d; methods: AES-256 + SHA1, Internal Password; user: a; product: VPN-1 Pro;

fw logswitch

Description

fw logswitch creates a new active Log File. The current active Log File is closed and renamed by default \$FWDIR/log/current_time_stamp.log unless you define an alternative name that is unique. The format of the default name current_time_stamp.log is YYYY-MM-DD_HHMMSS.log. For example: 2003-03-26_041200.log

Warning:

- The Logswitch operation fails if a log file is given an pre-existing file name.
- The rename operation fails on Windows if the active log that is being renamed, is open at the same time that the rename operation is taking place; however; the Logswitch will succeed and the file will be given the default name \$FWDIR/log/current_time_stamp.log.

The new Log File that is created is given the default name \$FWDIR/log/fw.log. Old Log Files are located in the same directory.
A SmartCenter Server can use `fw logswitch` to switch a Log File on a remote machine and transfer the Log File to the SmartCenter Server. This same operation can be performed for a remote machine of version NG FP2 and higher, using “fw lslogs” on page 604 and “fw fetchlogs” on page 606.

When a log file is sent to the SmartCenter Server, the data is compressed.

**Usage**

```
fw logswitch [-audit] [filename]
fw logswitch -h hostname [+-][filename]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-audit</code></td>
<td>Does logswitch for the SmartCenter audit file. This is relevant for local activation.</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>The name of the file to which the log is saved. If no name is specified, a default name is provided.</td>
</tr>
<tr>
<td><code>-h hostname</code></td>
<td>The resolvable name or IP address of the remote machine (running either a VPN-1 Pro Module or a SmartCenter Server) on which the Log File is located. The SmartCenter Server (on which the <code>fw logswitch</code> command is executed) must be defined as one of <code>hostname</code>’s SmartCenter Servers. In addition, you must initialize SIC between the SmartCenter Server and the <code>hostname</code>.</td>
</tr>
</tbody>
</table>

**Comments**

Files are created in the `$FWDIR/log` directory on both `hostname` and the SmartCenter Server when the `+` or `-` parameters are specified. Note that if `-` is specified, the Log File on the host is deleted rather than renamed.

`hostname` specified:
fw mergefiles

- **filename** specified - On hostname, the old Log File is renamed to `old_log`. On the SmartCenter Server, the copied file will have the same name, prefixed by hostname’s name. For example, the command `fw logswitch -h venus +xyz` creates a file named `venus_xyz` on the SmartCenter Server.

- **filename** not specified - On hostname, the new name is the current date, for example: `2003-03-26_041200.log`. On the SmartCenter Server, the copied file will have the same name, but prefixed by `hostname_`. For example, `target_2003-03-26_041200.log`.

hostname not specified

- **filename** specified - On the SmartCenter Server, the old Log File is renamed to `old_log`.

- **filename** not specified - On the SmartCenter Server, the old Log File is renamed to the current date.

If either the SmartCenter Server or hostname is an NT machine, the files will be created using the NT naming convention.

**Compression**

When log files are transmitted from one machine to another, they are compressed using the zlib package, a standard package used in the Unix gzip command (see RFC 1950 to RFC 1952 for details). The algorithm is a variation of LZ77 method.

The compression ratio varies with the content of the log records and is difficult to predict. Binary data are not compressed, but string data such as user names and URLs are compressed.

---

**fw mergefiles**

**Description**

This command merges several Log Files into a single Log File. The merged file can be sorted according to the creation time of the Log entries, and the times can be “fixed” according to the time zones of the origin Log Servers.

Logs entries with the same Unique-ID are unified. If a Log switch was performed before all the segments of a specific log were received, this command will merge the records with the same Unique-ID from two different files, into one fully detailed record.
Usage

```
fw mergefiles [-s] [-t time_conversion_file]
log_file_name_1 [... log_file_name_n].output_file
```

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s</td>
<td>Sort merged file by log records time field.</td>
</tr>
</tbody>
</table>
| -t time_conversion_file | Fix different GMT zone log records time in the event that the log files originated from Log Servers in different time zone. The `time_conversion_file` format is as follows:
```
ip-address signed_date_time_in_seconds
```
| log_file_name_n | Full pathnames of the Log File(s). |
| output_file     | Full pathname of the output Log File. |

Comments

It is not recommended to merge the current active `fw.log` file with other Log Files. Instead, run the `fw logswitch` command and then run `fw mergefiles`.

---

**fw lslogs**

**Description**

This command displays a list of Log Files residing on a remote or local machine. You must initialize SIC between the SmartCenter Server and the remote machine.

**Usage**

```
fw lslogs [[-f file name] ...] [-e] [-s name | size | stime | etime] [-r] [module]
```
Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f filename</td>
<td>The list of files to be displayed. The file name can include wildcards. In Unix, any file containing wildcards should be enclosed in quotes. The default parameter is *.log.</td>
</tr>
<tr>
<td>-e</td>
<td>Display an extended file list. It includes the following data:</td>
</tr>
<tr>
<td></td>
<td>• Size – The size of the file and its related pointer files together.</td>
</tr>
<tr>
<td></td>
<td>• Creation Time – The time the Log File was created.</td>
</tr>
<tr>
<td></td>
<td>• Closing Time – The time the Log File was closed.</td>
</tr>
<tr>
<td></td>
<td>• Log File Name – The file name.</td>
</tr>
<tr>
<td>-s</td>
<td>Specify the sort order of the Log Files using one of the following sort options:</td>
</tr>
<tr>
<td></td>
<td>• name – The file name.</td>
</tr>
<tr>
<td></td>
<td>• size – The file size.</td>
</tr>
<tr>
<td></td>
<td>• stime – The time the Log File was created.</td>
</tr>
<tr>
<td></td>
<td>• etime – The time the Log File was closed.</td>
</tr>
<tr>
<td></td>
<td>The default is stime.</td>
</tr>
<tr>
<td>-r</td>
<td>Reverse the sort order (descending order).</td>
</tr>
<tr>
<td>module</td>
<td>The name of the machine on which the files are located. It can be a module or a Log Server. The default is localhost.</td>
</tr>
</tbody>
</table>

Example

This example shows the extended file list you see when you use the `fw lslogs -e` command:

<table>
<thead>
<tr>
<th>fw lslogs -e module3</th>
<th>Size</th>
<th>Creation Time</th>
<th>Closing Time</th>
<th>Log file name</th>
</tr>
</thead>
<tbody>
<tr>
<td>16KB</td>
<td>10Jan2002 18:36:05</td>
<td>--</td>
<td>fw.log</td>
<td></td>
</tr>
</tbody>
</table>
**fw putkey**

**Description**  
This command installs a VPN-1 & FireWall-1 authentication password on a host. This password is used to authenticate internal communications between VPN-1 Pro Modules and between a Check Point Module and its SmartCenter Server. A password is used to authenticate the control channel the first time communication is established. This command is required for backward compatibility scenarios.

**Usage**  

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-opsec</td>
<td>Only VPN-1 &amp; FireWall-1 control connections are enabled.</td>
</tr>
<tr>
<td>-no_opsec</td>
<td>Only OPSEC control connections are enabled.</td>
</tr>
<tr>
<td>-ssl</td>
<td>The key is used for an SSL connection.</td>
</tr>
<tr>
<td>-no_ssl</td>
<td>The key is not used for an SSL connection.</td>
</tr>
<tr>
<td>-k num</td>
<td>The length of the first S/Key password chain for fwa1 authentication (Check Point's proprietary authentication protocol). The default is 7. When fewer than 5 passwords remain, the hosts renegotiate a chain of length 100, based on a long random secret key. The relatively small default value ensures that the first chain, based on a short password entered by the user, is quickly exhausted.</td>
</tr>
</tbody>
</table>
fw repairlog

Description
fw repairlog rebuilds a Log file's pointer files. The three files name.logptr, name.loginitial_ptr and name.logaccount_ptr are recreated from data in the specified Log file. The Log file itself is modified only if the -u flag is specified.

Usage
fw repairlog [-u] logfile

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-u</td>
<td>Indicates that the unification chains in the Log file should be rebuilt.</td>
</tr>
<tr>
<td>logfile</td>
<td>The name of the Log file to repair.</td>
</tr>
</tbody>
</table>

Comments
This command is never used in a script.
**fw sam**

**Description**  
This command is used to manage the Suspicious Activity Monitoring (SAM) server. Use the SAM server to block connections to and from IP addresses without the need to change the Security Policy.

SAM commands are logged. Use this command to (also) monitor active SAM requests (see -M option).

**To configure the SAM Server** on the SmartCenter Server or FireWall-1 Gateway machine, use SmartDashboard to edit the Advanced > SAM page of the Check Point Gateway object.

**Usage**  
Add/Cancel SAM rule according to criteria:

```
```

Delete all SAM rules:

```
fw sam [-v] [-s <sam server>] [-S <server sic name>] [-f <fw host>] -D
```

Monitor all SAM rules:

```
```

Monitor SAM rules according to criteria:

```
```

**Syntax**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>-v</td>
<td>Verbose mode. Writes one message (describing whether the command was successful or not) to stderr for each VPN-1 Pro Gateway machine on which the command is enforced.</td>
</tr>
<tr>
<td>-s sam_server</td>
<td>The IP address (in dot format) or the resolvable name of the FireWalled host that will enforce the command. The default is localhost.</td>
</tr>
</tbody>
</table>
fw sam

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>-S server_sic_name</td>
<td>The SIC name for the SAM server to be contacted. It is expected that the SAM server will have this SIC name, otherwise the connection will fail. If no server SIC name is supplied the connection will proceed without SIC names comparison. For more information about enabling SIC refer to the OPSEC API Specification.</td>
</tr>
</tbody>
</table>
| -f <fw host>        | Specify the host, the VPN-1 Pro Gateway machine on which to enforce the action. host can be one of the following (default is All):  
  - localhost—Specify the computer running the SAM server to enforce the action on it.  
  - The name of the object or group—the action is enforced on this object; if this object is a group, on every object in the group.  
  - Gateways—Action enforced on FireWalls defined as gateways and managed by SmartCenter Server where the SAM server runs.  
  - All—Enforced on FireWalls managed by SmartCenter Server where SAM server runs. |
| -D                  | Cancel all inhibit (-i, -j, -I, -J) and notify (-n) commands. To “ uninhibit” inhibited connections, execute fw sam with the -C or -D parameters. It is also possible to use this command for active SAM requests. |
| -C                  | Cancel the command to inhibit connections with the specified parameters. These connections will no longer be inhibited (rejected or dropped). The command parameters must match the ones in the original command, except for the -t (timeout) parameter. |
| -t timeout          | The time period (in seconds) for which the action will be enforced. The default is forever or until cancelled.                                                                                                                                                  |
| -l log              | The type of the log for enforced actions can be one of the following: nolog, long_noalert, long_alert. The default is long_alert.                                                                                                                                  |
Criteria are used to match connections, and are composed of various combinations of the following parameters:

<src ip><source netmask><destination ip><destination netmask><service><protocol>

Possible combinations are:
src <ip>
dst <ip>
any <<ip>
subsrc <ip><netmask>
subdst <ip><netmask>
subany <ip><netmask>
srv <src ip><dest ip><service><protocol>

Usage

-M
Monitor the active SAM requests with the specified actions and criteria.

all
Get all active requests. For monitoring purposes only.
Syntax

<table>
<thead>
<tr>
<th>Criteria Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>src &lt;ip&gt;</td>
<td>Match the source IP address of the connection.</td>
</tr>
<tr>
<td>dst &lt;ip&gt;</td>
<td>Match the destination IP address of the connection.</td>
</tr>
<tr>
<td>any &lt;ip&gt;</td>
<td>Match either the source IP address or destination IP address of the connection.</td>
</tr>
<tr>
<td>subsrc &lt;ip&gt; &lt;netmask&gt;</td>
<td>Match the source IP address of the connections according to the netmask.</td>
</tr>
<tr>
<td>subdst &lt;ip&gt; &lt;netmask&gt;</td>
<td>Match the destination IP address of the connections according to the netmask.</td>
</tr>
<tr>
<td>subany &lt;ip&gt; &lt;netmask&gt;</td>
<td>Match either the source IP address or destination IP address of connections according to the netmask.</td>
</tr>
<tr>
<td>srv &lt;src ip&gt; &lt;dst ip&gt; &lt;service&gt; &lt;protocol&gt;</td>
<td>Match the specific source IP address, destination IP address, service and protocol.</td>
</tr>
<tr>
<td>subsrv &lt;src ip&gt; &lt;netmask&gt; &lt;dst ip&gt; &lt;netmask&gt; &lt;service&gt; &lt;protocol&gt;</td>
<td>Match the specific source IP address, destination IP address, service and protocol. Source and destination IP addresses are assigned according to the netmask.</td>
</tr>
</tbody>
</table>
Example

This command inhibits all connections originating on louvre for 10 minutes. Connections made during this time will be rejected:

```
fw sam -t 600 -i src louvre
```

This command inhibits all FTP connections from the louvre subnet to the eifel subnet. All existing open connections will be closed. New connection will be dropped, a log is kept and an alert is sent:

```
fw sam -l long_alert -J subsrvs louvre 255.255.255.0 eifel 21 6
```

The previous command will be enforced forever - or until canceled by the following command:

```
fw sam
```
This command monitors all active “inhibit” or “notify SAM” requests for which lourve is the source or destination address:

```
fw sam -M -niJ any lourve
```

This command cancels the command in the first example:

```
fw sam -C -i src lourve
```

---

**fwm**

**Description**

This command is used to perform management operations on VPN-Pro. It controls *fwd* and all Check Point daemons.

**Usage**

```
fwm
```

---

**fwm dbimport**

**Description**

*fwm dbimport* imports users into the Check Point User Database from an external file. You can create this file yourself, or use a file generated by *fwm dbexport*.

**Usage**

```
fwm dbimport [-m] [-s] [-v] [-r] [-k errors] [-f file] [-d delim]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-m</td>
<td>If an existing user is encountered in the import file, the user’s default values will be replaced by the values in the template (the default template or the one given in the attribute list for that user in the import file), and the original values will be ignored.</td>
</tr>
<tr>
<td>-s</td>
<td>Suppress the warning messages issued when an existing user’s values are changed by values in the import file.</td>
</tr>
<tr>
<td>-v</td>
<td>verbose mode</td>
</tr>
<tr>
<td>-r</td>
<td><em>fwm dbimport</em> will delete all existing users in the database.</td>
</tr>
</tbody>
</table>

---

74  CLI Guide
Comments

The IKE pre shared secret does not work when exporting from one machine and importing to another.

To ensure that there is no dependency on the previous database values, use the \(-r\) flag together with the \(-m\) flag.

File Format

The import file must conform to the following Usage:

- The first line in the file is an attribute list.
  - The attribute list can be any partial set of the following attribute set, as long as \(\text{name}\) is included:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-k) errors</td>
<td>Continue processing until (\text{nerror}) errors are encountered. The line count in the error messages starts from 1 including the attributes line and counting empty or commented out lines.</td>
</tr>
<tr>
<td>(-f) file</td>
<td>The name of the import file. The default import file is ($FWDIR/conf/user_def_file). Also see the requirements listed under “File Format” on page 75.</td>
</tr>
<tr>
<td>(-d) delim</td>
<td>Specifies a delimiter different from the default value ((;)).</td>
</tr>
</tbody>
</table>

- The attributes must be separated by a delimiter character. The default delimiter is the \(\;\) character. However, you can use a different character by specifying the \(-d\) option in the command line.
- The rest of the file contains lines specifying the values of the attributes per user. The values are separated by the same delimiter character used for the attribute list. An empty value for an attribute means use the default value.
- For attributes that contain a list of values (for example, days), enclose the values in curly braces, that is, \{"\}. Values in a list must be separated by commas. If there is only one value in a list, the braces may be
omitted. A + or - character appended to a value list means to add or 
delete the values in the list from the current default user values.
Otherwise the default action is to replace the existing values.

• Legal values for the days attribute are: MON, TUE, WED, THU, FRI, SAT, SUN.
• Legal values for the authentication method are: Undefined, S/Key, 
  SecurID, Unix Password, VPN-1 & FireWall-1 Password, RADIUS, 
  Defender.
• Time format is hh:mm.
• Date format is dd-mmm-yy, where mmm is one of {Jan, Feb, Mar, 
  Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec}.
• If the S/Key authentication method is used, all the other attributes 
  regarding this method must be provided.
• If the VPN-1 & FireWall-1 password authentication method is used, a 
  valid VPN-1 & FireWall-1 password should be given as well. The 
  password should be encrypted with the C language encrypt function.
• Values regarding authentication methods other than the one specified 
  are ignored.
• The userc field specifies the parameters of the user’s SecuRemote 
  connections, and has three parameters, as follows:
  key encryption method - DES, CLEAR, Any
  data encryption method - DES, CLEAR, Any
  integrity method - MD5,[blank] = no data integrity
  “Any” means the best method available for the connection. This 
  depends on the encryption methods available to both sides of the 
  connection. For example,
  (DES,CLEAR,) means: key encryption method is DES; no data 
  encryption; no data integrity
• A line beginning with the ! character is considered a comment.

---

**fwm dbexport**

**Description**  
*fwm dbexport* exports the Check Point User Database to a file. The file 
may be in one of the following formats:

• the same Usage as the import file for *fwm dbimport*

• LDIF format, which can be imported into an LDAP Server using 
  *ldapmodify*

**Usage**  
To export the User Database to a file that can be used with *fwm dbimport*:
fwm dbexport [ [-g group | -u user] [-d delim] 
[-a {attrib1, attrib2, ...}] ] [-f file] ]

To export the User Database as an LDIF file:
fwm dbexport -l -p [-d] -s subtree [-f file] [-k IKE-shared-secret]

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-g group</td>
<td>Specifies a group (group) to be exported. The users in the group are not exported.</td>
</tr>
<tr>
<td>-u user</td>
<td>Specifies that only one user (user) is to be exported.</td>
</tr>
<tr>
<td>-d</td>
<td>Debug flag</td>
</tr>
<tr>
<td>-a {attrib1, attrib2, ...}</td>
<td>Specifies the attributes to export, in the form of a comma-separated list, between {} characters, for example, -a {name,days}. If there is only one attribute, the {} may be omitted.</td>
</tr>
<tr>
<td>-f file</td>
<td>file specifies the name of the output file. The default output file is $FWDIR/conf/user_def_file.</td>
</tr>
<tr>
<td>-l</td>
<td>Create an LDIF format file for importation by an LDAP server.</td>
</tr>
<tr>
<td>-p</td>
<td>The profile name.</td>
</tr>
<tr>
<td>-s</td>
<td>The branch under which the users are to be added.</td>
</tr>
<tr>
<td>-k</td>
<td>This is the Account Unit’s IKE shared secret (IKE Key in the Encryption tab of the Account Unit Properties window)</td>
</tr>
</tbody>
</table>

Comments

Note:

- The IKE pre shared secret does not work when exporting from one machine and importing to another.
- If you use the -a parameter to specify a list of attributes, and then import the created file using fwm dbimport, the attributes not exported will be deleted from the user database.
- fwm dbexport and fwm dbimport (non-LDIF Usage) cannot export and import user groups. To export and import a user database, including groups, proceed as follows:
* Run `fwm dbexport` on the source SmartCenter Server.
* On the destination SmartCenter Server, create the groups manually.
* Run `fwm dbimport` on the destination SmartCenter Server.

The users will be added to the groups to which they belonged on the source SmartCenter Server.

- If you wish to import different groups of users into different branches, run `fwm dbexport` once for each subtree, for example:

  ```
  fwm dbexport -f f1 -l -s ou=marketing,o=WidgetCorp,c=us
  fwm dbexport -f f2 -l -s ou=rnd,o=WidgetCorp,c=uk
  ```

Next, import the individual files into the LDAP server one after the other. For information on how to do this, refer to the documentation for your LDAP server.

- The LDIF file is a text file which you may wish to edit before importing it into an LDAP server. For example, in the Check Point user database, user names may be what are in effect login names (such as “maryj”) while in the LDAP server, the DN should be the user’s full name (“Mary Jones”) and “maryj” should be the login name.

**Example**

Suppose the User Database contains two users, “maryj” and “ben”.

```
fwm dbexport -l -s o=WidgetCorp,c=us
```

creates a LDIF file consisting of two entries with the following DNs:

```plaintext
  cn=ben,o=WidgetCorp,c=us
  cn=maryj,o=WidgetCorp,c=us
```

---

**fwm dbload**

**Description**

This command downloads the user database and network objects information to selected targets. If no target is specified, then the database is downloaded to localhost.

**Usage**

```bash
fwm dbload [-all | -conf conffile] [targets]
```
**fw hastat**

**Description**  The `fw hastat` command displays information about High Availability machines and their states.

**Usage**  `fw hastat [<target>]`

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-all</td>
<td>Execute command on all targets specified in the default system configuration file (<code>$FWDIR/conf/sys.conf</code>). This file must be manually created.</td>
</tr>
<tr>
<td>-conf conffile</td>
<td>Only OPSEC control connections are enabled.</td>
</tr>
<tr>
<td>targets</td>
<td>Execute command on the designated targets.</td>
</tr>
</tbody>
</table>

**fw ikecrypt**

**Description**  The `fw ikecrypt` command line encrypts the password of a SecuRemote user using IKE. The resulting string must then be stored in the LDAP database.

**Usage**  `fw ikecrypt shared-secret user-password`

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>shared-secret</td>
<td>The IKE Key defined in the Encryption tab of the LDAP Account Unit Properties window.</td>
</tr>
<tr>
<td>user-password</td>
<td>The SecuRemote user’s password.</td>
</tr>
</tbody>
</table>
Comments
An internal CA must be created before implementing IKE encryption. An Internal CA is created during the initial configuration of the SmartCenter Server, following installation.

---

**fwm load**

**Description**
This command compiles and installs a Security Policy or a specific version of the Security Policy on the target's VPN-1 Pro Modules. This is done in one of two ways:

- `fwm load` compiles and installs an Inspection Script (*.pf) file on the designated VPN-1 Pro Modules.
- `fwm load` converts a Rule Base (*.W) file created by the GUI into an Inspection Script (*.pf) file then installs it to the designated VPN-1 Pro Modules.

Versions of the Security Policy and databases are maintained in a version repository on the SmartCenter Server. Using this command specific versions of the Security Policy can be installed on a Module (local or remote) without changing the definition of the current active database version on the SmartCenter Server.

To protect a target, you must load a Policy that contains rules whose scope matches the target. If none of the rules are enforced on the target, then all traffic through the target is blocked.

**Usage**
```
fwm load [-all | -conf conffile] [<filter-file> | <rulebase>]
 [-ip IPaddress] <targets>
fwm load [-v version number] <rulebase> <targets>
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-all</td>
<td>Execute command on all targets specified in the default system configuration file ($FWDIR/conf/sys.conf). This file must be manually created.</td>
</tr>
<tr>
<td>conf conffile</td>
<td>Execute command on targets specified in the conffile.</td>
</tr>
<tr>
<td>filter-file</td>
<td>An inspection Script (*.pf)</td>
</tr>
<tr>
<td>rulebase</td>
<td>A Rule Base file (*.W) created by the GUI. The file's full pathname must be given.</td>
</tr>
</tbody>
</table>
Comments

If you are installing a specific version of a Security Policy on a remote Module, the local user database is not installed.

Backward Compatibility. The version repository can maintain Security Policy versions from NG FP2 and higher. Currently, only VPN-1/FireWall-1 Security Policies that were defined and saved from version NG FP3 and higher, can be installed on Modules.

Example

The following command installs the Security Policy standard.W from the database version -v18 in the target module johnny.

fwm load -v18 standard.W johnny

**fwm lock_admin**

**Description**

This commands enables you to view and unlock locked administrators.

**Usage**

fwm lock_admin [-v] [-u administrator] [-ua]

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-v</td>
<td>View the names of all locked administrators</td>
</tr>
<tr>
<td>-u administrator</td>
<td>Unlock a single administrator</td>
</tr>
<tr>
<td>-ua</td>
<td>Unlock all locked administrators</td>
</tr>
</tbody>
</table>

**fwm logexport**

**Description**

fwm logexport exports the Log file to an ASCII file.
Usage


Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d delimiter</td>
<td>Set the output delimiter. The default is a semicolon (;)</td>
</tr>
<tr>
<td>-i filename</td>
<td>The name of the input Log file. The default is the active Log file, fw.log</td>
</tr>
<tr>
<td>-o outputfile</td>
<td>The name of the output file. The default is printing to the screen.</td>
</tr>
<tr>
<td>-n</td>
<td>Do not perform DNS resolution of the IP addresses in the Log file (this option significantly speeds the processing).</td>
</tr>
<tr>
<td>-p</td>
<td>Do not perform service resolution. A service port number is displayed.</td>
</tr>
<tr>
<td>-f</td>
<td>If this is the active Log file (fw.log), wait for new records and export them to the ASCII output file as they occur.</td>
</tr>
<tr>
<td>-m</td>
<td>This flag specifies the unification mode. • initial - the default mode. Complete the unification of log records; that is, output one unified record for each id. • semi - step-by-step unification, that is, for each log record, output a record that unifies this record with all previously-encountered records with the same id. • raw - output all records, with no unification.</td>
</tr>
<tr>
<td>-a</td>
<td>Show account records only (the default is to show all records)</td>
</tr>
</tbody>
</table>

Comments

Controlling the Output of fwm logexport using logexport.ini

The output of fwm logexport can be controlled by creating a file called logexport.ini and placing it in the conf directory: $FWDIR/conf. The logexport.ini file should be in the following format:

[Fields_Info]

included_fields = field1,field2,field3,<REST_OF_FIELDS>,field100
excluded_fields = field10,field11

note that:
- the num field will always appear first, and cannot be manipulated using logexport.ini
- <REST_OF_FIELDS> is a reserved token that refers to a list of fields. It is optional. If -f option is set, <REST_OF_FIELDS> is based on a list of fields taken from the file logexport_default.C.
- If -f is not set, <REST_OF_FIELDS> will be based on the given input log file.
- It is not mandatory to specify both included_fields and excluded_fields.

Format:
The fwm logexport output appears in tabular format. The first row lists the names of all fields included in the subsequent records. Each of the subsequent rows consists of a single log record, whose fields are sorted in the same order as the first row. If a record has no information on a specific field, this field remains empty (as indicated by two successive semi-colons).

Example
num;date;time;orig;type;action;alert;i/f_name;i/f_dir;product;sys_message;
;service;s_port;src;dst;
0; 5Dec2002;9:08:44;jam.checkpoint.com;control;
;;daemon;inbound;VPN-1 & FireWall-1;The hme0 interface is not protected by the anti-spoofing feature. Your network may be at risk;;;;;
1; 5Dec2002;9:08:44;jam.checkpoint.com;control;
;;daemon;inbound;VPN-1 & FireWall-1;;ftp;23456;1.2.3.4;3.4.5.6;

---

fwm sic_reset

Description
This command is used to reset the Internal CA and delete all the certificates from the Internal CA and the Internal CA itself. After running sic_reset, the ICA should be initialized through the cpconfig command. If this command is run all the certified IKE from the Internal CA should be removed (using the SmartConsole).

Usage
fwm sic_reset
fwm unload <targets>

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sic_reset</td>
<td>Resets the internal CA SIC certificates and deletes the Internal CA.</td>
</tr>
</tbody>
</table>

**fwm unload <targets>**

**Description**  
This command uninstalls the currently loaded Inspection Code from selected targets.

**Usage**  
fwm unload <targets>[-all | -conf conffile]

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targets</td>
<td>Execute command on the designated targets.</td>
</tr>
<tr>
<td>-all</td>
<td>Execute command on all targets specified in the default system configuration file ($FWDIR/conf/sys.conf). This file must be manually created.</td>
</tr>
<tr>
<td>conf conffile</td>
<td>Execute command on targets specified in the conffile.</td>
</tr>
</tbody>
</table>

**fwm ver**

**Description**  
fwm ver displays the build number.

**Usage**  
fwm ver [-f <filename>]
SmartCenter Commands

GeneratorApp

Description
This command generates a report for the SmartView Reporter. Both command line parameters are required.

Syntax

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f &lt;filename&gt;</td>
<td>Print the version name and build number to the specified file.</td>
</tr>
<tr>
<td>-p port</td>
<td>The ELA Proxy’s port number. Default is 18187.</td>
</tr>
</tbody>
</table>
| -f token value  | A field to be added to the log, represented by a token-value pair as follows:  
|                 | • token is the name of the field to be added to the log. token may not contain spaces.  
|                 | • value is the field’s value. value may not contain spaces.  
|                 | This option may be used multiple times to add multiple token-value pairs to the log.  
|                 | If token is a reserved log field name, the specified field’s value will appear in the corresponding column in the Log Viewer. Otherwise, the token-value pair will be displayed in the Info. column in the Log Viewer. |
| -m alertty      | The alert to be triggered at the ISP site. This alert overrides the alert specified in the log message generated by the alert daemon.  
|                 | The response to the alert is handled according to the actions specified in the ISP’s Security Policy:  
|                 | The following alerts execute the OS commands defined in the corresponding fields of the Log and Alert tab of the Properties Setup window in Global Properties:  
|                 | • alert. Popup alert command.  
|                 | • mail. Mail alert command.  
|                 | • snmptrap. SNMP trap alert command.  
|                 | • spoofalert. Anti-spoof alert command.  
|                 | The following NetQuota and ServerQuota alerts execute the OS commands specified in:  
|                 | $FWDIR/conf/objects.C:  
|                 | value=clientquotaalert.  
|                 | Parameter=clientquotaalertcmd |
Usage

GeneratorApp [Directory/]"" ] {ReportID}

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directory</td>
<td>The result directory (that is, the location at which the result is placed).</td>
</tr>
<tr>
<td>ReportID</td>
<td>The report ID required for command line generations.</td>
</tr>
<tr>
<td></td>
<td>The Report ID must be enclosed within curly braces.</td>
</tr>
<tr>
<td></td>
<td>For a list of all Report IDs see Appendix B “Predefined Reports” in the SmartView Reporter User Guide.</td>
</tr>
</tbody>
</table>

Example

For automatic directory computation use “"". In such a case, the directory should be as follows:

<Result location>/ <Report Name>/ <Generation Date and Time>

inet_alert

Description

This command notifies a company’s Internet Service Provider (ISP) when the company’s corporate network is under attack. The \texttt{inet\_alert} utility forwards log messages generated by the alert daemon to an external Management Station, typically located at the ISP site. The ISP can then analyze the alert and decide how to react.

\texttt{inet\_alert} uses the ELA Protocol to send the alert. The Management Station receiving the alert must be running the ELA Proxy.

If communication with the ELA Proxy is to be authenticated or encrypted, a key exchange must be performed between the Management Station running the ELA Proxy and the VPN-1 Pro Module generating the alert.

To use this utility, enter it into a script. From \texttt{Global Properties > Logs and alert > alert commands > early versions compatibility > run 4.x alert script}, and enter the name of the script.

Usage

\texttt{inet\_alert -s ipaddr [-o] [-a auth\_type] [-p port] [-f token value] [-m alert\_type]}
### Syntax

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-s ipaddr</code></td>
<td>The IP address (in dot format) of the ELA Proxy to be contacted.</td>
</tr>
<tr>
<td><code>-o</code></td>
<td>Print the alert log received by <code>inet_alert</code> to <code>stdout</code>. Use this option when <code>inet_alert</code> is part of a pipe.</td>
</tr>
<tr>
<td><code>-a auth_type</code></td>
<td>The type of connection to the ELA Proxy. One of the following values:   &lt;br&gt;  • <code>ssl_opsec</code>. Means the connection is authenticated and encrypted, (Default) &lt;br&gt;  • <code>auth_opsec</code>. Means the connection is authenticated. &lt;br&gt;  • <code>clear</code>. Means the connection is neither authenticated nor encrypted.</td>
</tr>
</tbody>
</table>
### Return Value

<table>
<thead>
<tr>
<th>exit status</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Execution was successful.</td>
</tr>
<tr>
<td>102</td>
<td>Undetermined error.</td>
</tr>
<tr>
<td>103</td>
<td>Unable to allocate memory.</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p port</td>
<td>The ELA Proxy’s port number. Default is 18187.</td>
</tr>
<tr>
<td>-f token value</td>
<td>A field to be added to the log, represented by a token-value pair as follows:</td>
</tr>
<tr>
<td></td>
<td>• <code>token</code> is the name of the field to be added to the log. <code>token</code> may not contain spaces.</td>
</tr>
<tr>
<td></td>
<td>• <code>value</code> is the field’s value. <code>value</code> may not contain spaces.</td>
</tr>
<tr>
<td></td>
<td>This option may be used multiple times to add multiple token-value pairs to the log.</td>
</tr>
<tr>
<td></td>
<td>If <code>token</code> is a reserved log field name, the specified field’s value will appear in the corresponding column in the Log Viewer. Otherwise, the <code>token-value</code> pair will be displayed in the Info column in the Log Viewer.</td>
</tr>
<tr>
<td>-m alertty</td>
<td>The alert to be triggered at the ISP site. This alert overrides the alert specified in the log message generated by the alert daemon.</td>
</tr>
<tr>
<td></td>
<td>The response to the alert is handled according to the actions specified in the ISP's Security Policy:</td>
</tr>
<tr>
<td></td>
<td>The following alerts execute the OS commands defined in the corresponding fields of the Log and Alert tab of the Properties Setup window in Global Properties:</td>
</tr>
<tr>
<td></td>
<td>• <code>alert</code>. Popup alert command.</td>
</tr>
<tr>
<td></td>
<td>• <code>mail</code>. Mail alert command.</td>
</tr>
<tr>
<td></td>
<td>• <code>snmptrap</code>. SNMP trap alert command.</td>
</tr>
<tr>
<td></td>
<td>• <code>spoofalert</code>. Anti-spoof alert command.</td>
</tr>
<tr>
<td></td>
<td>The following NetQuota and ServerQuota alerts execute the OS commands specified in:</td>
</tr>
<tr>
<td></td>
<td><code>$FWDIR/conf/objects.C: value=clientquotaalert.</code></td>
</tr>
<tr>
<td></td>
<td>Parameter=clientquotaalertcmd</td>
</tr>
</tbody>
</table>
Example

```
inet_alert -s 10.0.2.4 -a clear -f product cads -m alert
```

This command specifies that in the event of an attack, \texttt{inet_alert} should take the following actions:

- Establish a clear connection with the ELA Proxy located at IP address \texttt{10.0.2.4}.
- Send a log message to the specified ELA Proxy. The product field of this log message should be set to \texttt{“cads”}. This means that \texttt{“cads”} will be displayed in the \texttt{product} column of the Log Viewer.

Trigger the OS command specified in the \texttt{Popup Alert Command} field of the \texttt{Log and Alert} tab of the \texttt{Properties} Setup window in the Policy Editor.

**ldapcmd**

**Description**

\texttt{ldapcmd} is used to manage processes running on the FireWall-1 collectively or individually. It includes:

**Cache**

cache operations, such as emptying the cache, as well as providing debug information.

**Statistics**

lookup statistics such as, all user search, pending lookups (when two or more lookups are identical) and total lookup time (the total search time for a specific lookup)

cache statistics such as hits and misses

**Logging**

view the alert and warning log regarding debug

**Usage**

```
ldapcmd -p process_name | all command [-d debug_level] [command_arg]
```

where command is:

\begin{tabular}{|c|l|}
\hline
\textbf{exit status} & \textbf{meaning} \\
\hline
104 & Unable to obtain log information from stdin. \\
106 & Invalid command line arguments. \\
107 & Failed to invoke the OPSEC API. \\
\hline
\end{tabular}
- cacheclear (either all or UserCacheObject or TemplateCacheObject or TemplateExtGrpCacheObject)
- cachetrace (either all or UserCacheObject or TemplateCacheObject or TemplateExtGrpCacheObject)
- stat [either print_interval (reset interval time in secs) or 0 (stop statistics)]
- log (either on or off)

### Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p</td>
<td>run a specified process or run all processes</td>
</tr>
<tr>
<td>command</td>
<td>specify a command</td>
</tr>
<tr>
<td>log</td>
<td>specify whether or not to create LDAP logs</td>
</tr>
</tbody>
</table>

### ldapcompare

#### Description

ldapcompare is used to perform compare queries that prints a message whether the result returned a match or not. ldapcompare opens a connection to an LDAP directory server, binds, and performs the comparison specified on the command line or from a specified file.

#### Usage

```bash
ldapcompare -d [options] dn attribute value
```

#### Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>Debug flag</td>
</tr>
<tr>
<td>options</td>
<td>See below.</td>
</tr>
<tr>
<td>dn</td>
<td>The DN object.</td>
</tr>
<tr>
<td>attribute</td>
<td>The attribute of the DN object.</td>
</tr>
<tr>
<td>value</td>
<td>The value of the attribute of the DN object.</td>
</tr>
</tbody>
</table>

The ldapcompare options are as follows:

- -u - Include user-friendly entry names in the output.
- -d <level> - Set LDAP debugging level to “level”.
- -P sep - Print “sep” instead of “=” between attribute names and values.
- -f <file> - Perform sequence of compares listed in “file”.
- -D <binddn> - Bind DN.
-w <passwd> - Bind password (for simple authentication).
-h <host> - LDAP server.
-p <port> - Port on the LDAP server.
-T <timeout> - Client side timeout for all operations (in milliseconds).
-l <time limit> - Server Side time limit (in seconds) for compare.
-z <size limit> - Server Side size limit (in entries) for compare.

---

**Ldapconvert**

**Description**

Ldapconvert is a utility program to port from Member mode to MemberOf mode. This is done by searching all specified group/template entries and fetching their Member attribute values.

Each value is the DN of a member entry. The entry identified by this DN will be added the MemberOf attribute value of the group/template DN at hand. In addition, those Member attribute values will be deleted from the group/template unless Both mode is specified.

While running the program, a log file, named ldapconvert.log, is generated in the current directory, logging all modifications done and errors encountered.

**Usage**

ldapconvert -d -h <host> -p <port> -D user_DN -w <secret> [-g group_DN | -f <file>] -m mem_attr -o memberof_attr -c memberobjectclass[extra options]

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>Debug flag</td>
</tr>
<tr>
<td>-h &lt;host&gt;</td>
<td>LDAP Server IP address.</td>
</tr>
<tr>
<td>-p &lt;port&gt;</td>
<td>LDAP Server port number.</td>
</tr>
<tr>
<td>-D user_DN</td>
<td>LDAP bind DN.</td>
</tr>
<tr>
<td>-w &lt;secret&gt;</td>
<td>LDAP bind password.</td>
</tr>
<tr>
<td>-g group_DN</td>
<td>Group or template DN to perform the conversion on. May appear multiple times for multiple entries.</td>
</tr>
<tr>
<td>-f &lt;file&gt;</td>
<td>File containing a list of group DNs each separated by a new line.</td>
</tr>
</tbody>
</table>
The `ldapconvert` extra options are as follows:

- `-M` - Maximum number of member LDAP updated simultaneously (default is 20).
- `-B` - Convert to Both mode
- `-p <port>` - LDAP port (default is 389).
- `-T <timeout>` - Client side timeout for LDAP operations, in milliseconds: default is “never”.
- `-l <time limit>` - Server side time limit for LDAP operations, in seconds: default is “never”.
- `-s` - Server side size limit for LDAP operations (in entries) (default is “none”).
- `-z` - Use SSL.

**Comments**

It is recommended to backup the LDAP server before running the conversion program in case unrecoverable errors are encountered.

There are two GroupMembership modes: template-to-groups and user-to-groups. It is imperative to keep these modes consistent. For instance, if you apply conversion on LDAP users to include ‘MemberOf’ attributes for their groups, then this conversion should also be applied on LDAP defined templates for their groups.

**Why does a command run with the option `-M` fail?**

The program terminates with an error message stating the connection terminated unexpectedly.
This means that the LDAP server could not handle so many LDAP requests simultaneously and closed the connection. The solution is to run the program again with a lower value for the \(-M\) option (the default value should be adequate but could also cause a connection failure in extreme situation). Continue to reduce the value until the program exits normally. Each time you run the program with the same set of groups the program will pick up where it left off.

**Example**

A group is defined with the DN: `cn=cpGroup, ou=groups, ou=cp, c=il` and the following attributes:

```
... 
  cn=cpGroup
  uniquemember="cn=member1, ou=people, ou=cp, c=il"
  uniquemember=" cn=member2, ou=people, ou=cp, c=il"
  ... 
```

For the 2 member entries:

```
... 
  cn=member1
  objectclass=fw1Person
  ... 
```

and:

```
... 
  cn=member2
  objectclass=fw1Person
  ... 
```

Run `ldapconvert` with the following arguments:

```
ldapconvert -g cn=cpGroup, ou=groups, ou=cp, c=il -h myhost -
d cn=admin -w secret \ -m uniquemember -o memberof -c fw1Person
```

The result for the group DN will be as follows:

```
... 
  cn=cpGroup
  ... 
```

The result for the 2 member entries will be as follows:

```
... 
  cn=member1
  objectclass=fw1Person
  memberof="cn=cpGroup, ou=groups, ou=cp, c=il"
  ... 
```
and

```plaintext
...          
  cn=member2
  objectclass=fw1Person
  memberof=" cn=cpGroup,ou=groups, ou=cp, c=il"
  ...          
```

Running the same command with the `-s` options, will produce the same result but the group entry will not be modified.

If there exists another member attribute value for the same group entry:

```plaintext
uniquemember="cn=template1,ou=people, ou=cp,c=il"
```

and the template is:

```plaintext
  cn=member1
  objectclass=fw1Template
```

after running the same command line the template entry will stay intact because the command line specified the option `-c fw1Person` but the object class of template1 is `fw1Template`.

---

**ldapmodify**

**Description**
ldapmodify imports users to an LDAP server. The input file must be in the LDIF format.

**Usage**

```bash
ldapmodify -a -c -d -h <host> -p <port> -D <LDAPadminDN> -p <LDAPadminPassword> -f <exportfilename>.ldif -d
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-a</code></td>
<td>Add users.</td>
</tr>
<tr>
<td><code>-c</code></td>
<td>Continue on errors.</td>
</tr>
<tr>
<td><code>-h &lt;host&gt;</code></td>
<td>LDAP Server IP address.</td>
</tr>
<tr>
<td><code>-d</code></td>
<td>Debug flag</td>
</tr>
<tr>
<td><code>-p &lt;port&gt;</code></td>
<td>LDAP Server port number.</td>
</tr>
<tr>
<td><code>-D &lt;LDAPadminDN&gt;</code></td>
<td>LDAP Administrator DN.</td>
</tr>
<tr>
<td><code>-p &lt;LDAPadminPassword&gt;</code></td>
<td>LDAP Administrator password.</td>
</tr>
<tr>
<td><code>-f &lt;exportfilename&gt;.ldif</code></td>
<td>Specifies the name of the input file. This file must be in the LDIF format.</td>
</tr>
</tbody>
</table>
Comments  You can import the VPN-1 Pro User Database to an LDAP server by first generating an LDIF file using \texttt{fwm dbexport}, and then using \texttt{ldapmodify}.

Before importing, prepare the LDAP directory as follows:
- Make sure the root branch is defined as an allowed branch on your LDAP server.
- Restart the LDAP server.
- Create the branch into which the users will be imported, either by using \textbf{Create Tree Object} in the Account Management Client or with the \texttt{ldapmodify} command:

  \begin{center}
  \texttt{ldapmodify -a -h <host> -p <port> -D <LDAPadminDN> -w}
  \\
  \texttt{<LDAPadminPassword>}
  \\
  \texttt{dn: o=myOrg,c=US}
  \\
  \texttt{objectclass: organization}
  \\
  \texttt{o=myOrg}
  \end{center}

Example  Importing Users using \texttt{ldapmodify}:

1. Export the users using \texttt{fwm dbexport} using \texttt{hello1234} as the pre-shared secret..

\begin{center}
\texttt{fwm dbexport -l -f ./o_file.ldif -s "o=bigcorp,c=uk" -k hello1234}
\end{center}

2. Create the \texttt{o=bigcorp,c=uk} branch.

3. Import the users:

\begin{center}
\texttt{ldapmodify -a -c -h <host> -p <port> -D bindDN -w bindPas -f}
\texttt{./o_file.ldif}
\end{center}

4. Define an Account Unit with these parameters.
Idapsearch

Description
ldapsearch queries an LDAP directory and returns the results.

Usage
ldapsearch [options] filter [attributes] -d

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>See the options attributes below.</td>
</tr>
<tr>
<td>filter</td>
<td>RFC-1558 compliant LDAP search filter. For example, objectclass=fw1host.</td>
</tr>
<tr>
<td>attributes</td>
<td>The list of attributes to be retrieved. If no attributes are given, all attributes are retrieved.</td>
</tr>
<tr>
<td>-d</td>
<td>Debug flag</td>
</tr>
</tbody>
</table>

The following are the attributes for options:

- `-A` - Retrieve attribute names only (without values).
- `-B` - Do not suppress printing of non-ASCII values.
- `-D bindDN` - The DN to be used for binding to the LDAP Server.
- `-F separator` - Print separator between attribute name and value instead of “=”.
- `-h host` - The LDAP server identified by IP address or resolvable name.
- `-l timelimit` - The server side time limit for search, in seconds.
- `-p portnum` - The port number. The default is standard LDAP port 389.
- `-S attribute` - Sort the results by the values of attribute.
- `-s scope` - One of the following: “base”, “one”, “sub”.
- `-b` - Base distinguished name (DN) for search.
- `-t` - Write values to files in /tmp. Each attribute-value pair is written to a separate file, named: /tmp/ldapsearch-<attribute>-<value>.

For example, for the fw1color attribute, the file written is named /tmp/ldapsearch-fw1color-a00188.

- `-T timeout` - Client-side timeout (in milliseconds) for all operations.
- `-u` - Show “user friendly” entry names in the output. For example, show "cn=Babs Jensen, users, omi" instead of "cn=Babs Jensen, cn=users, cn=omi"
- `-w password` - The password.
-z - Encrypt using SSL.
-z sizelimit - Server-side size limit for search, in entries.

Example
ldapsearch -p 18185 -b cn=omi objectclass=fw1host objectclass

This means that the LDAP directory will be queried for fw1host objects using port number 18185 with DN common name “omi”. For each object found, the value of its objectclass attribute will be printed.

log_export

Description
log_export is a utility that allows you to transfer Log data to an external database. This utility behaves as a LEA client. LEA (Log Export API) enables VPN-1 Pro Log data to be exported to third-party applications. log_export receives the Logs from the SmartCenter Server via LEA so it can be run from any host that has a SIC connection with the SmartCenter Server and is defined as an OPSEC host. To run log_export, you need a basic understanding and a working knowledge of:

- Oracle database administration
- LEA

Usage

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f conf_file</td>
<td>The Configuration File from which log_export reads the Log file parameters. If conf_file is not specified, the default Configuration File log_export.conf, located in the current working directory.</td>
</tr>
<tr>
<td>-l &lt;lea_server_ip_address&gt;</td>
<td>The IP address of the LEA server.</td>
</tr>
<tr>
<td>-g log_file_name,log_file_name,...</td>
<td>A comma separated list of log file names from where the logs will be taken.</td>
</tr>
<tr>
<td>-t &lt;database_table_name&gt;</td>
<td>The name of the table in the database to which the logs will be added.</td>
</tr>
</tbody>
</table>

SmartCenter Commands  97
For more information about LEA, see Check Point LEA (Log Export API) Specification

Comments

Only Oracle database is currently supported.

Before you can run log_export, the Oracle client must be installed and configured. Make sure that:

- the ORACLE_HOME environment variable is set correctly.
- $ORACLE_HOME/lib is located in the PATH environment variable on the NT platform or LD_LIBRARY_PATH on Solaris and Linux platforms.
- If log_export is running from another machine, you must install and configure at least SVN Foundation and Reporting Module.

The log_export Configuration File

log_export has a Configuration File. The Configuration File is a Check Point Set file and should be configured according to Set file conventions. The Configuration File contains the default parameters for log_export. log_export reads all parameters from the Configuration File that is specified in the command line.

Modifying the Configuration File

log_export parameters are defined in the Configuration File. To change the parameters, you can either modify the Configuration File or use the command line. Any parameter entered using the command line will override the parameters in the Configuration File.

Modify the Configuration File according to the following parameters:

- db_connection_string - The string that defines the Oracle database server. For example, the name of the server.
- db_table_name - The name of the table in the database to which the logs will be added.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p &lt;database_password&gt;</td>
<td>The database login password. If you do not want to specify the password in the Configuration File for security reasons, you can enter the password using the command line where it will not be saved anywhere.</td>
</tr>
<tr>
<td>-h</td>
<td>Display log_export usage.</td>
</tr>
<tr>
<td>-d</td>
<td>Display debugging information.</td>
</tr>
</tbody>
</table>

Further Info.
create_db_table - Following are the available options:
  - 1 - create a new table in the database
  - 0 - use the existing table.
If there is an existing table, the logs will be added to that table. This requires that the existing table have the same format as the logs you are adding. If you enter 0 and there is no existing table, you will get an error message. The default is 1.
db_user_name - The database login user name.
db_password - The database login password.
log_server_ip_address - The IP address of the LEA server.
log_server_port - Port number of the LEA server. The default LEA port is 18184.
log_file_name - A list of log file names from where the logs will be taken.
log_fields - The name of the Log file as known by LEA.
db_field_name - The Log field name as represented in the database table.
db_field_type - The Log field type in the database table. This parameter can be one of the following:
  - STRING
  - NUMBER
  - DATE
db_field_size - The size of the field in the database table. This parameter is required only if the db_field_type is either STRING or NUMBER.
Example  Configuration File Example

```
$db_table_name (fw_log)
$db_connection_string (database_service_name)
$db_user_name (scott)
$db_password (tiger)
$log_server_ip_address (127.0.0.1)
$log_server_port (18184)
$create_db_table (1)
$log_file_name (fw.log)
$log_fields {
  :time
    :db_field_name (log_time)
    :db_field_type (DATE)
  }
  :product
    :db_field_name (product)
    :db_field_type (STRING)
    :db_field_size (25)
  }
  :i/f_name
    :db_field_name (interface)
    :db_field_type (STRING)
    :db_field_size (100)
  }
  :orig
    :db_field_name (origin)
    :db_field_type (STRING)
    :db_field_size (16)
  }
  :action
    :db_field_name (action)
    :db_field_type (STRING)
    :db_field_size (16)
  }
  :service
    :db_field_name (service)
    :db_field_type (STRING)
    :db_field_size (40)
}
```

**queryDB_util**

**Description**  queryDB_util enables searching the object database according to search parameters.
Usage

```
queryDB_util [-t <table_name>] [-o <object_name>] [-a]
[-mu <modified_by>] [-mh <modified_from>]
[-ma <modified_after>] [-mb <modified_before>] [-p|m|u|h|t|f]
[-f filename] [-h] [-q]
```

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-t &lt;table_name&gt;</td>
<td>The name of the table.</td>
</tr>
<tr>
<td>-o &lt;object_name&gt;</td>
<td>The name of the object.</td>
</tr>
<tr>
<td>-a</td>
<td>All objects.</td>
</tr>
<tr>
<td>-mu &lt;modified_by&gt;</td>
<td>The name of the administrator who last modified the object.</td>
</tr>
<tr>
<td>-mh &lt;modified_from&gt;</td>
<td>The host from which the object was last modified.</td>
</tr>
<tr>
<td>-ma &lt;modified_after&gt;</td>
<td>The date after which the object was modified &lt;hh:mm:ss&gt;[ddmmmyyyy]. Either or both options may be used. Omitting hh:mm:ss defaults to today at midnight, omitting ddmmmyyyy defaults to today's date on the client.</td>
</tr>
<tr>
<td>-mb &lt;modified_before&gt;</td>
<td>The date before which the object was modified &lt;hh:mm:ss&gt;[ddmmmyyyy]. Either or both options may be used. Omitting hh:mm:ss defaults to today at midnight, omitting ddmmmyyyy defaults to today's date on the client.</td>
</tr>
<tr>
<td>-p</td>
<td>m</td>
</tr>
<tr>
<td></td>
<td>- c -creation details</td>
</tr>
<tr>
<td></td>
<td>- m -last_modification details</td>
</tr>
<tr>
<td></td>
<td>- u - administrator name (create/modify)</td>
</tr>
<tr>
<td></td>
<td>- h -host name (create/modify)</td>
</tr>
<tr>
<td></td>
<td>- t -time (create/modify)</td>
</tr>
<tr>
<td></td>
<td>- f -field details</td>
</tr>
<tr>
<td>-f filename</td>
<td>The name of the output file.</td>
</tr>
<tr>
<td>-h</td>
<td>Display command usage information.</td>
</tr>
<tr>
<td>-q</td>
<td>Quit.</td>
</tr>
</tbody>
</table>
Example

Print modification details of all objects modified by administrator “aa”

```
query> -a -mu Bob -pm
Object Name:my_object
Last Modified by:Bob
Last Modified from:london
Last Modification time:Mon Jun 19 11:44:27 2000

Object Name:internal_ca
Last Modified by:Bob
Last Modified from:london
Last Modification time:Tue Jun 20 11:32:58 2000
```

A total of 2 objects match the query.

---

**rs_db_tool**

**Description**

`rs_db_tool` is used to manage DAIP Modules in a DAIP database.

**Usage**

```
rs_db_tool [-d] <-operation <add <-name object_name> <-ip module_ip> <-TTL Time-To-Live> >
rs_db_tool [-d] <-operation fetch <-name object_name> >
rs_db_tool [-d] <-operation <delete <-name object_name> >
rs_db_tool [-d] <-operation <list> >
rs_db_tool [-d] <-operation <sync> >
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>debug file</td>
</tr>
<tr>
<td>-operation add</td>
<td>Add entry to database.</td>
</tr>
<tr>
<td>&lt;-name object_name&gt;</td>
<td>Enter the name of the module object.</td>
</tr>
<tr>
<td>&lt;-ip module_ip</td>
<td>Enter the IP Address of the module</td>
</tr>
<tr>
<td>&lt;-TTL Time-To-Live&gt;</td>
<td>The relative time interval (in seconds) during which the entry is valid. A value of zero specifies “unlimited”.</td>
</tr>
<tr>
<td>- operation fetch</td>
<td>Get entry from database.</td>
</tr>
</tbody>
</table>
### sam_alert

**Description**
This tool executes FW-1 SAM (Suspicious Activity Monitoring) actions according to information received through Standard input. This tool is for executing FW-1 SAM actions with the FW-1 User Defined alerts mechanism.

**Usage**
```
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-o</td>
<td>Prints the input of this tool to the standard output (for pipes).</td>
</tr>
<tr>
<td>-v</td>
<td>Turns on verbose mode (of the <code>fw sam</code> command).</td>
</tr>
<tr>
<td>-s sam_server</td>
<td>The sam server to be contacted. Localhost is the default.</td>
</tr>
<tr>
<td>-t timeout</td>
<td>The time period, in seconds, for which the action will be enforced. The default is forever.</td>
</tr>
<tr>
<td>-f fw_host</td>
<td>Identifies the FireWalls to run the operation on. Default is “all FireWalls.”</td>
</tr>
<tr>
<td>-C</td>
<td>Cancels the specified operation.</td>
</tr>
<tr>
<td>-n</td>
<td>Notify every time a connection that matches the specified criteria passes the FireWall.</td>
</tr>
<tr>
<td>-i</td>
<td>Inhibit connections that match the specified criteria.</td>
</tr>
<tr>
<td>Argument</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>-I</td>
<td>Inhibit connections that match the specified criteria and close all existing connections that match the criteria.</td>
</tr>
<tr>
<td>-src</td>
<td>Match the source address of connections.</td>
</tr>
<tr>
<td>-dst</td>
<td>Match the destination address of connections.</td>
</tr>
<tr>
<td>-any</td>
<td>Match either the source or destination address of the connection.</td>
</tr>
<tr>
<td>-srv</td>
<td>Match specific source, destination, protocol and service.</td>
</tr>
</tbody>
</table>
FireWall-1 Commands

**comp_init_policy**

**Description**  Use the `comp_init_policy` command to generate and load, or to remove, the Initial Policy.

**Usage**  

```
$FWDIR/bin/comp_init_policy [-u | -g]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-u</code></td>
<td>Removes the current Initial Policy, and ensures that it will not be generated in future when <code>cpconfig</code> is run.</td>
</tr>
<tr>
<td><code>-g</code></td>
<td>Can be used if there is no Initial Policy. If there is, make sure that after removing the policy, you delete the <code>$FWDIR\state\local\FW1\</code> folder. Generates the Initial Policy and ensures that it will be loaded the next time a policy is fetched (at <code>cpstart</code>, or at next boot, or via the <code>fw fetch localhost</code> command). After running this command, <code>cpconfig</code> will add an Initial Policy when needed. The <code>comp_init_policy -g</code> command will only work if there is no previous Policy. If you perform the following commands: <code>comp_init_policy -g + fw fetch localhost</code> <code>comp_init_policy -g + cpstart</code> <code>comp_init_policy -g + reboot</code> The original policy will still be loaded.</td>
</tr>
</tbody>
</table>
fw isp_link

fw

Description The fw commands are used for working with various aspects of FireWall-1. All fw commands are executed on the FireWall-1 enforcement module.

Typing fw at the command prompt sends a list of available fw commands to the standard output.

Usage fw

fw isp_link

Description This command takes down (or up) a redundant ISP link.

Usage fw isp_link [target] link-name {up|down}

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>The name of the enforcement module.</td>
</tr>
<tr>
<td>link-name</td>
<td>The name of the ISP link as defined in the ISP-redundancy tab.</td>
</tr>
</tbody>
</table>

Comments This command can be executed locally on the enforcement module or remotely from the management module. In the latter case, the target argument must be supplied. For this command to work, the enforcement module should be using the ISP redundancy feature.

fw monitor

Description Inspecting network traffic is an essential part of troubleshooting network deployments. fw monitor is a powerful built-in tool to simplify the task of capturing network packets at multiple capture points within the FireWall-1 chain. These packets can be inspected using industry-standard tools later on.

In many deployment and support scenarios capturing network packets is an essential functionality. tcpdump or snoop are tools normally used for this task. fw monitor provides an even better functionality but omits many requirements and risks of these tools.

- No Security Flaws — tcpdump and snoop are normally used with network interface cards in promiscuous mode. Unfortunately the promiscuous mode allows remote attacks against these tools. fw monitor does not use the promiscuous mode to capture packets. In
addition most FireWalls’ operating systems are hardened. In most cases this hardening includes the removal of tools like tcpdump or snoop because of their security risk.

- **Available on all FireWall-1 installations** — *fw monitor* is a built-in firewall tool which needs no separate installation in case capturing packets is needed. It is a functionality provided with the installation of the FireWall package.

- **Multiple capture positions within the FireWall-1 kernel module chain** — *fw monitor* allows you to capture packets at multiple capture positions within the FireWall-1 kernel module chain; both for inbound and outbound packets. This enables you to trace a packet through the different functionalities of the firewall.

- **Same tool and syntax on all platforms** — Another important fact is the availability of *fw monitor* on different platforms. Tools like snoop or tcpdump are often platform dependent or have specific “enhancements” on certain platforms. *fw monitor* and all its related functionality and syntax is absolutely identical across all platforms.

  There is no need to learn any new “tricks” on an unknown platform.

Normally the Check Point kernel modules are used to perform several functions on packets (like filtering, encrypting and decrypting, QoS ...). *fw monitor* adds its own modules to capture packets. Therefore *fw monitor* can capture all packets which are seen and/or forwarded by the FireWall.

Only one instance of *fw monitor* can be run at a time.

Use `^C` (that is Control + C) to stop *fw monitor* from capturing packets.
fw monitor

### Usage

```
```

### Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`-u</td>
<td>s`</td>
</tr>
<tr>
<td><code>-i</code></td>
<td><strong>Flushing the standard output:</strong> Use to make sure that captured data for each packet is at once written to standard output. This is especially useful if you want to kill a running fw monitor process and want to be sure that all data is written to a file.</td>
</tr>
<tr>
<td><code>[-d] [-D]</code></td>
<td><strong>Debugging fw monitor:</strong> The <code>–d</code> option is used to start fw monitor in debug mode. This will give you an insight into fw monitor’s inner workings. This option is only rarely used outside Check Point. It is also possible to use <code>–D</code> to create an even more verbose output.</td>
</tr>
<tr>
<td>`&lt;{-e expr}+</td>
<td>-f &lt;filter-file&gt;</td>
</tr>
</tbody>
</table>
Limiting the packet length: fw monitor allow you to limit the packet data which will be read from the kernel with -l. This is especially useful if you have to debug high sensitive communication. It allows you to capture only the headers of a packet (e.g. IP and TCP header) while omitting the actual payload. Therefore you can debug the communication without seeing the actual data transmitted. Another possibility is to keep the amount of data low. If you don't need the actual payload for debugging you can decrease the file site by omitting the payload. It's also very useful to reduce packet loss on high-loaded machines. fw monitor uses a buffer to transfer the packets from kernel to user space. If you reduce the size of a single packet this buffer won't fill up so fast.

Setting capture masks: By default fw monitor captures packets before and after the virtual machine in both directions. These positions can be changed. This option allows you to specify in which of the four positions you are interested.

Printing packet/payload data: In addition to the IP and Transport header fw monitor can also print the packets’ raw data using the -x option. Optionally it is also possible to send all data that is written only to the screen the data written.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-l len</td>
<td>Limiting the packet length: fw monitor allow you to limit the packet data which will be read from the kernel with -l. This is especially useful if you have to debug high sensitive communication. It allows you to capture only the headers of a packet (e.g. IP and TCP header) while omitting the actual payload. Therefore you can debug the communication without seeing the actual data transmitted. Another possibility is to keep the amount of data low. If you don't need the actual payload for debugging you can decrease the file site by omitting the payload. It's also very useful to reduce packet loss on high-loaded machines. fw monitor uses a buffer to transfer the packets from kernel to user space. If you reduce the size of a single packet this buffer won’t fill up so fast.</td>
</tr>
<tr>
<td>-m mask</td>
<td>Setting capture masks: By default fw monitor captures packets before and after the virtual machine in both directions. These positions can be changed. This option allows you to specify in which of the four positions you are interested.</td>
</tr>
<tr>
<td>-x offset[,]len</td>
<td>Printing packet/payload data: In addition to the IP and Transport header fw monitor can also print the packets’ raw data using the -x option. Optionally it is also possible to send all data that is written only to the screen the data written.</td>
</tr>
</tbody>
</table>
**Argument** | **Description**  
---|---  
-o <file> | **Write output to file:** Save the raw packet data to a file in a standard (RFC 1761) format. The file can be examined using by tools like snoop, tcpdump or Ethereal.  
  
  **Note:** The snoop file format is normally used to store Layer 2 frames. For “normal” capture files this means that the frame includes data like a source and a destination MAC address. fw monitor operates in the FireWall-1 kernel and therefore has no access to Layer 2 information like MAC addresses. Instead of writing random MAC addresses, fw monitor includes information like interface name, direction and chain position as “MAC addresses”.  

-T | Print time stamp in microseconds. -T is needed only when -o is not used. When -o is used the exact time is written to the snoop file by default as Corsica.  

<[-pi pos] [-pI pos] [-po pos] [-pO pos] | **Insert fw monitor chain module at a specific position:** In addition to capture masks (which give the ability to look at packets in a specific position) fw monitor has the ability to define where exactly in the FireWall-1 chain the packets should be captured. This can be defined using these options.
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td><strong>Use absolute chain positions:</strong> If you use <code>fw monitor</code> to output the capture into a file (option <code>-o</code>), one of the fields written down to the capture file is the chain position of the <code>fw monitor</code> chain module. Together with a simultaneous execution of <code>fw ctl chain</code> you can determine where the packet was captured. Especially when using <code>-p all</code> you will find the same packet captured multiples times at different chain positions. The option <code>-a</code> changes the chain id from an relative value (which only makes sense with the matching <code>fw ctl chain</code> output) to an absolute value. These absolute values are known to CPEthereal and can be displayed by it.</td>
</tr>
</tbody>
</table>
The easiest way to use `fw monitor` is to invoke it without any parameter. This will output every packet from every interface that passes (or at least reaches) the enforcement module. Please note that the same packet is appearing several times (two times in the example below). This is caused by `fw monitor` capturing the packets at different capture points.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>[-ci count] [-co count]</code></td>
<td><strong>Capture a specific number of packets:</strong> <code>fw monitor</code> enables you to limit the number of packets being captured. This is especially useful in situations where the firewall is filtering high amounts of traffic. In such situations <code>fw monitor</code> may bind so many resources (for writing to the console or to a file) that recognizing the break sequence (Control-C) might take very long.</td>
</tr>
<tr>
<td><code>[-vs vsid or vsname]</code></td>
<td><strong>Capture on a specific Virtual Router or Virtual Machine:</strong> <code>FireWall-1 VSX</code> enables you to run multiple Virtual Routers and FireWalls on one physical machine. Using the option <code>--vs</code> you can specify on which virtual component the packets should be captured. This option is only available on a <code>FireWall-1 VSX</code> module. Please refer to <code>fw monitor</code> on <code>FireWall-1 VSX</code> for more information.</td>
</tr>
<tr>
<td><code>-h</code></td>
<td>Displays the usage.</td>
</tr>
</tbody>
</table>

**Example**

The easiest way to use `fw monitor` is to invoke it without any parameter. This will output every packet from every interface that passes (or at least reaches) the enforcement module. Please note that the same packet is appearing several times (two times in the example below). This is caused by `fw monitor` capturing the packets at different capture points.
The first line of the `fw monitor` output is

```
eth0:i[285]: 172.16.1.133 -> 172.16.1.2 (TCP) len=285 id=1075
TCP: 1050 -> 18190 ...PA. seq=bf8bc98e ack=941b05bc
```

This packet was captured on the first network interface (eth0) in inbound direction before the virtual machine (lowercase i). The packet length is 285 bytes (in square parenthesis; repeated at the end of the line). Note that these two values may be different. The packets ID is 1075. The packet was sent from 172.16.1.133 to 172.16.1.2 and carries a TCP header/payload.

The second line of the `fw monitor` output is

```
TCP: 1050 -> 18190 ...PA. seq=bf8bc98e ack=941b05bc
```

The second line tells us that this is an TCP payload inside the IP packet which was sent from port 1050 to port 18190. The following element displays the TCP flags set (in this case PUSH and ACK). The last two elements are showing the sequence number (seq=bf8bc98e) of the TCP packet and the acknowledged sequence number (ack=941b05bc). You will see similar information for UDP packets.
You will only see a second line if the transport protocol used is known to fw monitor. Known protocols are for example TCP, UDP and ICMP. If the transport protocol is unknown or can not be analyzed because it is encrypted (e.g. ESP or encapsulated (e.g. GRE) the second line is missing.

**Further Info.** See the document *How to use fw monitor* at [http://www.checkpoint.com/techsupport/downloadsng/utilities.html](http://www.checkpoint.com/techsupport/downloadsng/utilities.html).
fw tab

Description
The fw tab command enables you to view kernel table contents and change them (that is, only dynamic tables since the content of a static table is indeed static).

Usage

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-t &lt;table&gt;</td>
<td>Specifies a table for the command.</td>
</tr>
<tr>
<td>-s</td>
<td>Displays a short summary of the table (s) information.</td>
</tr>
<tr>
<td>-y</td>
<td>Specifies to not prompt a user before executing any commands.</td>
</tr>
<tr>
<td>-f</td>
<td>Displays a formatted version of the table content. Every table may have its own specific format style.</td>
</tr>
<tr>
<td>-o &lt;filename&gt;</td>
<td>Dumps CL formatted output to filename, which can later be read by fw log or any other entity that can read FW log formats.</td>
</tr>
<tr>
<td>-c</td>
<td>Displays formatted table informatin in common format.</td>
</tr>
<tr>
<td>-r</td>
<td>Resolves IP addresses in formatted output.</td>
</tr>
<tr>
<td>-x, -a, -e</td>
<td>It is possible to add or remove an entry from an existing dynamic table by using the -a or the -x flags, respectively. These flags must be followed by the -e flag and an entry description (&lt;entry&gt;).</td>
</tr>
</tbody>
</table>

[hostname] A list of one or more targets. When not used, the local machine is used as the default target.

Example
fw tab -t <table-name> -a -e “1,2;3,4,5” or
fw tab -t <table-name> -a -e “<1,2;3,4,5>”
Adds an entry:
<00000001,00000002,00000003,00000004,00000005,>to<table-name>

fw tab -t <table-name> -a -e “1,2,” or
fw tab -t <table-name> -a -e “<1,2>”
Adds an entry with only a key field: <00000001,00000002>
If table<table-name> contains the following entry:
<000000000000001,00000002>
fw tab -t <table-name> -x -e “0,1” or
fw tab -t <table-name> -x -e “0,1;2”

Removes the entry from the specified table.

Comments If table has the ‘expire’ attribute, entries added using the -a flag will receive the default table timeout.
This feature only works on local machine kernel tables and does not work on a remote machine’s tables like additional fw tab commands.
The -x flag can be used independantly of the -e flag in which case the entire table content is deleted.
This feature should only be used for debug purposes. It is not advisable to arbitrarily change the content of any kernel table since doing so may have unexpected results including unexpected security and connectivity impacts.

fw stat

Description State tables are used to keep state information which the FireWall-1 virtual machine, and other components of FireWall-1 need in order to correctly inspect the packet. The tables are actually the 'memory' of the virtual machine in the kernel, and are the key component of Check Point Stateful Inspection technology. State tables are implemented as dynamic hash tables in kernel memory. All field values are in hexadecimal, apart from the time-out value at the end of the entry, when present.

The fw tab command displays the content of state tables on the target hosts in various formats. For each host, the default format displays the host name and a list of all tables with their elements.

Usage
fw tab [-all |-conf conffile] [-s][-m number][-u][-t tname][-x tname][-d] <targets>
Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-all</td>
<td>The command is to be executed on all targets specified in the default system configuration file ($FWDIR/conf/sys.conf).</td>
</tr>
<tr>
<td>-conf conffile</td>
<td>The command is to be executed on the targets specified in conffile.</td>
</tr>
<tr>
<td>-s</td>
<td>Summary of the number of entries in each table: host name, table name, table ID, and its number of entries.</td>
</tr>
<tr>
<td>-m number</td>
<td>For each table, display only its first number of elements (default is 16 entries at most).</td>
</tr>
<tr>
<td>-u</td>
<td>Do not limit the number of entries displayed for each table.</td>
</tr>
<tr>
<td>-t tname</td>
<td>Display only tname table.</td>
</tr>
<tr>
<td>-x tname</td>
<td>Delete all entries in all tables</td>
</tr>
<tr>
<td>-d</td>
<td>Debug mode</td>
</tr>
<tr>
<td>targets</td>
<td>The command is executed on the designated targets.</td>
</tr>
</tbody>
</table>

A table has a list of associated attributes.

Example

To display only the arp_table table,

```
fw tab -t arp_table
fw sam -C -i src louvre
```

fw ver

**Description**

This command displays the VPN-1 Pro major and minor version number and build number.

**Usage**

```
fw ver [-k][-f <filename>]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-k</td>
<td>Print the version name and build number of the Kernel module.</td>
</tr>
<tr>
<td>-f &lt;filename&gt;</td>
<td>Print the version name and build number to the specified file.</td>
</tr>
</tbody>
</table>
svr_webupload_config

**Description**  
This utility is used to configure the SmartView Reporter web upload script. For the complete upload procedure and additional information refer to the section “How to upload reports to a web server” in the SmartView Reporter User Guide.

**Usage**  
svr_webupload_config [-i perl_int_loc] [-p rep_dir_root]

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i</td>
<td>Specifies the Perl interpreter location.</td>
</tr>
<tr>
<td>-p</td>
<td>Specifies the path for the reports virtual directory.</td>
</tr>
</tbody>
</table>
VPN-1 Commands

**VPN**

**Description**  This command and subcommands are used for working with various aspects of VPN-1. VPN commands executed on the command line generate status information regarding VPN processes, or are used to stop and start specific VPN services. All VPN commands are executed on the VPN-1 Pro module. The vpn command sends to the standard output a list of available commands.

**Usage**  vpn

**Comments**  Sends to the standard output a list of available commands.

**vpn accel**

**Description**  This command performs operations on VPN accelerator cards (encryption only cards, not the full SecureXL cards) and VPNx. VPNx is a software module that takes advantage of multiple CPUs to accelerate VPN operations. The command comes in three flavours -- for turning the accelerator card on and off, for collecting statistics, and enabling or disabling the accelerator card or acceleration software.

**Usage**

```
vpn accel [-d vpnx] on|off
vpn accel [-d vpnx] stat[-l]
vpn accel -d vpnx autostart on|off
```
Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autostart on/off</td>
<td>Automatically starts/stops the vpnx accelerator software</td>
</tr>
<tr>
<td>on/off</td>
<td>Enable/disable accelerator card or vpnx accelerator module</td>
</tr>
<tr>
<td>stat [-l]</td>
<td>Reports the status of the accelerator card in long format</td>
</tr>
</tbody>
</table>

Example

vpn accel -d vpnx stat

VPN-1: VPNx started
  Number of initialization errors: 0
  Number of processing errors: 0

vpn accel -d vpnx stat -l
VPN-1: VPNx started
  Number of initialization errors: 0
  Number of processing errors: 0
  Number of ESP valid contexts: 0
  Number of packets queued to the accelerator: 0
  High water mark of number of packets in queue: 1
Example

```
vpn accel -d vpnx stat -l
```

Output

```
VPN-1: VPNx started
Number of initialization errors: 0
Number of processing errors: 0

vpn accel -d vpnx stat -l
VPN-1: VPNx started
Number of initialization errors: 0
Number of processing errors: 0
Number of ESP valid contexts: 0
Number of packets queued to the accelerator: 0
High water mark of number of packets in queue: 1

Number of packets and bytes since last activation

<table>
<thead>
<tr>
<th>Packets</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESP decrypted</td>
<td>52</td>
</tr>
<tr>
<td>ESP encrypted</td>
<td>52</td>
</tr>
<tr>
<td>ESP total</td>
<td>104</td>
</tr>
<tr>
<td>Total</td>
<td>104</td>
</tr>
</tbody>
</table>

Average rates for the last 42.343 seconds

<table>
<thead>
<tr>
<th>Packets/sec</th>
<th>Kbit/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESP decrypted</td>
<td>0</td>
</tr>
<tr>
<td>ESP encrypted</td>
<td>0</td>
</tr>
<tr>
<td>ESP total</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>0</td>
</tr>
</tbody>
</table>
```

---

**vpn compreset**

**Description**

This command resets the compression/decompression statistics to zero.

**Usage**

```
vpn compreset
```

**Comments**

Run this command before running **vpn compstat**. This command is mostly obsolete. More compression/decompression information is available via **cpstat**.
**vpn compstat**

**Description**
This command displays compression/decompression statistics

**Usage**
vpn compstat

**Comments**
This command is mostly obsolete. More compression/decompression information is available via cpstat.

---

**vpn crl_zap**

**Description**
This command is used to erase all Certificate Revocation Lists (CRLs) from the cache.

**Usage**
vpn crl_zap

**Return Value**
0 for success; any other value equals failure.

---

**vpn crlview**

**Description**
This command retrieves the Certificate Revocation List (CRL) from various distribution points and displays it for the user. The command comes in three flavors:

- `vpn crlview -obj <MyCA> -cert <MyCert>`: The VPN daemon contacts the Certificate Authority called MyCA and locates the certificate called MyCert. The VPN daemon extracts the certificate distribution point from the certificate then goes to the distribution point, which might be an LDAP or HTTP server. From the distribution point, the VPN daemon retrieves the CRL and displays it to the standard output.

- `vpn crlview -f d:\temp\MyCert`: The VPN daemon goes to the specified directory, extracts the certificate distribution point from the certificate, goes to the distribution point, retrieves the CRL, and displays the CRL to the standard output.

- `vpn crlview -view <lastest_CRL>`: If the CRL has already been retrieved, this command instructs the VPN daemon to display the contents to the standard output.

**Usage**

- `vpn crlview -obj <object name> -cert <certificate name>`
- `vpn crlview -f <filename>`
- `vpn crlview -view`
VPN debug

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-obj -cert</td>
<td>-obj refers to the name of the CA network object</td>
</tr>
<tr>
<td></td>
<td>-cert refers to the name of the certificate</td>
</tr>
<tr>
<td>-f</td>
<td>Refers to the filename of the certificate</td>
</tr>
<tr>
<td>-view</td>
<td>Views the CRL</td>
</tr>
<tr>
<td>-d</td>
<td>Debug option</td>
</tr>
</tbody>
</table>

Return Value 0 for success; any other value equals failure.

vpn debug

Description

This command instructs the VPN daemon to write debug messages to the VPN log file: in $FWDIR/log/vpnd.elg. Debugging of the VPN daemon takes place according to topics and levels. A topic is a specific area on which to perform debugging, for example if the topic is LDAP, all traffic between the VPN daemon and the LDAP server are written to the log file. Levels range from 1-5, where 5 means “write all debug messages”.

This command makes use of TdError, a Check Point infrastructure for reporting messages and debug information. There is no legal list of topics. It depends on the application or module being debugged.

To debug all available topics, use: ALL for the debug topic.

IKE traffic can also be logged. IKE traffic is logged to $FWDIR/log/IKE.elg

Usage

Usage: vpn debug < on [ DEBUG_TOPIC=level ] | off | ikeon | ikeoff | trunc | timeon <SECONDS>| timeoff

vpn debug on DEBUG_TOPIC=level |off timeon<SECONDS>|timeoff

vpn debug ikeon | ikeoff timeon|timeoff

vpn debug trunc
### vpn drv

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>on</td>
<td>Turns on high level vpn debugging.</td>
</tr>
<tr>
<td>on topic=level</td>
<td>Turns on the specified debug topic on the specified level. Log messages</td>
</tr>
<tr>
<td></td>
<td>associated with this topic at the specified level (or higher) are sent to</td>
</tr>
<tr>
<td></td>
<td>$FWDIR/log/vpnd.elg</td>
</tr>
<tr>
<td>off</td>
<td>Turns off all vpn debugging.</td>
</tr>
<tr>
<td>timeon/timeoff</td>
<td>Number of seconds to run the debug command</td>
</tr>
<tr>
<td>ikeon</td>
<td>Turns on IKE packet logging to: $FWDIR/log/IKE.elg</td>
</tr>
<tr>
<td>ikeoff</td>
<td>Turns off IKE logging</td>
</tr>
<tr>
<td>trunc</td>
<td>Truncates the $FWDIR/log/IKE.elg file, switches the cyclic vpnd.elg file,</td>
</tr>
<tr>
<td></td>
<td>switches the cyclic vpnd.elg (changes the current vpnd.elg file to</td>
</tr>
<tr>
<td></td>
<td>vpnd0.elg and creates a new vpnd.elg), enables vpnd and ike debugging and</td>
</tr>
<tr>
<td></td>
<td>adds a timestamp to the vpnd.elg file.</td>
</tr>
</tbody>
</table>

**Return Value**

0= success, failure is some other value, typically -1 or 1.

**Example**

```
vpn debug on all=5 timeon 5.
```

This writes all debugging information for all topics to the vpnd.elg file for five seconds.

**Comments**

IKE logs are analyzed using the support utility IKEView.exe.

### vpn drv

**Description**

This command installs the VPN-1 kernel (vpnk) and connects to the FireWall-1 kernel (fwk), attaching the VPN-1 driver to the FireWall-1 driver.

**Usage**

```
vpn drv on|off

vpn drv stat
```

124 VPN
**vpn export_p12**

**Description**  
This command exports information contained in the network objects database and writes it in the PKCS#12 format to a file with the p12 extension.

**Usage**  
vpn export_p12 -obj <network object> -cert <certificate object> -file <filename> -passwd <password>

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>on/off</td>
<td>Starts/stops the VPN-1 kernel</td>
</tr>
<tr>
<td>stat</td>
<td>Returns the status of the VPN-1 kernel, whether the kernel is on or off</td>
</tr>
</tbody>
</table>

**Return Value**  
0 for success; any other value equals failure.

**Example**  
vpn export_p12 -obj Gateway1 -cert MyCert -file mycert.p12 -passwd kdd432

---

**vpn macutil**

This command is related to Remote Access VPN, specifically Office mode, generating a MAC address per remote user. This command is relevant only when allocating IP addresses via DHCP.

Remote access users in Office mode receive an IP address which is mapped to a hardware or MAC address. This command displays a generated hardware or MAC address for each name you enter.

**Usage**  
vpn macutil <username>

**Example**  
vpn macutil John

**Output**  
20-0C-EB-26-80-7D, "John"
**vpn nssm_toplogy**

**Description**  
This command generates and uploads a topology (in NSSM format) to a Nokia NSSM server for use by Nokia clients.

**Usage**  
```
vpn nssm_toplogy -url <"url"> -dn <"dn"> -name <"name"> -pass <"password"> [-action <bypass|drop>] [-print_xml]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-url</td>
<td>URL of the Nokia NSSM server</td>
</tr>
<tr>
<td>-dn</td>
<td>Distinguished name of the NSSM server needed to establish an SSL connection</td>
</tr>
<tr>
<td>-name</td>
<td>Valid Login name for NSSM Server</td>
</tr>
<tr>
<td>-pass</td>
<td>Valid password for NSSM Server</td>
</tr>
<tr>
<td>-action</td>
<td>Specifies the action the symbian client should take if the packet is not destined for an IP address in the VPN domain. Legal options are Bypass (default) or Drop</td>
</tr>
<tr>
<td>-print_xml</td>
<td>The topology is in XLM format. This flag writes that topology to a file in XLM format.</td>
</tr>
</tbody>
</table>

**vpn overlap_encdom**

**Description**  
This command displays all overlapping VPN domains. Some IP addresses might belong to two or more VPN domains. The command alerts for overlapping encryption domains if one or both of the following conditions exist:

- The same VPN domain is defined for both Gateway
- If the Gateway has multiple interfaces, and one or more of the interfaces has the same IP address and netmask.

If the Gateway has multiple interfaces, and one or more of the interfaces have the same IP address and netmask

**Usage**  
```
vpn overlap_encdom [communities | traditional]
```
Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communities</td>
<td>With this flag, all pairs of objects with overlapping VPN domains are displayed -- but only if the objects (that represent VPN sites) are included in the same VPN community. This flag is also used if the same destination IP can be reached via more than one community.</td>
</tr>
<tr>
<td>Traditional</td>
<td>Default flag. All pairs of objects with overlapping VPN domains are displayed.</td>
</tr>
</tbody>
</table>

Example

```
vpn overlap_encdom communities
```

Output

```
c:/> vpn overlap_encdom communities
The objects Paris and London have overlapping encryption domains.
The overlapping domain is:
10.8.8.1 - 10.8.8.1
10.10.8.0 - 10.10.9.255
- This overlapping encryption domain generates a multiple entry points configuration in MyIntranet and RemoteAccess communities.
- Same destination address can be reached in more than one community (Meshed, Star). This configuration is not supported.

The objects Paris and Chicago have overlapping encryption domains. The overlapping domain is:
10.8.8.1 - 10.8.8.1
- Same destination address can be reached in more than one community (MyIntranet, NewStar). This configuration is not supported.

The objects Washington and Tokyo have overlapping encryption domains.
The overlapping domain is:
10.12.10.68 - 10.12.10.68
- This overlapping encryption domain generates a multiple entry points configuration in Meshed, Star and NewStar communities.
```

**vpn sw_topology**

**Description**
This command downloads the topology for a SofaWare Gateway.

**Usage**

```
vpn [-d] sw_topology -dir <directory> -name <name> -profile <profile> [-filename <filename>]
```
**vpn ver**

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>Debug flag</td>
</tr>
<tr>
<td>-dir</td>
<td>Output directory for file</td>
</tr>
<tr>
<td>-name</td>
<td>Nickname of site which appears in remote client</td>
</tr>
<tr>
<td>-profile</td>
<td>Name of the sofaware profile for which the topology is created</td>
</tr>
<tr>
<td>-filename</td>
<td>Name of the output file</td>
</tr>
</tbody>
</table>

**Description**

This command displays the VPN-1 major version number and build number.

**Usage**

vpn ver [-k] -f <filename>

---

**vpn tu**

**Description**

This command launches the TunnelUtil tool which is used to control VPN tunnels.

**Usage**

vpn tu

vpn tunnelutil

**Example**

vpn tu
**Output**

<table>
<thead>
<tr>
<th>************</th>
<th>Select Option</th>
<th>************</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>List all IKE SAs</td>
<td></td>
</tr>
<tr>
<td>(2)</td>
<td>List all IPsec SAs</td>
<td></td>
</tr>
<tr>
<td>(3)</td>
<td>List all IKE SAs for a given peer</td>
<td></td>
</tr>
<tr>
<td>(4)</td>
<td>List all IPsec SAs for a given peer</td>
<td></td>
</tr>
<tr>
<td>(5)</td>
<td>Delete all IPsec SAs for a given peer</td>
<td></td>
</tr>
<tr>
<td>(6)</td>
<td>Delete all IPsec+IKE SAs for a given peer</td>
<td></td>
</tr>
<tr>
<td>(7)</td>
<td>Delete all IPsec SAs for ALL peers</td>
<td></td>
</tr>
<tr>
<td>(8)</td>
<td>Delete all IPsec+IKE SAs for ALL peers</td>
<td></td>
</tr>
<tr>
<td>(A)</td>
<td>Abort</td>
<td></td>
</tr>
</tbody>
</table>

In Progress ...

**ALL IKE SA**

---

Peer: 194.29.40.225  Cookies ebc5cf1c68c2925b27cb65c1af28bc6

Peer: 194.29.40.225  Cookies 8670f30aa0a4a304672a6998758071d

Hit <Enter> key to continue ...

**Further Info.** When viewing Security Associations for a specific peer, the IP address must be given in dotted decimal notation.
VPN

130 VPN
SmartView Monitor Commands

**RTM**

**Description**
This command and all its derivatives are used to execute SmartView Monitor operations.

**rtm debug**

**Description**
This command sends debug printouts to the $FWDIR/log/rtmd.elg file.

**Usage**
```
rtm debug <on | off> [OPSEC_DEBUG_LEVEL | TDERROR_<AppName>_<Topic>=<ErrLevel>]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>on</td>
<td>Start debug mode</td>
</tr>
<tr>
<td>off</td>
<td>Stop debug mode</td>
</tr>
<tr>
<td>OPSEC_DEBUG_LEVEL</td>
<td>Turn on OPSEC debug printouts</td>
</tr>
<tr>
<td>TDERROR_RTM_ALL</td>
<td>Turn on SmartView Monitor debug printouts</td>
</tr>
</tbody>
</table>

**rtm drv**

**Description**
This command starts, stops or checks the status of the SmartView Monitor kernel driver.

**Usage**
```
rtm drv <on | off | stat>
```
rtm monitor

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>on</td>
<td>Start the SmartView Monitor kernel driver</td>
</tr>
<tr>
<td>off</td>
<td>Stop the SmartView Monitor kernel driver</td>
</tr>
<tr>
<td>stat</td>
<td>SmartView Monitor kernel driver status</td>
</tr>
</tbody>
</table>

**Description**

This command starts the monitoring process and specifies parameters for monitoring an interface.

**Usage**

```
rtm monitor <module_name><interface_name>[options]-g<grouping>[entity-1...entity-n]
or
rtm monitor <module_name>-v<virtual_link_name>[options]entity-1...entity-n
or
rtm monitor <module_name>-filter["complex filter"][options]-g<grouping>[entity-1...entity-n]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>&lt;aggregate</td>
</tr>
<tr>
<td>-w</td>
<td>&lt;bandwidth</td>
</tr>
<tr>
<td>-t</td>
<td>&lt;wire</td>
</tr>
<tr>
<td>-i</td>
<td>&lt;number of seconds&gt;</td>
</tr>
<tr>
<td>@@</td>
<td>specifies subrule (for example, 'rule@@subrule')</td>
</tr>
<tr>
<td>default values</td>
<td>-y bytes -a aggregate -w bandwidth -i2</td>
</tr>
<tr>
<td>grouping types</td>
<td>svc</td>
</tr>
<tr>
<td>module-name</td>
<td>The name of the SmartView Monitor module.</td>
</tr>
<tr>
<td>interface-name</td>
<td>The name of the monitored interface.</td>
</tr>
<tr>
<td>Argument</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>-d</td>
<td>Specifies one of the following monitor directions:</td>
</tr>
<tr>
<td></td>
<td>- inbound</td>
</tr>
<tr>
<td></td>
<td>- outbound</td>
</tr>
<tr>
<td></td>
<td>- eitherbound</td>
</tr>
<tr>
<td>inbound</td>
<td>Monitors the inbound direction.</td>
</tr>
<tr>
<td>outbound</td>
<td>Monitors the outbound direction.</td>
</tr>
<tr>
<td>eitherbound</td>
<td>Monitors both directions.</td>
</tr>
<tr>
<td>-y</td>
<td>Specifies one of the following measurement units:</td>
</tr>
<tr>
<td></td>
<td>- bytes</td>
</tr>
<tr>
<td></td>
<td>- pkts</td>
</tr>
<tr>
<td></td>
<td>- line</td>
</tr>
<tr>
<td>c</td>
<td>Indicates the number of new connections opened per second.</td>
</tr>
<tr>
<td>C</td>
<td>Average concurrent connections</td>
</tr>
<tr>
<td>-a</td>
<td>Aggregate - displays a specific type of connections as an aggregate.</td>
</tr>
<tr>
<td></td>
<td>Individual - displays a specific type of connections as an individual. The default is eitherbound.</td>
</tr>
<tr>
<td>-g</td>
<td>Specifies one of the following grouping options for monitored traffic:</td>
</tr>
<tr>
<td></td>
<td>- svc</td>
</tr>
<tr>
<td></td>
<td>- src</td>
</tr>
<tr>
<td></td>
<td>- dst</td>
</tr>
<tr>
<td></td>
<td>- ip</td>
</tr>
<tr>
<td></td>
<td>- fgrule</td>
</tr>
<tr>
<td></td>
<td>- topsvc</td>
</tr>
<tr>
<td></td>
<td>- topsrc</td>
</tr>
<tr>
<td></td>
<td>- topdst</td>
</tr>
<tr>
<td></td>
<td>- topfwm</td>
</tr>
<tr>
<td></td>
<td>- topfgrule</td>
</tr>
<tr>
<td>Argument</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>svc</td>
<td>Monitors according to a service.</td>
</tr>
<tr>
<td>src</td>
<td>Monitors according to a network object (source only).</td>
</tr>
<tr>
<td>dst</td>
<td>Monitors according to a network object (destination only).</td>
</tr>
<tr>
<td>ip</td>
<td>Monitors according to a network object (source and destination).</td>
</tr>
<tr>
<td>fgrule</td>
<td>Monitors according to a QoS Policy rule.</td>
</tr>
<tr>
<td>topsvc</td>
<td>Monitors the traffic of the top 50 services.</td>
</tr>
<tr>
<td>topsrc</td>
<td>Monitors the traffic of the top 50 sources.</td>
</tr>
<tr>
<td>topdst</td>
<td>Monitors the traffic of the top 50 destinations.</td>
</tr>
<tr>
<td>topdston</td>
<td>Monitors traffic to and from the top 50 IP addresses (source of destination).</td>
</tr>
<tr>
<td>topfwn</td>
<td>Monitors according to the top 50 FireWall-1 rules.</td>
</tr>
<tr>
<td>topfgrule</td>
<td>Monitors according to the top 50 QoS Policy rules.</td>
</tr>
<tr>
<td>-p</td>
<td>Specifies whether or not thousands will be separated by commas.</td>
</tr>
<tr>
<td>-filter</td>
<td>[&quot;&lt;complex filter&gt;&quot;] Only monitors traffic that matches the complex - filter boolean expression.</td>
</tr>
</tbody>
</table>

**Example**

The following command line displays monitoring data in bytes-per-sec for the top 50 services passed on any interface in both directions:

```
rtm monitor localhost -filter -g topsvc
```
The following command will display monitoring data in Conncurrent-Connections for the top 50 sources passed on interface eth0, inbound (that is, not telnet of http).

```
rtm monitor localhost -filter "[and[[interface 0 [[eth0in]]][svc 1 [telnet http]]]]" -y C -g topsrc
```

The default monitors all traffic on any interface in both directions.

**Comments**
The specified entities should correspond to the specified grouping option. For example, if the monitoring process works according to a service (svc), all of the monitored services should be listed and separated by single spaces.

When monitoring occurs according to the QoS Policy rule (fgrule), ‘rule@@subrule’ should be used to specify a subrule entity.

There is no need to specify the top grouping options since they automatically monitor the top 50 entities according to the specified group.

**Example**
The following command displays monitoring data in bytes-per-sec for the top 50 services passed on interface hme1.

```
rtm monitor localhost hme1 -g topsvc -y b
```

---

**rtm monitor**

**Description**
This command starts the monitoring process and specifies parameters for monitoring a Virtual Link.

**Usage**
```
rtm monitor
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>module-name</td>
<td>The name of the SmartView Monitor module.</td>
</tr>
<tr>
<td>-virtual-link-name</td>
<td>The name of the monitored Virtual Link.</td>
</tr>
<tr>
<td>-d</td>
<td>Specifies one of the following monitoring directions:</td>
</tr>
<tr>
<td></td>
<td>- a2b</td>
</tr>
<tr>
<td></td>
<td>- b2a</td>
</tr>
<tr>
<td></td>
<td>- a2b_b2a</td>
</tr>
</tbody>
</table>
## Description
This command starts the SmartView Monitor daemon manually. This also occurs manually when rtmstart is run.

## Usage
```bash
rtm rtmd
```
**rtm stat**

**Description**
This command displays the general SmartView Monitor status. In addition, it displays the status of the daemon, driver, opened views and active virtual links.

**Usage**
```
rtm stat [flavor(s)] [-h] [-v[v][v]]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-h</td>
<td>Help</td>
</tr>
<tr>
<td>-v</td>
<td>Verbose</td>
</tr>
<tr>
<td>vl</td>
<td>Current virtual links</td>
</tr>
<tr>
<td>view</td>
<td>Current views</td>
</tr>
</tbody>
</table>

**rtm ver**

**Description**
This command displays the SmartView Monitor version.

**Usage**
```
rtm ver [-k]
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-k</td>
<td>Displays the SmartView Monitor kernel version.</td>
</tr>
</tbody>
</table>

**rtmstart**

**Description**
This command loads the SmartView Monitor kernel module and starts the SmartView Monitor daemon.

**Usage**
```
rtmstart
```

**rtmstop**

**Description**
This command kills the SmartView Monitor daemon and unloads the SmartView Monitor kernel module.

**Usage**
```
rtmstop
```
SecureClient Commands

SecureClient Commands

**SCC**

**Description**
VPN commands executed on SecureClient are used to generate status information, stop and start services, or connect to defines sites using specific user profiles. Typically, a SecureClient does not need to shell out to a command prompt and use these commands but the site administrator may wish to include them in a script which is then transferred to remote users. In this way, the SecureClient CLI exposes SecureClient operations (such as Connect/Disconnect) to external third party applications via scripting.

The general format for SecureClient commands is:

```
C:> scc <command> [optional arguments]
```

Some of the commands have keyboard shortcuts. Some of the commands require you to be in command line mode. Use the `setmode` command for switching to command line mode. Once in CLI mode, the system tray SecureClient icon is disabled.

**Return Value**
All the scc commands return 0 on success and [-1] on error. Any textual output goes to stdout on success (for example:'scc numprofiles'), and any error string goes to stderr.

---

**scc connect**

**Description**
This command connects to the site using the specified profile, and waits for the connection to be established. In other words, the OS does not put this command into the background and executes the next command in the queue.

**Usage**
```
connect [-p] <profilename>
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p</td>
<td>Displays connection progress</td>
</tr>
</tbody>
</table>

**Comments**
Shortcut: `scc c`

You must be in CLI mode to run this command.
**scc connectnowait**

**Description**  This command connects asynchronously to the site using the specified profile. This means, the OS moves onto the next command in the queue and this command is run in the background.

**Usage**  
```shell
connectnowait <profilename>
```

**Comments**  Shortcut: `scc cn`

You must be in CLI mode to run this command.

**scc disconnect**

**Description**  This command disconnects from the site using a specific profile.

**Usage**  
```shell
scc disconnect -p <profilename>
```

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p</td>
<td>Displays disconnect progress</td>
</tr>
</tbody>
</table>

**Comments**  Shortcut: `scc d`

You must be in CLI mode to run this command.

**scc erasecreds**

**Description**  This command unsets authorization credentials

**Usage**  
```shell
scc erasecreds
```

**Comments**  Shortcut: `scc ep`

You need to be in CLI mode to run this command.

**scc listprofiles**

**Description**  This command lists all profiles

**Usage**  
```shell
scc listprofiles
```

**Comments**  Shortcut: `scc lp`

You must be in CLI mode to run this command.
**scc numprofiles**

Description: This command displays the number of profiles.

Usage: `scc numprofiles`

Comments: Shortcut: `scc np`

You need to be in CLI mode to run this command.

---

**scc restartsc**

Description: This command restarts SecureClient services.

Usage: `scc restartsc`

Comments: You need administrator privileges to run this command.

---

**scc passcert**

Description: This command sets the user’s authentication credentials when authentication is performed using certificates.

Usage: `scc passcert <certificate> <password>`

Comments: Shortcut: `scc pc`

You need to be in CLI mode to run this command.

---

**scc setmode <mode>**

Description: This command switches the SecuRemote/SecureClient mode

Usage: `scc setmode [-cli | -con]`

Syntax:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-cli</td>
<td>command line interface mode</td>
</tr>
<tr>
<td>-con</td>
<td>connect mode</td>
</tr>
</tbody>
</table>

Comments: You need administrator privileges to run this command.
scc setpolicy

scc setpolicy
Description: This command enables or disables the current default security policy.
Usage: scc setpolicy [on|off]
Comments: Shortcut: scc sp
You need administrator privileges to run this command.

scc sp

scc sp
Description: This command displays the current default security policy.
Usage: scc sp
Comments: You need to be in CLI mode to run this command.

scc startsc

scc startsc
Description: This command starts SecureClient services.
Usage: scc startsc
Comments: You need administrator privileges to run this command.

scc status

scc status
Description: This command displays the connection status.
Usage: scc status
Comments: Shortcut: scc s

scc stopsc

scc stopsc
Description: This command stops SecureClient services.
Usage: scc stopsc
Comments: You need administrator privileges to run this command.
**scc suppressdialogs**

**Description**
This command enables or suppresses dialog popups. By default, `scc suppressdialogs` is off.

**Usage**
`scc suppressdialogs [on|off]

**Comments**
When using `scc suppressdialogs on`, only popups requesting authentication credentials appear.

Shortcut: `scc sd`

You need to be in CLI mode to run this command.

---

**scc userpass**

**Description**
This command sets the user’s authentication credentials -- username, and password.

**Usage**
`scc userpass <username> <password`

**Comments**
Shortcut `scc up`

You need to be in CLI mode to run this command.

---

**scc ver**

**Description**
This command displays the current SecureClient version.

**Usage**
`scc ver`
ClusterXL Commands

cphaconf

Description
The cphaconf command configures ClusterXL.

Warning • Running this command is not recommended. It should be run automatically, only by Firewall-1 or by Check Point support. The only exception to this rule is running this command with set_cpp option, as described below.

Usage
```
cphaconf [-i <machine id>] [-p <policy id>] [-b <db_id>] [-n <cluster num>][c <cluster size>] [-m <service>]
[-t <secured IF 1>...] start

cphaconf [-t <secured IF 1>...] [-d <disconnected IF 1>...] add

cphaconf clear-secured
cphaconf clear-disconnected
cphaconf stop
cphaconf init
cphaconf forward <on/off>
cphaconf debug <on/off>
cphaconf set_ccp <broadcast/multicast>
cphaconf mc_reload
cphaconf debug_data
```

Syntax

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cphaconf set_ccp &lt;broadcast/multicast&gt;</td>
<td>Sets whether Cluster Control Protocol (CCP) packets should be sent with a broadcast or multicast destination MAC address. The default behavior is multicast. The setting created using this command will survive reboot. Note, the same value (either broadcast or multicast) should be set on all cluster members.</td>
</tr>
</tbody>
</table>
**cphaprob**

**Description**
The `cphaprob` command verifies that the cluster and the cluster members are working properly.

**Usage**
- `cphaprob -d <device> -t <timeout(sec)> -s <ok|init|problem> [-p] register`
- `cphaprob -f <file> register`
- `cphaprob -d <device> [-p] unregister`
- `cphaprob -d <device> -s <ok|init|problem> report`
- `cphaprob [-i[a]] [-e] list`
- `cphaprob state`
- `cphaprob [-a] if`

**Syntax**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`cphaprob -d &lt;device&gt; -t &lt;timeout(sec)&gt; -s &lt;ok</td>
<td>init</td>
</tr>
<tr>
<td><code>cphaprob -f &lt;file&gt; register</code></td>
<td>Register all the user defined critical devices listed in <code>&lt;file&gt;</code>.</td>
</tr>
<tr>
<td><code>cphaprob -d &lt;device&gt; [-p] unregister</code></td>
<td>Unregister a user defined <code>&lt;device&gt;</code> as a critical process. This means that this device is no longer considered critical.</td>
</tr>
<tr>
<td><code>cphaprob -a unregister</code></td>
<td>Unregister all the user defined <code>&lt;device&gt;</code>.</td>
</tr>
<tr>
<td>`cphaprob -d &lt;device&gt; -s &lt;ok</td>
<td>init</td>
</tr>
<tr>
<td><code>cphaprob [-i[a]] [-e] list</code></td>
<td>View the list of critical devices on a cluster member, and of all the other machines in the cluster.</td>
</tr>
<tr>
<td><code>cphaprob state</code></td>
<td>View the status of a cluster member, and of all the other members of the cluster.</td>
</tr>
<tr>
<td><code>cphaprob [-a] if</code></td>
<td>View the state of the cluster member interfaces and the virtual cluster interfaces.</td>
</tr>
</tbody>
</table>
**cphastart**

**Description**  
Running `cphastart` on a cluster member activates ClusterXL on the member. It does not initiate full synchronization. `cpstart` is the recommended way to start a cluster member.

**cphastop**

**Description**  
Running `cphastop` on a cluster member stops the cluster member from passing traffic. State synchronization also stops. It is still possible to open connections directly to the cluster member. In High Availability Legacy mode, running `cphastop` may cause the entire cluster to stop functioning.