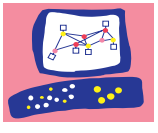
Check Point™ VPN-1/FireWall-1® Reference Guide

Check Point 2000

Part No.: 700058
January 2000

CHECK POINT™
Software Technologies Ltd.



We Secure the Internet.

Check Point Software Technologies Ltd.

International Headquarters:
3A Jabotinsky Street
Ramat Gan 52520, Israel
Tel: 972-3-753 4555
Fax: 972-3-575 9256

e-mail: info@CheckPoint.com

U.S. Headquarters:
Three Lagoon Drive, Suite 400
Redwood City, CA 94065
Tel: 800-429-4391 ; (650) 628-2000
Fax: (650) 654-4233

http://www.checkpoint.com

February 2, 2000

Please direct all comments regarding this publication to **techwriters@checkpoint.com.**

Contents

Figures

Tables

Preface

Scope

The VPN-1/FireWall-1 User Guide describes CheckPoint VPN-1/FireWall-1, and consists of the following books:

Getting Started with VPN-1/FireWall-1

This book introduces VPN-1/FireWall-1 and describes the VPN-1/FireWall-1 installation process.

VPN-1/FireWall-1 Administration Guide

This book is the technical reference to VPN-1/FireWall-1 features, including authentication and address translation. In addition, chapters on troubleshooting and Frequently Asked Questions (FAQ) are included.

VPN-1/FireWall-1 Virtual Private Networks

This book describes how to implement the Virtual Private Network features in Check Point VPN-1/FireWall-1.

VPN-1/FireWall-1 Reference Guide

This book describes INSPECT, the command line interface and other reference subjects, and includes a glossary.

Account Management Client

This book describes how to install and use the Check Point Account Management Client.

Who Should Use this User Guide

This User Guide is written for system administrators who are responsible for maintaining network security. It assumes you have a basic understanding and a working knowledge of:

- system administration
- the Unix or Windows operating system
- the Windows GUI
- Internet protocols (IP, TCP, UDP *etc*.)

What's New in FireWall-1 Version Check Point 2000?

Summary of Contents

Chapter 1, "Command Line Interface," describes the command-line interface.

Chapter 2, "VPN-1/FireWall-1 – Windows Interaction" describes the interaction between VPN-1/FireWall-1 and Windows NT, including the Registry.

Chapter 3, "The INSPECT Language," describes the INSPECT language.

Appendix A, "Glossary" is a glossary of VPN-1/FireWall-1 and Internet communications terminology.

What Typographic Changes Mean

The following table describes the typographic changes used in this book.

TABLE P-1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% You have mail.`
`AaBbCc123`	What you type, when contrasted with on-screen computer output	`machine_name% u` `Password:`
AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type `rm` *filename*.
AaBbCc123	Book titles, new words or terms, or words to be emphasized	Read Chapter 6 in *User's Guide*. These are called *class* options. You *must* be root to do this.
Save	Text that appears on an object in a window	Click on the **Save** button.

Note – This note draws the reader's attention to important information.

Warning – This warning cautions the reader about an important point.

Tip – This is a helpful suggestion.

Shell Prompts in Command Examples

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, Korn shell and DOS.

TABLE P-2 Shell Prompts

Shell	Prompt
C shell prompt	*machine_name*%
C shell superuser prompt	*machine_name*#
Bourne shell and Korn shell prompt	$
Bourne shell and Korn shell superuser prompt	#
DOS	*current-directory*>

Network Topology Examples

Network topology examples usually show a gateway's name as a city name (for example, Paris or London) and the names of hosts behind each gateway as names of popular sites in those cities (for example, Eiffel and BigBen).

CHAPTER 1

Command Line Interface

In This Chapter

Unix-NT Syntax Differences

The command line syntax presented here is the Unix syntax. Differences between the Unix and NT command line syntax are described in TABLE 1-1.

TABLE 1-1 Unix-NT syntax differences

Unix	NT
/ in file names	\ in file names
fwm	fw m (space after fw)
fwstart	fw start (space after fw)

Overview

The fw program is used to manage the VPN-1/FireWall-1. With the exception of the setup commands cpconfig, fwstart and fwstop (see "Setup" on page 4), all commands have the following syntax:

```
fw action [-d] [targets]
```

FW Options

option	meaning
action	This determines the specific command (e.g. fw load, fw stat, fw ctl). The rest of this chapter describes each command's action and options. These commands are grouped by the following categories: Control, Monitor, Certificates, Utilies, VPN-1 Accelerator Card.
-d	If this flag is the first argument to an fw command, then debug information is generated as the command runs.
targets	Some commands can be executed on the specified targets. See below for more information.

Targets

There are three options for specifying the targets on which a given command is to be executed (see TABLE 1-2). If more than one option is used, the command executes on the combination of targets. If none of these options is specified, the Inspection Code is installed on the local host.

TABLE 1-2 Target options

parameter	meaning
`-conf conffile`	The command is executed on targets specified in `confile`. Each line in `conffile` has the syntax of a target in a target list (see "Target Syntax" on page 3).
`-all`	The command is executed on all targets specified in the default system configuration file (`$FWDIR/conf/sys.conf`).
`target`	The command is executed on the specific named `target`. See "Target Syntax" on page 3 for an explanation of the syntax of this argument.

Target Syntax

Targets can be specified using any of the following formats:

```
interface.direction@host
host
interface.direction
```

Where:

parameter	meaning
`interface`	The name of an interface on the target host. If `all` is specified, all configured interfaces on the target host will be loaded. Examples: `le0`; `lo0`; `all`.
`direction`	One of: `in`, `out`, or `all`. `all` specifies both directions. The reference point for the direction is the target host.
`host`	The name of the network object (as returned by the `hostname` command) or its IP address.
`all`	The meaning of all varies according to its placement. It may specify: both directions, all interfaces or both directions on all interfaces.

- The dot (`.`) and the at-sign (`@`) are part of the syntax; spaces around them are not allowed.
- If `host` is not specified, localhost is assumed.

- If only `host` is specified, `all` is assumed (meaning both directions on all interfaces).

Several targets may be specified in various formats. Command-line separators are subject to the rules of the shell (spaces and tabs are the most common separators).

The format of configuration files is identical to the format of targets. In configuration files, the following separators may be used: spaces, tabs, comma, or new line.

Examples

```
le0.in@host1
all@host2
host3
all.out
all.all
```

Setup

cpconfig

`cpconfig` reconfigures an existing VPN-1/FireWall-1 installation.

Syntax

```
cpconfig
```

Windows NT

In Windows NT, the reconfiguration application is a GUI application that displays all the configuration windows from the VPN-1/FireWall-1 installation as tabs in the same window (FIGURE 1-1).

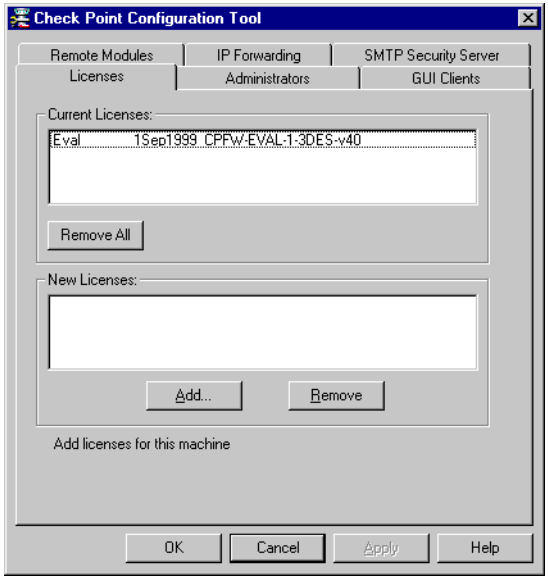


FIGURE 1-1 VPN-1/FireWall-1 Configuration window

To reconfigure an option, click on the appropriate tab and modify the fields as required. Click on **OK** to apply the changes.

The tabs and their fields are described in Chapter 1, “Pre-Installation Configuration” of *VPN-1/FireWall-1 Administration Guide*.

Unix

`cpconfig` displays the following screen. Choose the configuration options you wish to reconfigure.

```
Welcome to VPN-1/FireWall-1 Configuration Program.
==========================================
This program will let you re-configure your VPN-1/FireWall-1
configuration.

Configuration Options:
----------------------
(1)  Licenses
(2)  Administrators
(3)  GUI clients
(4)  Remote Modules
(5)  Security Servers
(6)  SMTP Server
(7)  SNMP Extension
(8)  Groups
(9)  IP Forwarding
(10) Default Filter
(11) Random Pool
(12) CA Keys
(13) Secured Interfaces
(14) High Availability MAC Addresses
(15) High Availability



(16) Exit

Enter your choice (1-16) :
Thank You...
```

Note – The option numbers will vary, depending on the installed configuration.

TABLE 1-3 cpconfig configuration options

option	description	see also ...
Licenses	Update VPN-1/FireWall-1 licenses.	"fw putlic" on page 51
Administrators	Update the list of administrators, users who are authorized to connect to a Management Server through the GUI.	"Access Control" on page 62" of *VPN-1/FireWall-1 Administration Guide*
GUI clients	Update the list of GUI Clients, machines from which administrators are authorized to connect to a Management Server through the GUI.	"Access Control" on page 62" of *VPN-1/FireWall-1 Administration Guide*
Remote Modules	Update the list of remote FireWall and Inspection Modules managed by a Management Module.	"Access Control" on page 62 of *VPN-1/FireWall-1 Administration Guide*
Security Servers	Configure the Security Servers.	"Security Servers" on page 341 of *VPN-1/FireWall-1 Administration Guide*
SMTP Server	Configure the SMTP Security Server.	"SMTP Security Server Configuration" on page 350" of *VPN-1/FireWall-1 Administration Guide*
SNMP Extension	Configure the SNMP Extension.	Chapter 18, "SNMP and Network Management Tools" of *VPN-1/FireWall-1 Administration Guide*
Groups	Update the list of Unix groups authorized to run VPN-1/FireWall-1.	
IP Forwarding	Configure IP Forwarding on the gateway.	"Enabling and Disabling IP Forwarding" on page 23 of *VPN-1/FireWall-1 Administration Guide*
Default Filter	Configure the default Security Policy.	"Default Security Policy" on page 305 of *VPN-1/FireWall-1 Administration Guide*
Random Pool	Configure RSA keys.	Chapter 3, "Certificate Authorities" of *VPN-1/FireWall-1 Virtual Private Networks*
CA keys	Configure Certificate Authority keys.	

TABLE 1-3 cpconfig configuration options(continued)

option	description	see also ...
Secured Interfaces	Configure secured interfaces for High Availability.	"Secured Interfaces" in Chapter 16, "Active Network Management" of *VPN-1/FireWall-1 Administration Guide*
High Availability MAC Addresses	Configure High Availability MAC addresses.	"Sharing IP and MAC Addresses" in Chapter 16, "Active Network Management" of *VPN-1/FireWall-1 Administration Guide*
High Availability	Configure High Availability.	"High Availability" in Chapter 16, "Active Network Management" of *VPN-1/FireWall-1 Administration GuideVPN-1/FireWall-1 Virtual Private Networks*

fwstart

`fwstart` loads the VPN/FireWall Module and starts the following processes:

- VPN-1/FireWall-1 daemon (`fwd`)
- the Management Server (`fwm`)
- VPN-1/FireWall-1 SNMP daemon (`snmpd`)
- the authentication daemons

Syntax

```
fwstart
```

fwstop

`fwstop` kills the following processes:

- VPN-1/FireWall-1 daemon (`fwd`)
- the Management Server (`fwm`)
- VPN-1/FireWall-1 SNMP daemon (`snmpd`)
- the authentication daemons

`fwstop` then unloads the VPN/FireWall Module.

Syntax

```
fwstop
```

Control

fw load

`fw load` compiles and installs a Security Policy to the target's VPN/FireWall Modules. This is done in one of two ways:

1. `fw load` compiles and installs an Inspection Script (`*.pf`) file to the designated VPN/FireWall Modules.
2. `fw load` converts a Rule Base (`*.W`) file created by the GUI into an Inspection Script (`*.pf`) file then installs it to the designated VPN/FireWall Modules.

Note – The scope of a set of rules in a Rule Base and the targets of a Rule Base installation are not the same. The system will install the entire Rule Base on the designated targets. However, only the rules whose scope includes the target system will actually be enforced on a target.

Loading any interface of a target host first completely unloads it. Hence, some interfaces on a target host might be left unloaded if the new Rule Base or compiled VPN/FireWall Module does not contain a rule for them.

To protect a target, you must load a Rule Base 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.

For more information, see "fw gen" on page 25.

Syntax

```
fw load [-all | -conf conffile] [filter-file | rule-base] targets
```

Options

TABLE 1-4 fw load options

parameter	meaning
`-all`	The command is to be executed on all targets specified in the default system configuration file (`$FWDIR/conf/sys.conf`). For more information, see “Targets” on page 3.
`-conf conffile`	The command is to be executed on the targets specified in `conffile`. For more information, see “Targets” on page 3.
`filter-file`	An Inspection Script (`*.pf`).
`rule-base`	A Rule Base file (`*.W`) created by the GUI.
`targets`	The command is to be executed on the designated VPN/FireWall Modules. For more information, see “Targets” on page 3.

Examples

```
fw load my_rules.W
fw load gateway.pf gateway1
fw load -all complex_rules.pf
```

fw bload

`fw bload` compiles and installs a Security Policy to the target’s embedded VPN/FireWall Modules. This is done in one of two ways:

1. `fw bload` compiles and installs an Inspection Script (`*.pf`) file to the FireWall embedded system specified by `targets`.
2. `fw bload` converts a Rule Base (`*.W`) file created by the GUI into an Inspection Script (`*.pf`) file and then cmpiles and installs it to the FireWall embedded system specified by `targets`.

This command is used to install a Security Policy on a system, such as a router a bridge or a switch, that has an embedded VPN/FireWall Module.

For more information, see “fw gen” on page 25.

Syntax

```
fw bload [-all | -conf conffile] [inspect-file | rule-base] targets
```

Options

TABLE 1-5 fw bload options

parameter	meaning
`-all`	The command is to be executed on all targets specified in the default system configuration file (`$FWDIR/conf/sys.conf`). For more information, see "Targets" on page 3.
`-conf conffile`	The command is to be executed on the targets specified in `conffile`. For more information, see "Targets" on page 3.
`rule-base`	A Rule Base file (`*.W`) created by the GUI.
`inspect-file`	An Inspection Script (`*.pf`) file containing the compiled version of the Rule Base.
`targets`	The Security Policy is to be loaded on the designated FireWall imedded system. For more information, see "Target Syntax" on page 3.

fw unload

`fw unload` uninstalls the currently loaded Inspection Code from selected targets.

Syntax

```
fw unload [-all | -conf conffile] targets
```

Options

TABLE 1-6 fw unload options

parameter	meaning
`-all`	The command is to be executed on all targets specified in the default system configuration file (`$FWDIR/conf/sys.conf`). For more information, see "Targets" on page 3.
`-conf conffile`	The command is to be executed on the targets specified in `conffile`. For more information, see "Targets" on page 3.
`targets`	The command is to be executed on these specified VPN/FireWall Modules. For more information, see "Targets" on page 3.

Examples

```
fw unload gateway1
fw unload -a
```

fw fetch

`fw fetch` fetches the Inspection Code from the specified host and installs it to the kernel.

Syntax

```
fw fetch targets
```

Options

TABLE 1-7 fw fetch options

parameter	meaning
`targets`	The Master where the Inspection Code is found. Use `localhost` if there is no Master or if the Master is down. You may specify a list of Masters, which will be searched in the order listed. See also "Target Syntax" on page 3.

Examples

```
fw fetch gateway1
```

fw putkey

`fw putkey` installs a VPN-1/FireWall-1 authentication password on a host. This password is used to authenticate internal communications between VPN/FireWall Modules and between a VPN/FireWall Module and its Management Station. That is, the password is used to authenticate the control channel the first time communication is established. For a detailed example of how `fw putkey` is used, see "Distributed Configurations" on page 69 of *VPN-1/FireWall-1 Administration Guide*.

Syntax

```
fw putkey [-no_opsec] [-opsec] [-ssl] [-p password]
          [-k num] [-n name] target
```

Options

TABLE 1-8 fw putkey options

parameter	meaning
`-no_opsec`	Only VPN-1/FireWall-1 control connections are enabled.
`-opsec`	Only OPSEC control connections are enabled.
`-ssl`	The key is used for an SSL connection.
`-k num`	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.
`-n name`	The IP address (in dot notation) to be used by VPN-1/FireWall-1 when identifying this host to all other hosts, instead of, for example, the resolution of the `hostname` command.
`-p password`	The key (password). If you do not enter the password on the command line, you will be prompted for it.
`target`	The IP address(es) or the resolvable name(s) of the other host(s) on which you are installing the key (password). This should be the IP address of the interface "closest" to the host on which the command is run. If it is not, you will get error messages such as the following: "./fwd: Authentication with *hostname* for command sync failed"

If neither `-opsec` nor `-no_opsec` is specified, then both VPN-1/FireWall-1 and OPSEC connections are enabled.

fw dbload

`fw dbload` downloads the user database and network objects information (for example, encryption keys) to selected targets. If no target is specified, then the database is downloaded to `localhost`.

Syntax

```
fw dbload [-all | -conf conffile] [targets]
```

Options

TABLE 1-9 fw dbload options

parameter	meaning
`-all`	The command is to be executed on all targets specified in the default system configuration file (`$FWDIR/conf/sys.conf`). For more information, see "Targets" on page 3.
`-conf conffile`	The command is to be executed on the targets specified in `conffile`. For more information, see "Targets" on page 3.
`targets`	The command is executed on the designated targets. For more information, see "Target Syntax" on page 3.

fw confmerge

`fw confmerge` merges two `objects.C` files and sends the result to the `stdout`. To obtain the result of the merge in a file you must apply shell redirection to the command.

Syntax

```
fw confmerge file1 file2
```

Options

TABLE 1-10 fw confmerge options

parameter	meaning
`file1`	An `objects.C` file.
`file2`	An `objects.C` file.

Monitor

fw stat

`fw stat` displays the status of target hosts in various formats. The default format displays the following information for each host: host name, Rule Base (or VPN/FireWall Module) file name, date and time loaded, and the interface and direction loaded.

Syntax

```
fw stat [-all | -conf conffile] [-long | -short]
    [-inactive] targets
```

Options

TABLE 1-11 fw stat options

parameter	meaning
`-all`	The command is to be executed on all targets specified in the default system configuration file (`$FWDIR/conf/sys.conf`). For more information, see "Targets" on page 3.
`-conf conffile`	The command is to be executed on the targets specified in `conffile`. For more information, see "Targets" on page 3.

TABLE 1-11 fw stat options (continued)

parameter	meaning
`-long`	In addition to the information displayed in the short format, the long format displays the number of packets in each of the following categories: total, rejected, dropped, accepted, and logged.
`-short`	The short format displays: host name, direction, interface, Rule Base file name and loading date. This is the default format.
`-inactive`	This option displays the status of inactive interfaces too (using the selected format). An inactive interface is an interface that had no packet flow since the last time the Rule Base was loaded on that interface.
`targets`	The command is to be executed on the designated targets. If `targets` is not specified, the status of `localhost` is shown.For more information, see "Targets" on page 3.

Examples

```
fw stat
fw stat -s -a
fw stat -l gateway1
```

fw lichosts

`fw lichosts` prints a list of hosts protected by the VPN-1/FireWall-1/*n* products.

The list of hosts is in the file `$FWDIR/database/fwd.h`.

Syntax

```
fw lichosts [-x] [-l]
```

Options

TABLE 1-12 fw lichosts options

parameter	meaning
`-x`	use hexadecimal format
`-l`	use long format

fw ver

`fw ver` displays the VPN-1/FireWall-1 version number. This is the version of the VPN-1/FireWall-1 daemon, the compiler and the Inspection Script generator (`fw gen`). The version of the GUI is displayed in the opening screen, and can be viewed at any time from the **Help** menu.

Syntax

```
fw ver [ -k ]
```

Options

TABLE 1-13 fw ver options

parameter	meaning
-k	Print the version number of the Kernel Module

fw intelrng

`fw intelrng` displays the status of the Intel RNG. This command is a Windows NT only command.

Syntax

```
fw intelrng
```

Examples

```
#fw checklic
Using Intel(R) Security Driver.
```

```
#fw checklic
Intel(R) Security Driver not detected.
```

fw sam

`fw sam` inhibits (blocks) connections to and from specific IP addresses without the need to change the Security Policy. The command is logged.

To "uninhibit" inhibited connections, execute `fw sam` again with the `-C` or `-D` parameters.

Syntax

```
fw sam [-v] [-s sam_server][-f fwm] [-t timeout] [-C]
      [-n | -i | -I] criterion ip_address
fw sam [-v] [-s sam_server][-f fwm] [-t timeout] [-C]
       [-n | -i | -I] srv source dest dport ip_protocol
fw sam [-v] [-s sam_server][-f fwm] [-t timeout] [-D]
```

Options

TABLE 1-14 fw sam options

<table>
<tr><th>parameter</th><th>meaning</th></tr>
<tr><td>-v</td><td>Verbose mode — writes one message (describing whether the command was successful or not) to stderr for each VPN/FireWall Module 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. See “Configuration Files” on page 19 for more information.</td></tr>
<tr><td>-f fwm</td><td>The VPN/FireWall Modules on which to enforce the action. Can be the name of a FireWalled object, group or one of the following (default is “All”):
<table>
<tr><th>value</th><th>the action will be enforced on ...</th></tr>
<tr><td>All</td><td>all the FireWalls which are defined as gateways or hosts on the machine on which the fw sam command is executed</td></tr>
<tr><td>Gateways</td><td>all the FireWalls which are defined as gateways on the machine on which the fw sam command is executed</td></tr>
</table>
See “Configuration Files” on page 19 for more information.</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>-C</td><td>Cancel the specified command (that is, inhibited connections with the specified parameters will no longer be inhibited). The parameters must match the ones in the original command.</td></tr>
<tr><td>-D</td><td>Cancel all inhibit (-i and -I) and notify (-n) commands.</td></tr>
<tr><td>-n</td><td>Notify, that is, generate a long-format log entry and an alert when connections that match the specified services or IP addresses pass through the FireWall. This action does not inhibit or close connections.</td></tr>
</table>

TABLE 1-14 fw sam options

<table>
<tr><th>parameter</th><th>meaning</th></tr>
<tr><td>-i</td><td>Inhibit the specified connections (that is, do not allow new connections with the specified parameters). Each inhibited connection is logged (long format) and an alert is generated.</td></tr>
<tr><td>-I</td><td>Inhibit the specified connections, and close all existing connections with the specified parameters. Each inhibited connection is logged (long format) and an alert is generated.</td></tr>
<tr><td>criterion ip_address</td><td>Match on the ip_address specified by criterion.
ip_address is an IP address in dot format or a resolvable name.
criterion designates that ip_address is one of the following:
<table>
<tr><th>value</th><th>match on...</th></tr>
<tr><td>src</td><td>the IP address of the source IP address</td></tr>
<tr><td>dst</td><td>the IP address of the destination IP address</td></tr>
<tr><td>any</td><td>the IP address of either the source IP address or the destination IP address</td></tr>
</table></td></tr>
<tr><td>srv source dest dport ip_protocol</td><td>Match on the service that has the following parameters:
<table>
<tr><th>value</th><th>match on...</th></tr>
<tr><td>source</td><td>The source IP address (in dot format or resolvable name).</td></tr>
<tr><td>dest</td><td>The destination IP address (in dot format or resolvable name).</td></tr>
<tr><td>dport</td><td>The destination port (integer or name, for example, “telnet”)</td></tr>
<tr><td>ip_protocol</td><td>The IP protocol (integer or name, for example, “tcp”).</td></tr>
</table></td></tr>
</table>

Configuration Files

There are two configuration files in $FWDIR/conf that affect the functionality of the fw sam command:

product.conf

This file (which you should not modify) has two parameters relevant to fw sam:

- `Management`

  When VPN-1/FireWall-1 is installed, this parameter is set to `1` on Management Stations and to `0` on VPN/FireWall Modules. On machines which are both Management Stations and VPN/FireWall Modules, this parameter is set to `1`.

- `FireWall`

  When VPN-1/FireWall-1 is installed, this parameter is set to `0` on Management Stations and to `1` on VPN/FireWall Modules. On machines which are both Management Stations and VPN/FireWall Modules, this parameter is set to `1`.

On a machine on which `Management` is 0, the `fw sam` command cannot perform remote actions (that is, it cannot inhibit connections through other machines).

On a machine on which `FireWall` is 0, the `fw sam` command cannot perform local actions (that is, it can inhibit connections *only* through other machines).

fwopsec.conf

The `sam_allowed_remote_requests` parameter (default value "`no`") determines whether the `fw sam` command on this machine can perform remote commands. To enable a VPN/FireWall Module to inhibit connections through other FireWalled machines, set `sam_allowed_remote_requests` to "`yes`". Do not try to accomplish this by modifying `product.conf`.

Examples

The command:

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

inhibits all connections originating on `louvre` for 10 minutes.

The command:

```
fw sam -C src louvre
```

is an invalid command because there is no `timeout` parameter. The cancel command's parameters must match the parameters of the command it is meant to cancel.

The command:

```
fw sam -C src louvre -t 60
```

is also an invalid command, because `timeout` is incorrect (it does not match `timeout` in the original command).

The command:

```
fw sam -C any louvre -t 600
```

is also invalid, because `criterion` does not match `criterion` in the original command.

The command:

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

cancels the command in the first example.

Utilities

fw ctl

`fw ctl` sends control information to the VPN-1/FireWall-1 Kernel Module.

Syntax

```
fw ctl [ip_forwarding option] | pstat |  install | uninstall | iflist|
arp
```

Options

TABLE 1-15 fw ctl options

<table>
<tr><th>parameter</th><th>meaning</th></tr>
<tr><td>ip_forwarding option</td><td>option is one of the following:
<table>
<tr><th>value</th><th>match</th></tr>
<tr><td>never</td><td>VPN-1/FireWall-1 does not control (and thus never changes) the status of IP Forwarding.</td></tr>
<tr><td>always</td><td>VPN-1/FireWall-1 controls the status of IP Forwarding as described below.</td></tr>
<tr><td>default</td><td>VPN-1/FireWall-1 controls the status of IP Forwarding only if IP Forwarding is disabled in the kernel. Otherwise, VPN-1/FireWall-1 does not control (and thus does not change) the status of IP Forwarding. This is the default setting.</td></tr>
</table>
For more information, see “IP Forwarding” on page 22.</td></tr>
<tr><td>pstat</td><td>Display VPN-1/FireWall-1 internal statistics.</td></tr>
<tr><td>install</td><td>Install the VPN-1/FireWall-1 kernel.</td></tr>
<tr><td>uninstall</td><td>Uninstall the VPN-1/FireWall-1 kernel.</td></tr>
<tr><td>iflist</td><td>Displays the IP interfaces known to the kernel by name and internal number</td></tr>
<tr><td>arp</td><td>Displays ARP proxy table, which is a mapping of IP and MAC addresses, and utilizes local.arp file.</td></tr>
</table>

IP Forwarding

Consider the following command:

```
fw ctl  ip_forwarding always
```

When VPN-1/FireWall-1 controls the status of IP Forwarding, then VPN-1/FireWall-1 changes the status as follows:

- When VPN-1/FireWall-1 is stopped (fwstop), IP Forwarding is disabled.
- When VPN-1/FireWall-1 is started (fwstart), IP Forwarding is enabled.

This ensures that once VPN-1/FireWall-1 has been started for the first time, there is never a time when the host is forwarding packets while the Security Policy is not fully loaded.

It is recommended that IP Forwarding be disabled in the kernel. See “Enabling and Disabling IP Forwarding” below for instructions on how to do this. In this way, IP Forwarding will be never be enabled unless VPN-1/FireWall-1 is working, no matter which of the above options you have chosen.

In IBM AIX, IP Forwarding is by default disabled during boot, so it is not necessary to disable it in the kernel.

Enabling and Disabling IP Forwarding

This section specifies how to enable and disable IP Forwarding on the following platforms: Solaris 2.x, HP-UX 10, HP-UX 11, Windows NT and IBM AIX.

Solaris 2.x (source routed packets)

To turn off IP Forwarding and source routed packets, edit `/etc/rc2.d/S69inet` and change:

```
ndd -set /dev/ip ip_forwarding 1
```

to:

```
ndd -set /dev/ip ip_forwarding 0
ndd -set /dev/ip ip_forward_src_routed 0
```

For additional information, refer to the man pages for `ndd(1M)` and `ip(7)`.

HP–UX 10

On HP-UX 10, the following commands can be put early in the `rc2.d` directory (whose files are executed one after the other, in alphabetical sequence of their names), provided that `/usr` is mounted locally.

If /usr is mounted locally, put these statements in /sbin/init.d/noipforward:

```
#!/sbin/sh
PATH=/sbin:/usr/sbin:/usr/bin
export PATH
case "$1" in
    start_msg)
    echo "Turn IP-Forwarding OFF"
    ;;

    stop_msg)
    echo "(Not Turning IP-Forwarding on)"
    ;;

    'start')
    if [ -x /usr/bin/adb ]; then
      echo "ipforwarding/W 0" | adb -w /stand/vmunix
        /dev/kmem
    fi
    ;;
    esac
exit 0
```

Make sure /sbin/init.d/noipforward is executable and link it to /sbin/rc2.d/S001noipforward.

If /usr is not mounted locally, then put the above statements in a file that is executed after /usr is mounted.

To enable IP Forwarding, enter the following command:

```
echo "ipforwarding/W 1" | adb -w /stand/vmunix /dev/mem
```

HP–UX 11

To turn off IP Forwarding and source routed packets, edit /etc/rc2.d/S69inet and change:

```
ndd -set /dev/ip ip_forwarding 1
```

to:

```
ndd -set /dev/ip ip_forwarding 0
```

Windows NT

1. When you install VPN-1/FireWall-1, check **Control IP Forwarding** in the **IP Forwarding** window (see FIGURE 2-22 on page 40 of *VPN-1/FireWall-1 Administration Guide*).

If you have already installed VPN-1/FireWall-1, reconfigure VPN-1/FireWall-1 using the VPN-1/FireWall-1 Configuration application. When you do so, the different configuration options will be displayed as different tabs in the Configuration window.

2. Enable the **IP Enable Routing** option in the **TCP/IP Properties** window. To display this window:

   a. Open the **Network** applet in the Windows Control Panel.

   b. In the **Protocols** tab, select **TCP/IP** and click on **Properties**. The **TCP/IP Properties** window is displayed.

3. Reboot the computer.

IBM AIX

Warning – The AIX default is for IP Forwarding to be off. If you enable IP Forwarding while VPN-1/FireWall-1 is not running, you will be exposing your network. Make sure that it is not turned on in one of the `.rc` scripts during boot. Turn it on (with the `no -o ipforwarding=1` command) in the `fwstart` script after VPN-1/FireWall-1 starts enforcing a Security Policy, and turn it off (with the `no -o ipforwarding=0` command) in the `fwstop` script just before VPN-1/FireWall-1 stops.

To enable IP Forwarding, enter the following command:

```
no -o ipforwarding=1
```

To disable IP Forwarding, enter the following command:

```
no -o ipforwarding=0
```

fw gen

`fw gen` generates an Inspection Script (`*.pf`) file from a Rule Base (`*.W`) file. Rule Base files are created by the GUI, but you may edit them and use this command to generate Inspection Scripts (though this is *not* recommended).

Syntax

```
fw gen filename
```

Options

TABLE 1-16 fw gen options

parameter	meaning
`filename`	The Rule Base file.

Examples

```
fw gen $FWDIR/conf/default.W
fw gen $FWDIR/conf/corporate.W | more
fw gen $FWDIR/conf/corporate.W > /tmp/corporate.pf
```

fw kill

`fw kill` sends a signal to a VPN-1/FireWall-1 daemon.

Syntax

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

Options

TABLE 1-17 fw kill options

parameter	meaning
`[-t sig_no] proc-name`	If the file `$FWDIR/log/proc-name.pid` exists, send signal `sig_no` to the pid given in the file. If no signal is specified, signal `15` (`SIGTERM`) is sent.

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

Examples

```
fw kill snmpd
```

sends signal 15 to the VPN-1/FireWall-1 snmp daemon.

```
fw kill -t 1 snmpd
```

sends signal 1 to the VPN-1/FireWall-1 snmp daemon.

fwc

`fwc` is the VPN-1/FireWall-1 INSPECT language compiler. It compiles an Inspection Script (`*.pf`) file but does not install it. You may use this command to see if your Inspection Scripts can be compiled, without actually installing them on VPN/FireWall Modules.

`fwc` creates several files in `$FWDIR/tmp`.

- an Inspection Code (*.fc) file
- a VPN/FireWall Module tables (*.ft) file
- log format (*.lg) file
- *.set,*.db and *.objects files

Syntax

```
fwc filename
```

Options

TABLE 1-18 fw kill options

parameter	meaning
filename	An Inspection Script (*.pf).

fwm

fwm is the VPN-1/FireWall-1 Management Server in the Client/Server implementation of the Management Station, and is used for communicating with the GUI and adding, updating and removing administrators.

fwm must be running on the Management Server if you wish to use the GUI Client on one of the client machines.

For more information about the VPN-1/FireWall-1 Management Server, see Chapter 1, "Pre-Installation Configuration" of *VPN-1/FireWall-1 Administration Guide*.

Syntax

```
fwm [-a name [-w {w|u|r|m}] [-s password] [-q]| -r name | -p | -g]
```

Options

TABLE 1-19 fwm options

parameter	meaning
-a name	Add or update the administrator with username name.
-w	Set access level as follows: w — Read/Write u — User Edit r — Read Only m — Monitor Only
-s password	Set the administrator's password
-q	When adding an administrator, don't prompt for the administrator's password (useful for batch updates).

TABLE 1-19 fwm options

parameter	meaning
`-r name`	Delete an administrator.
`-p`	Print a list of administrators.
`-g`	Convert the old `*.W` files to one unified `rulebases.fws` that is used by `fwm`

Examples

To add an administrator, type:

```
fwm -a
```

You will be prompted to type the user's name and password, and then to confirm the password by typing it a second time.

To delete an administrator, type:

```
fwm -r
```

You will prompted to type the user's name.

fwell

`fwell` manages Access Lists for Wellfleet (Bay Networks) routers.

Syntax

```
fwell load rulebase-file [-s] [-u] [interface-name@]router-name
       [targets]
fwell unload  [-s] [-u] [interface-name@]router-name targets
fwell stat    targets
```

Options

TABLE 1-20 fwell options

parameter	meaning
`load rulebase-file`	Load the Access List specified by the Rule Base file (`*.W`) to the router.
`interface-name`	
`router-name`	
`targets`	The command is to be executed on these machines. For more information, see "Target Syntax" on page 3.

TABLE 1-20 fwell options (continued)

parameter	meaning
unload	Unload the Access List.
-s	Generate summary output.
-u	A list of interfaces.
stat	Show statistics.

Note – When loading a Rule Base to a router, all the router's interfaces are first unloaded. If the `-u` parameter is specified, then the virtual router's interfaces are unloaded. If the `-u` parameter is not specified, then the real router's interfaces are unloaded.

Examples

The command:

```
fwell stat well
```

produces output similar to the following:

```
CIRCUIT   IF              FILTERDATE
E21       -               -
S21       192.114.50.33   d423Mar95 10:34:13
S22       -               -                   -
```

Individual Interface Loading for Bay Routers (Wellfleet)

Rather than loading (or unloading) the Security Policy (Access Lists) to (or from) all the interfaces of a Bay Router, it is possible to specify individual interfaces.

Example 1

Suppose a Wellfleet router well has three interfaces: `E21`, `S21` and `S22`.

The user might wish to define (manually, in `objects.C`) two "virtual" routers, `well1` and `well2`, as follows:

```
(well1
    :ipaddr well
    :if-1E21
)
(well2
    :ipaddr well
    :if-0S21
    :if-2S22
)
```

The list of interfaces to be loaded or unloaded is specified in the command line

Example 2

The command:

```
fwell load p.W E21@well1
```

performs the following actions:

- unloads `E21`, `S21`, `S22` (all the interfaces of the real router `well1` — this is because the `-u` parameter was not specified)
- loads `E21` (all the interfaces of the virtual router `well1`)

In practice, specifying `E21` in the command line had no effect. All the interfaces were loaded, but as it happens, `well1` has only one interface.

Example 3

The command:

```
fwell load -u p.W well2
```

performs the following actions:

- unloads `S21` and `S22` (all `well2` interfaces — this is because the `-u` parameter was specified)
- load `S21` and `S22` (all `well2` interfaces)

Example 4

The command:

```
fwell load -u p.W S21@well2
```

performs the following actions:

- unload `S21` (the only interface specified in the command line)
- load `S21` (the only interface specified in the command line)

fw tab

`fw tab` displays the content of INSPECT 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.

Syntax

```
fw tab [-all | -conffile] [-a] [-s][-u | -m number] [-t tname] targets
```

Options

TABLE 1-21 fw tab options

parameter	meaning
`-all`	The command is to be executed on all targets specified in the default system configuration file (`$FWDIR/conf/sys.conf`). For more information, see "Targets" on page 3.
`-conf conffile`	The command is to be executed on the targets specified in `conffile`. For more information, see "Targets" on page 3.
`-a`	Display all tables.
`-s`	Use short format: host name, table name, table ID, and its number of elements.
`-u`	Do not limit the number of displayed entries.
`-m number`	For each table, display only its first `number` of elements (default is 16).
`-t tname`	Display only `tname` table.
`targets`	The command is executed on the designated targets. For more information, see "Target Syntax" on page 3.

Examples

```
fw tab
fw tab -t hostlist1 gateway1
```

snmp_trap

`snmp_trap` sends an SNMP trap to the specified host. The message may appear in the command line, or as one line in the program input (`stdin`).

`snmp_trap` is the default command in **SNMP Trap Alert Command** in the **Logging and Alerting** tab of the **Properties Setup** window. You can use the `-v` flag to send the value of one of the VPN-1/FireWall-1 MIB variables (see "VPN-1/FireWall-1 MIB Source" on page 600 of *VPN-1/FireWall-1 Administration Guide*).

Syntax

```
snmp_trap [-v var] [-g generic_trap] [-s specific_trap] host
[message]
```

Options

TABLE 1-22 snmp_trap options

parameter	meaning
`-v var`	An optional object ID to bind with the message.

`-g generic_trap`	One of the following values:
	`0 – coldStart`
	`1 – warmStart`
	`2 – linkDown`
	`3 – linkUp`
	`4 – authenticationFailure`
	`5 – egpNeighborLoss`
	`6 – enterpriseSpecific`

`-s specific_trap`	A unique number specifying the trap type; valid only if `generic_trap` value is `enterpriseSpecific` (default value is 0).
`host`	The name of the host that should receive the trap.
`message`	The message sent to the host.

status_alert

`status_alert` generates an alert. `status_alert` is meant for use in the **Command** field of the **Action on Transition** field in the **Options** window of the System Status Viewer (see "Options" on page 384 of *VPN-1/FireWall-1 Administration Guide*).

Syntax

```
status_alert
```

fw converthosts

`fw converthosts` converts a file in the `/etc/hosts` format to a file in the `dnsinfo.C` format. For information on why this might be needed as well as a description of the `dnsinfo.C` file format, see "DNS" on page 153 of *VPN-1/FireWall-1 Virtual Private Networks*.

Syntax

```
fw converthosts < input_file > output_file
```

Options

TABLE 1-23 fw converthosts options

parameter	meaning
`input_file`	The file in `/etc/hosts` format.
`output_file`	The file in `dnsinfo.C` format.

Examples

```
fw converthosts  /etc/hosts  /tmp/dnsinfo.C
```

Log File Management

fw log

`fw log` displays the content of Log Files.

Syntax

```
fw log [-f] [-c action] [-l] [-s starttime] [-e endtime]
    [-b stime etime]][-h hostname] [logfile] [-n]
```

Options

TABLE 1-24 fw log options

parameter	meaning
`-f`	After current display is completed, do not exit but continue to monitor the Log file and display it while it is being written.
`-c action`	Display only events whose action is `action`, that is, `accept`, `drop`, `reject`, `authorize`, `deauthorize`, `encrypt` and `decrypt`. Control actions are always displayed.
`-l`	Display the date for each record.
`-s starttime`	Display only events that were logged after time. `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 time. `endtime` may be a date, a time, or both.
`-b stime etime`	Display only events that were logged between `stime` and `etime`, each of which may be a date, a time, or both. If date is omitted, then today's date is assumed.
`-h hostname`	Display only log entries sent by the FireWalled machine `hostname`.
`logfile`	Use `logfile` instead of the default Log file. The default Log File is `$FWDIR/log/fw.log`.
`-n`	Don't perform DNS resolution of the IP addresses in the Log File (this option significantly speeds up the processing)

The syntax for `starttime`, `endtime`, `stime`, `etime` is as follows.

- `DD-mon-YYYY` (e.g. `12-jan-2001`)
- `mon DD, YYYY` (e.g. `Feb 14, 1999`)
- `MM/DD/YYYY`
- `DD.MM.YYYY`
- `YYYY-MM-DD`
- `DDMMYYYY`

`DD` is the day (e.g. `01`, `14`, `28`).

`YYYY` is the year (e.g. `1994`).

`MM` is the month (e.g. `01`, `04`, `12`).

`mon` is the month (e.g. `jan`, `feb`, `mar`)

Examples

```
fw log
fw log | more
fw log -c reject
fw log -s Jan1
fw log -f -s 16:00
```

fw logswitch

`fw logswitch` creates a new Log File. The current Log File is closed and renamed `$FWDIR/log/`*`date`*`.log`, and a new Log File with the default name (`$FWDIR/log/fw.log`) is created. Old Log Files are located in the same directory. You must have the appropriate file privileges to run `fw logswitch`.

A Management Station can use `fw logswitch` to switch a Log File on a remote machine and transfer the Log File to the Management Station. For information on how to direct logging to a specific machine, see "Redirecting Logging to Another Master" on page 420 of *VPN-1/FireWall-1 Administration Guide*.

See also "How can I switch my Log File on a periodic basis?" on page 615 of *VPN-1/FireWall-1 Administration Guide*.

Syntax

```
fw logswitch [-h target] [+|-][“”|old_log]
```

Options

TABLE 1-25 fw logswitch options

parameter	meaning
`-h target`	The resolvable name or IP address of the remote machine (running either a VPN/FireWall Module or a Management Station) on which the Log File is located. The Management Station (on which the `fw logswitch` command is executed) must be defined as one of `target`'s Management Stations. In addition, you must perform `fw putkey` to establish a control channel between the Management Station and `target`. For information about establishing control channels, see "Distributed Configurations" on page 69 of *VPN-1/FireWall-1 Administration Guide*. For information on target syntax, see "Target Syntax" on page 3.
`+`	The Log File is transferred from `target` to the Management Station. The transferred Log File is compressed and encrypted. The name of the copied Log File on the Management Station is prefixed by `target` (see "Targets" on page 3 for details). This parameter is ignored if `target` is not specified. There should be no white space between this parameter and the next one.
`-`	The same as +, but the Log File is deleted on `target`.
`""`	Delete the current Log File (on `target` if specified; otherwise on the Management Station).
`old_log`	The new name of the old Log File.

TABLE 1-26 lists the files created in the `$FWDIR/log` directory on both `target` and the Management Station when the + or - parameters are specified. Note that if - is specified, the Log File on target is deleted rather than renamed.

TABLE 1-26 Files created in $FWDIR/log

	old_log specified	old_log not specified
target specified	On `target`, the old Log File is renamed to `old_log`. On the Management Station, the copied file will have the same name, prefixed by `target`'s name. For example, the command `fw logswitch -h venus +xyz` creates a file named `venus.xyz` on the Management Station.	On `target`, the new name is the current date, for example, `04Feb98-10:04:20` in Unix and `04Feb98-100420` in NT. On the Management Station, the copied file will have the same name, but prefixed by `target`. For example, (`target.04Feb98-10:04:20` in Unix and `target.04Feb98-100420` in NT.)
target not specified	On the Management Station, the old Log File is renamed to `old_log`.	On the Management Station, the old Log File is renamed to the current date(see above).

If either the Management Station or `target` is an NT machine, the files will be created using the NT naming convention (see TABLE 1-1 on page 2).

Examples

The following command creates a new Log File and moves (renames) the old Log File to `old.log`.

```
fw logswitch old.log
```

fw logexport

`fw logexport` exports the Log File to an ASCII file.

Syntax

```
fw logexport [-d delimiter] [-i inputfile] [-o outputfile]
    [-r record_chunk_size] [-n] [-f]
```

Options

TABLE 1-27 fw logexport options

parameter	meaning
`-d delimiter`	Output fields will be separated by this character — default is semicolon (`;`)
`-i inputfile`	The name of the input Log File.
`-o outputfile`	The name of the output ASCII file.
`-r record_chunk_size`	This determines how many records should be read (during a single access to the Log File) into the internal buffer for processing.
`-n`	Do not perform DNS resolution of the IP addresses in the Log File (this option significantly speeds the processing).
`-f`	Stay online and export new logs to the ASCII output file as they occur.

User Database Management

fw dbimport

`fw dbimport` imports users into the VPN-1/FireWall-1 User Database from an external file. You can create this file yourself (see “File Format” on page 39), or use a file generated by `fw dbexport` (see “fw dbexport” on page 41).

See also “ldapmodify” on page 43.

Syntax

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

Options

TABLE 1-28 fw dbimport options

parameter	meaning
`-m`	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. If `-m` is not specified, then an existing user's original values will be not be modified.
`-s`	Suppress the warning messages issued when an existing user's values are changed by values in the import file.
`-v`	verbose mode
`-r`	`fw dbimport` will delete all existing users in the database.
`-k nerrors`	Continue processing until `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.
`-f file`	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 39.
`-d`	Specifies a delimiter different from the default value (`;`).

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 syntax:

1 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 `name` is included:

```
{name; groups; destinations; sources; auth_method; fromhour; tohour;
expiration_date; color; days; internal_password; SKEY_seed;
SKEY_passwd; SKEY_gateway; template; comments; userc}
```

2 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 (see below).

3. 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.

4. 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.

5. Legal values for the days attribute are: `MON`, `TUE`, `WED`, `THU`, `FRI`, `SAT`, `SUN`.

6. Legal values for the authentication method are: `Undefined`, `S/Key`, `SecurID`, `Unix Password`, `VPN-1/FireWall-1 Password`, `RADIUS`, `Defender`.

7. Time format is `hh:mm`.

8. Date format is `dd-mmm-yy`, where `mmm` is one of {`Jan`, `Feb`, `Mar`, `Apr`, `May`, `Jun`, `Jul`, `Aug`, `Sep`, `Oct`, `Nov`, `Dec`}.

9. If the S/Key authentication method is used, all the other attributes regarding this method must be provided.

10. 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.

11. Values regarding authentication methods other than the one specified are ignored.

12. The `userc` field specifies the parameters of the user's SecuRemote connections, and has three parameters, as follows:

TABLE 1-29 SecuRemote parameters

parameter	values
key encryption method	FWZ1, DES, CLEAR, Any
data encryption method	FWZ1, 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:

userc	means
{FWZ1,FWZ1,MD5}	key encryption method is FWZ1; data encryption method is FWZ1; data integrity method is MD5
{DES,CLEAR,}	key encryption method is FWZ1; no data encryption; no data integrity
{Any,Any,}	use “best” key encryption method; use “best” data encryption method; no data integrity

13 A line beginning with the ! character is considered a comment.

fw dbexport

`fw dbexport` exports the VPN-1/FireWall-1 User Database to a file. The file may be in one of the following formats:

- the same syntax as the import file for `fw dbimport` (see “fw dbimport” on page 38)
- LDIF syntax, which can be imported into an LDAP Server using `ldapmodify` (see “ldapmodify” on page 43),

Syntax

- To export the User Database to a file that can be used with `fw dbimport`:

```
fw dbexport [ [-g group | -u user] [-d delim]
    [-a {attrib1, attrib2, ...} ] [-f file] ]
```

- To export the User Database as an LDIF file:

```
fw dbexport -l [-d <delim>] [-a {attrib1, attrib2, ...} ]  -s <subtree>
[-f <file>]  [-k <IKE shared secret>]
```

Options

TABLE 1-30 fw dbexport options

parameter	meaning
`-g group`	Specifies a group of users (`group`) to be exported; users are not exported.
`-u user`	Specifies that only one user (`user`) be exported.
`-d delim`	Specifies a delimiter different from the default value (“;”).

TABLE 1-30 fw dbexport options

parameter	meaning
`-a {attrib1, attrib2, ...}`	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.
`-f file`	`file` specifies the name of the output file. The default output file is `$FWDIR/conf/user_def_file`.
`-l`	Create an LDIF format file for importation by an LDAP server.
`-s`	The branch under which the users are to be added.
`-k`	This is theAccount Unit's IKE shared secret (**IKE Key** in the **Encryption** tab of the **Account Unit Properties** window — see "LDAP Account Unit Properties Window — Encryption Tab" on page 339 of *VPN-1/FireWall-1 Administration Guide*).

Warning – If you use the `-a` parameter to specify a list of attributes, and then import the created file using `fw dbimport`, the attributes not exported will be deleted from the user database.

Notes

- `fw dbexport` and `fw dbimport` (non-LDIF syntax) cannot export and import user groups. To export and import a user database, including groups, proceed as follows:

  1. Run `fw dbexport` on the source Management Station.
  2. On the destination Management Station, create the groups manually.
  3. Run `fw dbimport` on the destination Management Station.

  The users will be added to the groups to which they belonged on the source Management Station.

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

```
fw dbexport -f f1 -l -s ou=marketing,o=WidgetCorp,c=us
fw 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 VPN-1/FireWall-1 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.

Examples

```
fw dbexport -l -s cn=maryj,o=WidgetCorp,c=us
```

creates a LDIF file consisting of one entry, where the DN is:

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

ldapmodify

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

You can import VPN-1/FireWall-1 User Database to an LDAP server by first generating an LDIF file using `fw dbexport` ("fw dbexport" on page 41), and then using `ldapmodify`.

Before importing, prepare the LDAP directory as follows:

1. Make sure the root branch is defined as an allowed branch on your LDAP server.
2. Restart the LDAP server.
3. Create the branch into which the users will be imported, either by using **Create Tree Object** in the Account Management Client or with the `ldapmodify` command:

```
ldapmodify -a -h host -p port -D LDAPadminDN -w LDAPadminPassword
dn: o=myOrg,c=US
objectclass: organization
o:myOrg
```

Syntax

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

Options

TABLE 1-31 ldapmodify options

parameter	meaning
`-a`	Add users.
`-c`	Continue on errors.
`-h host`	LDAP Server IP address.
`-p port`	LDAP Server port number.
`-D LDAPadminDN`	LDAP Administrator DN.
`-w LDAPadminPassword`	LDAP Administrator password.
`-f exportfilename.ldif`	Specifies the name of the input file. This file must be in the LDIF format.

Example

1. Export the users using `fw dbexport`.

```
fw dbexport -l -f ./o_file.ldif -s "o=bigcorp,c=uk" -k hello1234
```

For more information, see “fw dbexport” on page 41.

2. Create the `"o=bigcorp,c=uk"` branch (see step 3 above).

3. Import the users:

```
ldapmodify -a -c -h host -p port -D bindDN -w bindPas -f ./o_file.ldif
```

4. Define an Account Unit with these parameters, including hello1234 as the IKE shared secret.

fw ldapsearch

`fw ldapsearch` queries an LDAP directory and returns the results.

Syntax

```
fw ldapsearch [options] filter [attributes]
```

Options

TABLE 1-32 fw ldapsearch options

parameter	meaning	
options	Any of the following:	
	option	**meaning**
	-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 will be named /tmp/ldapsearch-fw1color-a00188.
	-T timeout	The 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	The server side size limit for search, in entries.
filter	RFC-1558 compliant LDAP search filter. For example, objectclass=fw1host.	
attributes	The list of attributes to be retrieved. If no attributes are given, all attributes are retrieved.	

Examples

```
fw 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.

fw expdate

`fw expdate` changes the expiration date of users (but not templates) in the VPN-1/FireWall-1 User Database to the date specified by the first parameter. This change can be optionally be applied only to selected users by specifying the second parameter.

Syntax

```
fw expdate dd-mmm-yyyy [-f dd-mmm-yyyy]
```

Options

TABLE 1-33 fw expdate options

parameter	meaning
`dd-mmm-yyyy`	The new expiration date, in the standard Check Point date format. This date must be in the future. `mmm` is one of {`Jan`, `Feb`, `Mar`, `Apr`, `May`, `Jun`, `Jul`, `Aug`, `Sep`, `Oct`, `Nov`, `Dec`}.
`[-f dd-mmm-yyyy]`	Only users whose expiration date is the one specified are affected.

Examples

The command:

```
fw expdate 31-Dec-2000 -f 31-Dec-1999
```

changes the expiration date only for those users who have the expiration date of December 31, 1999.

Certificates

fw ca putkey

`fw ca putkey` distributes the Certificate Authority Key to a FireWalled gateway. For more information see “Generating CA Keys” on page 510 of *VPN-1/FireWall-1 Administration Guide*.

Syntax

```
fw ca putkey [-p password] target
```

Options

TABLE 1-34 fw ca putkey options

parameter	meaning
`-p password`	A password that will be used to authenticate future communication between the Management Station and the gateway. The password can be entered on the command line (using the `-p` argument). If you do not enter a password on the command line, you will be prompted for one.
`target`	The IP address or resolvable name of the machine on which you are installing the CA key (the FireWalled gateway).

fw ca genkey

`fw ca genkey` is used to generate the Certifcate Authority Key on a Management Station.

For more information see “Generating CA Keys” on page 510 of *VPN-1/FireWall-1 Administration Guide*.

Syntax

```
fw ca genkey DN
```

Options

TABLE 1-35 fw ca genkey options

parameter	meaning
DN	The distinguished name of the Certificate Authority.

Examples

```
fw ca genkey ou=research,o=widgetcorp,c=us
```

fw certify ssl

`fw certify ssl` is used to generate a Certificate Authority certificate on a FireWalled gateway.

For more information see "Generating CA Keys" on page 510 of *VPN-1/FireWall-1 Administration Guide*.

Syntax

```
fw certify ssl management target
```

Options

TABLE 1-36 fw certify ssl options

parameter	meaning
management	The IP address or resolvable name of the FireWalled gateway's Management Station.
target	The IP address or resolvable name of the machine on which you are installing the CA key (the FireWalled gateway).

You will be prompted for a password. You must enter the same password you used when you issued the `fw ca putkey` command on the Management Station.

Internal CA

This utility enables hybrid authentication mode, which allows the server to perform IKE key exchange with the clients using authentication schemes non-interoperable with IKE. Typically, a server authenticates to a client using a certificate. However, in a

network with an extensive infrastructure of Secure ID cards, certificates may prove unfeasible because the only network objects requiring such a certificate are VPN/FireWall Modules, while the clients use another authentication method. To resolve this problem, the system administrator can create an Internal CA on the Management Station which issues certificates to VPN/FireWall Modules.

Note – Internal CA must be created prior to implementing IKE encryption.

fw internalca

`fw internalca` instructs the Management Station to initiate an Internal CA, which involves creating an Internal CA database, generating public and private keys, issuing a certificate and saving it.

Syntax

```
fw internalca create [-dn] | certify [-o] [-c] [-dn]
```

Options

TABLE 1-37 fw internalca options

parameter	meaning
`create`	This instructs the Management Station to initiate an Internal CA.
`-dn`	The Distinguished Name included in the issued certificate
`certify`	This enables you to specify the network objects which will obtain the issued certificate.
`-o`	A network object name.
`-c`	A nickname by which VPN-1/FireWall-1 recognizes the network object.

fw ikecrypt

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

Note – An internal CA must be created before implementing IKE encryption. For more information, see “fw internalca” on page 49.

Syntax

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

Options

TABLE 1-38 fw ikecrypt options

parameter	meaning
`shared-secret`	The IKE Key defined in the **Encryption** tab of the **LDAP Account Unit Properties** window.
`user-password`	The SecuRemote user's password.

Examples

The command

```
fw ikecrypt MySecret UsersPassword
```

returns the following string (in `stdout`):

```
KYTSLfvuOkzX14edJHIXcwqZsDWv
```

License Management

fw checklic

`fw checklic` prints or displays requested details of the VPN-1/FireWall-1 license.

Syntax

```
fw checklic [-product product-name][-version product-version]
[-kernel] [-quiet] [-count] [-time date] [-routers]
[-embedded] [-SRusers] feature
```

Options

TABLE 1-39 fw checklic options

parameter	meaning
`-product product-name`	The product name for which license information is requested.
`-version product-version`	The product version for which license information is requested.
`-kernel`	Check the license of the Kernel Module.
`-quiet`	Do not print any output.
`-count`	Count how many licenses exist for this feature.
`-time <date>`	Check license status on future date.
`-routers`	Check how many routers are allowed.
`-embedded`	Check how many embedded modules are allowed.
`-SRusers`	Check how many SecuRemote users are allowed.
`feature`	The feature for which license information is requested.

Examples

```
fw checklic fm
fw checklic -product fw1 -version 4.1 encryption
```

fw putlic

`fw putlic` installs a VPN-1/FireWall-1 license on a host.

You can also install licenses with the `cpconfig` command (see “cpconfig” on page 4).

After installing a license, it’s best to do the following:

1. Stop the VPN/FireWall Module (`fwstop`).
2. Start the VPN/FireWall Module (`fwstart`).
3. Determine the current licenses with the `fw printlic -k` command (see “fw printlic” on page 54).

Syntax

```
fw putlic [-overwrite]
    [-check-only] [-Check-one]  [-f licensefile]
    [-kernelonly] [-l inputfile] [-F outputfile]
    hostname|ip-addr|hostid|eval
    licensekey features certificatekey
```

Options

TABLE 1-40 fw putlic options

parameter	meaning
`-overwrite`	Overwrite (delete) all existing licenses with the new license.
`-check-only`	Verify the license.
`-Check-one`	Verify the license without printing its header.
`-f licensefile`	The name of the file with the license text.
`-kernelonly`	Copy the user level license to the kernel — takes no parameters.
`-l inputfile`	Add the license to the license database.
`-F outputfile`	Print the license to the designated file.

TABLE 1-40 fw putlic options

<table>
<tr><th>parameter</th><th>meaning</th></tr>
<tr><td>hostname</td><td>The host's name (the name returned by the hostname command).</td></tr>
<tr><td>ip-addr</td><td>The host's IP address</td></tr>
<tr><td>hostid</td><td>One of the following:<table><tr><th>platform</th><th>value</th></tr><tr><td>Sun OS4 and Solaris2</td><td>The response to the hostid command (beginning with 0x).</td></tr><tr><td>HP-UX</td><td>The response to the uname -i command (beginning with 0d).</td></tr><tr><td>NT</td><td>The IP address of the external interface (in dot notation); last part cannot be 0 or 255.</td></tr><tr><td>AIX</td><td>The response to the uname -l command (beginning with 0d), or the response to the uname -m command.</td></tr></table></td></tr>
<tr><td>eval</td><td>This is an evaluation license.</td></tr>
<tr><td>licensekey</td><td>The License Key string you received from the License Distribution Center, for example:
aa6uwknDc-CE6CRtjhv-zipoVWSnm-z98N7Ck3m</td></tr>
<tr><td>features</td><td>A string listing the features included in the license, for example:
CPSUITE-EVAL-3DES-v41</td></tr>
<tr><td>certificatekey</td><td>The Certificate Key string, for example:
CK0123456789ab</td></tr>
</table>

Example

This command:

```
fw putlic eval 2f540abb-d3bcb001-7e54513e-kfyigpwn CPSUITE-EVAL-3DES-v41
CK0123456789ab
```

produces output similar to the following:

```
        Host             Expiration Features
Eval    199.213.71.172  21Jul1999 CPSUITE-EVAL-3DES-v41 CK0123456789ab
License file updated
```

In this example:

- The license is an evaluation license.
- The license expires on July 21, 1999.

- The features are “CPSUITE-EVAL-3DES-v41”.
- The Certificate Key is “CKO123456789ab”.

fw printlic

`fw printlic` prints details of the VPN-1/FireWall-1 license.

Syntax

```
fw printlic [-k]
```

Options

TABLE 1-41 fw printlic options

parameter	meaning
-k	Print the license in the Kernel Module.

Example

```
       Host            Expiration  Features
Eval   199.213.71.172  21Jul1999   CPSUITE-EVAL-3DES-v41 CK0123456789ab
License file updated
```

In this example:

- The license is an evaluation license.
- The license expires on July 21, 1999.
- The features are “CPSUITE-EVAL-3DES-v41”.
- The Certificate Key is “CKO123456789ab”.

A valid license may still be irrelevant, because the date may be expired, or the hostid may be incorrect.

If several relevant licenses are installed, their features are ORed together.

Remote Licensing Management

The Remote Licensing Management feature enables the system administrator to:

- install and reinstall licenses on remote VPN/FireWall Modules from the Management Station
- store the installed licenses in the Management Station’s internal database
- print the stored licenses

- import licenses from a VPN/FireWall Module to the internal database for synchronization purposes

Note – The following commands can be executed only on a Management Station.

fw rputlic

`rputlic` remotely installs a VPN-1/FireWall-1 license on a VPN/FireWall Module from the Management Station. The Management Station stores the license in its internal database. Licenses can be reinstalled using the `-r` option.

To store a license in the Management Station's internal database without installing it on a VPN/FireWall Module, use the `-s` option.

This command can be executed only on a Management Station.

Syntax

```
fw rputlic [-s] obj-name [putlic flags] license-info |
[-r] obj-name [putlic flags]
```

Options

TABLE 1-42 fw rputlic options

parameter	meaning
`-s`	Store the license in the internal database without installing it on a network object.
`obj-name`	The name of a network object defined in the Policy Editor.
`putlic flags`	The options supported by the `fw putlic` command. For more information on `fw rputlic`, see "fw rputlic" on page 55.
`license-info`	This is the string you received from the License Distribution Center. It is not needed if `-r` is used.
`-r`	Reinstall the license on the specified network object or all the available network objects (if `obj-name` is not specified).

fw rprintlic

`fw rprintlic` prints the details of the VPN-1/FireWall-1 licenses residing in the Management Station's internal database. If a network object is specified in the command, the licenses installed on this network object are printed. The Management Station does not establish communication with the VPN-1/FireWall-1 Modules while executing this command unless the `-f` option is selected.

This command can be executed only on a Management Station.

Syntax

```
fw rprintlic [-r obj-name] [-f]
```

Options

TABLE 1-43 fw rprintlic options

parameter	meaning
`-r obj-name`	The name of a network object defined in the Policy Editor.
`-f`	Retrieve the licenses from a host and print them without storing them in the database.

fw rgetlic

`rgetlic` retrieves all licenses installed on the specified network object and stores them in the Management Station's internal database. The imported licenses are displayed unless the `-q` option is selected.

You can validate the retrieved licenses, using the `-v` option.

This command can be executed only on a Management Station.

Syntax

```
fw rgetlic obj-name [-q] [-v]
```

Options

TABLE 1-44 fw rgetlic options

parameter	meaning
`obj-name`	The name of a network object defined in the Policy Editor.
`-q`	Do not display the retrieved licenses.
`-v`	Validate the retrieved licenses.

VPN-1 Accelerator Card

fw accel

If a VPN-1 Accelerator Card is installed, it is enabled by default when VPN-1/FireWall-1 starts. You can also enable or disable it manually as well as obtain its status using `fw accel`.

When you enable or disable the VPN-1 Accelerator Card, current connections are not dropped. Instead, encryption continues in the hardware or software, accordingly.

Syntax

```
fw accel on | off | stat [-l]
```

Options

TABLE 1-45 fw accel options

parameter	meaning
`on`	Enable VPN-1 Accelerator Card.
`off`	Disable VPN-1 Accelerator Card.
`stat`	Obtain the status of the VPN-1 Accelerator Card.
`-l`	Report the status of the VPN-1 Accelerator Card using long format.

Diagnostics

lunadiag

A software diagnostics utility specific to the Luna accelerator card is available in the Luna package. The utility is documented in the file `lunadiag.txt`.

The locations of these files are given in TABLE 1-46.

TABLE 1-46 File Locations

file	location
executable	■ Solaris — `$FWDIR/bin/lunadiag` ■ NT — `$FWDIR\bin\lunadiag.exe`
documentation	■ Solaris — `$FWDIR/doc/lunadiag.txt` ■ NT — `$FWDIR\doc\lunadiag.txt`

`lunadiag` should show firmware version 1.24.

To determine the VPN-1 Accelerator Card driver version, enter the following command:

Solaris

```
modinfo | grep luna
```

The version number should be 3.9a.

NT

In the Explorer, right-click on `C:\WINNT\system32\drivers\LunaVPN.sys`. The version number, displayed in the **Properties** tab, should be 3.9a.

CHAPTER 2

VPN-1/FireWall-1 – Windows Interaction

In This Chapter

Registry

HKEY_LOCAL_MACHINE Entries

VPN-1/FireWall-1 modifies the Windows Registry under HKEY_LOCAL_MACHINE as follows:

SOFTWARE\CheckPoint\

Check Point Policy Editor

These values relate to the Check Point Policy Editor.

TABLE 2-1 SOFTWARE\CheckPoint\Policy Editor\4.1

Value Name	Value Data
Vendor	"CheckPoint" (The value determines the GUI icon.) This is a read only field.
Version	"4.1" (The VPN-1/FireWall-1 version) This is a read only field.
server_timeout	This is a field that is entered manually. It should be used only if your server's response (including network delay) is very slow. Enter the value in seconds. (The default value is 15 seconds.)

FW1

These values relate to the Management Server and the VPN/FireWall Module for backward compatibility.

TABLE 2-2 SOFTWARE\CheckPoint\FW1

Value Name	Value Data
AddSnmp	flag indicating whether the connection was made to NT SNMP. If NT SNMP is not installed, VPN-1/FireWall-1 adds an SNMP extension. ■ Ox0 — Connection made to NT SNMP ■ Ox1 — FireWall-1 SNMP extension was added
Encryption	flag indicating whether encryption is installed ■ Ox0 — non VPN ■ Ox1 — VPN
FireWall	flag indicating whether VPN/FireWall Module is installed
FWDIR	directory under which VPN-1/FireWall-1 software is installed
Management	flag indicating whether Management Server is installed
ProductName	product number of installed product (for example, CPFW-IGW-1)
Unlimit	flag indicating whether unlimited gateway is installed (used by configuration application)
CurrentVersion	4.1

FW1/4.1

These values relate to the Management Server and the FireWall Module version 4.1.

Starting with VPN-1/FireWall-1 version 4.1, the Registry will list products and packages that depend on VPN-1/FireWall-1 to function. The multi-string parameter DependentPkgs will indicate which dependent products and packages are installed. Each string value will be equal to the sub-key representing a product. Some examples of products are:

- FloodGate-1 (FG-1)
- Reporting Tool (RT)
- Certificate Manager
- SecuRemote
- Check Point Suite

For example, the value `DependentPkgs` is defined under the `FW1` key during installation and stays empty. If FloodGate-1 is subsequently installed, the FloodGate-1 installation program updates the key FW1/4.1 `DependentPkgs` to contain `FG-1`. If the

user attempts to uninstall VPN-1/FireWall-1 without uninstalling FloodGate-1, the user will see a warning and the uninstall will fail. During uninstallation of FloodGate-1, the FloodGate-1 uninstall program deletes the string `FG-1` from `DependentPkgs`.

TABLE 2-3 SOFTWARE\CheckPoint\FW1\4.1

Value Name	Value Data
AddSnmp	flag indicating whether the connection was made to NT SNMP. If NT SNMP is not installed, FireWall-1 adds an SNMP extension. ■ Ox0 — Connection made to NT SNMP ■ Ox1 — VPN-1/FireWall-1 SNMP extension was added
Auth	flag indicating whether authentication is installed ■ Ox0 — not installed ■ Ox1 — installed
DependentPkgs	flag indicating the installation of Check Point products in addition to VPN-1/FireWall-1 in the format: ProductXProductY where ■ ProductX is another product, e.g. FG-1 for FloodGate-1 ■ ProductX is another product, e.g. RT for Reporting Tool
Encryption	flag indicating whether encryption is installed ■ Ox0 — non VPN ■ Ox1 — VPN
FireWall	flag indicating whether VPN/FireWall Module is installed ■ Ox0 — not installed ■ Ox1 — installed
FWDIR	directory under which VPN-1/FireWall-1 software is installed: FW1-BV
HotFixes	flag indicating the Hot Fix version of the software, one of the following: ■ 1 ■ 3 ■ 9
Management	flag indicating whether Management Server is installed: ■ 0x0 — not installed ■ Ox1 — installed
PrevVersion	flag indicating the previous version:

TABLE 2-3 SOFTWARE\CheckPoint\FW1\4.1(continued)

Value Name	Value Data
ProductName	product name of installed product: VPN-1/FireWall-1
ServicePack	flag indicating the Service Pack version of the software:
Unlimit	flag indicating whether unlimited gateway is installed (used by configuration application): ■ 0x0 — not installed ■ 0x1 — installed

FW1\SnmpAgent

This value serves to make the connection between NT SNMP and the VPN-1/ FireWall-1 SNMP agent.

TABLE 2-4 SOFTWARE\FW1\SnmpAgent

Value Name	Value Data
Pathname	VPN-1/FireWall-1 SNMP DLL

License

These values relate to the Check Point license. Each new license appears as a separate value.

TABLE 2-5 SOFTWARE\CheckPoint\License

Value Name	Value Data
License	Check Point license string

SYSTEM\

CurrentControlSet\Services\FW1

TABLE 2-6 SYSTEM\CurrentControlSet\Services\FW1

Value Name	Value Data
DependOnService	standard NT service attribute (not modifiable)
DisplayName	“FireWall-1”
Error Control	NT value
Group	“NDISWAN”
ImagePath	location of binary
LoadMode	indicates which stage of the two stage process is taking place
Start	0x2 - automatic
Type	0x1 - kernel driver

CurrentControlSet\Services\FW1\Linkage

TABLE 2-7 SYSTEM\CurrentControlSet\Services\FW1\Linkage

Value Name	Value Data
Bind	device below to which VPN-1/FireWall-1 is bound
Export	name under which VPN-1/FireWall-1 is exported above
Route	NT value - set automatically during boot

CurrentControlSet\Services\FW1\Parameters

TABLE 2-8 SYSTEM\CurrentControlSet\Services\FW1\Parameters

Value Name	Value Data
Debug	debug level
IPForwarding	flag indicated whether
NewRAS	0x1 - indicates Microsoft Remote Access Service was installed after VPN-1/FireWall-1 installation

CurrentControlSet\Services\FW1\Performance

TABLE 2-9 SYSTEM\CurrentControlSet\Services\FW1\Performance values

Value Name	Value Data
Library	DLL containing functions for Close, Collect and Open values
Close	function (in Library DLL above)
Connect	function (in Library DLL above)
Open	function (in Library DLL above)
FirstCounter	Performance Monitor data
FirstHelp	Performance Monitor data
LastCounter	Performance Monitor data
LastHelp	Performance Monitor data

CurrentControlSet\Services\FW0 (for NT 4.0 only)

Under NT 4.0, the VPN-1/FireWall-1 driver is loaded in a two-stage process, as follows:

TABLE 2-10 FireWall-1 driver - two stage loading process (NT 4.0)

Step	Module Loading
1	TCP
2	FWO (VPN-1/FireWall-1 first stage)
3	NDIS
4	NDISWAN
5	FW1 (VPN-1/FireWall-1 second stage)

The FWO values relate to the first stage.

TABLE 2-11 SYSTEM\CurrentControlSet\Services\FW0

Value Name	Value Data
DisplayName	“FireWall-1”
Error Control	NT value
Group	“PNP_TDI”
ImagePath	location of binary - another copy of the VPN-1/FireWall-1 binary
LoadMode	indicates which stage of the two stage process is taking place
Start	■ 0x02 — automatic startup ■ 0x03 — manual startup ■ 0x04 — startup disabled
Type	0x1 - kernel driver

CurrentControlSet\Services\FW1SVC

TABLE 2-12 SYSTEM\CurrentControlSet\Services\FW1SVC

Value Name	Value Data
DisplayName	"Check Point FireWall-1 daemon"
Error Control	NT value
Group	"NDIS"
ImagePath	location of binary - another copy of the VPN-1/FireWall-1 binary
ObjectName	NT attribute
Start	■ 0x02 — automatic startup ■ 0x03 — manual startup ■ 0x04 — startup disabled
Type	0x1 - kernel driver

CurrentControlSet\Services\FW1SVC\Security

There are no FireWall-1 values under this key.

CurrentControlSet\Services\FW1-<NIC>\Parameters\Tcpip

These values are copied from the real NIC at boot time.

HKEY_CURRENT_USER Entries

VPN-1/FireWall-1 modifies the Windows Registry under HKEY_CURRENT_USER as follows:

SOFTWARE\CheckPoint\

FireWall-1 Policy Editor

These values relate to the VPN-1/FireWall-1 Windows Policy Editor.

TABLE 2-13 SOFTWARE\CheckPoint\Policy Editor\4.1

Value Name	Value Data
User Name	user name from last successful logon
Server	server name from last successful logon

Windows NT Performance Monitoring

VPN-1/FireWall-1 provides performance statistics for the Windows NT Performance Monitor.

To view VPN-1/FireWall-1 statistics, proceed as follows:

1 Open the **Performance Monitor** (in the **Administrative Tools** group).

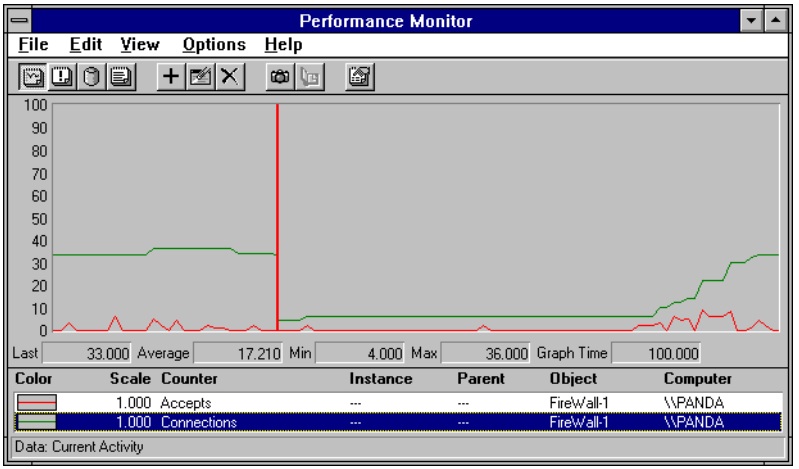


FIGURE 2-1 Performance Monitor window

2 Select the **Add Counter** button (+) in the toolbar.

The **Add to Chart** window (FIGURE 2-2) is displayed.

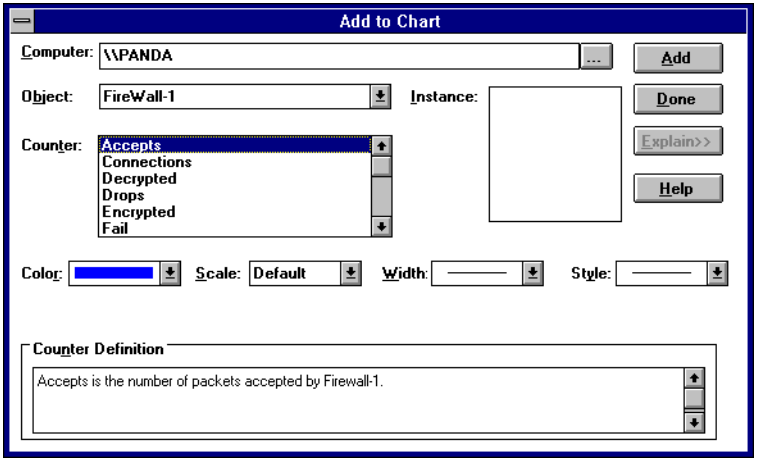


FIGURE 2-2 Add to Chart window

3 In the **Add to Chart** window, select **FireWall-1** under **Object**.

The **Counters** listbox shows all the VPN-1/FireWall-1 counters available.

4 Choose a counter you wish to monitor and click on **Add**.

You may choose as many counters as you like.

5 If you wish to see an definition of each counter (under **Counter Definition** at the bottom of the window), click on **Explain**.

6 Select **Done** to close the **Add to Chart** window.

The Performance Monitor window then shows the VPN-1/FireWall-1 statistics (see FIGURE 2-1). For the most part, these are the same statistics that are available in the VPN-1/FireWall-1 System Status View.

Windows NT Event Viewer

VPN-1/FireWall-1 posts System and Application (but not Security) events to the Windows NT Event Log. You can use the Windows NT Event Viewer application to view the Event Log.

VPN-1/FireWall-1 events are those whose Source is one of the following:

- FW1
- VPN-1/FireWall-1
- FW1SVC

CHAPTER 3

The INSPECT Language

In This Chapter

Introduction

INSPECT is an object-oriented, high-level script language that specifies packet handling by classifying packet content and state. The INSPECT engine examines virtually all communications taking place at and above the network level, and efficiently extracts the relevant data from each packet.

A Security Policy is defined using VPN-1/FireWall-1's graphical user interface. From the Security Policy, VPN-1/FireWall-1 generates an Inspection Script, written in INSPECT. Inspection Code is compiled from the script and loaded to the VPN/FireWall Modules on the network's FireWalled enforcement points.

The flow of information for VPN-1/FireWall-1 inspection is illustrated in FIGURE 3-1.

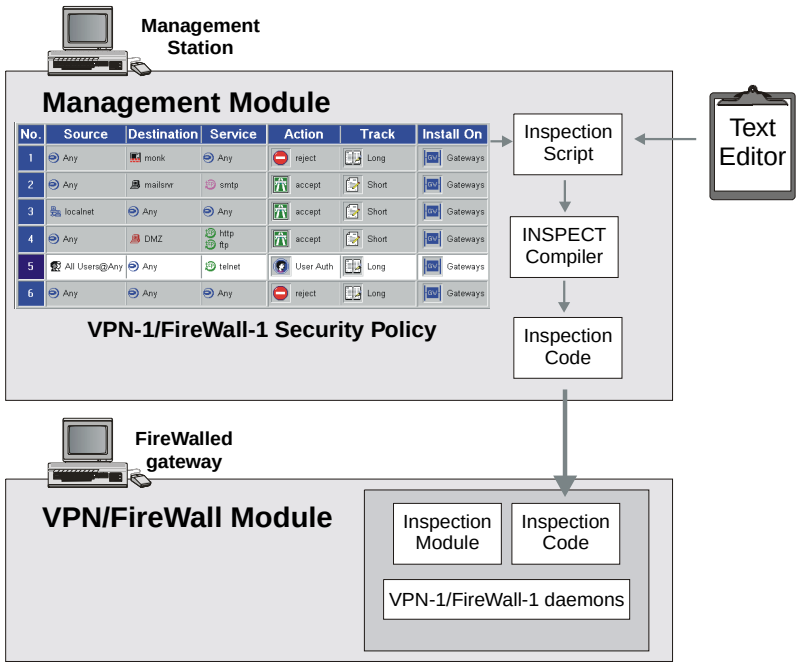


FIGURE 3-1 VPN-1/Firewall-1 Inspection - flow of information

You can view the Inspection Script corresponding to a given Security Policy by choosing **View** from the **Policy** menu in the Policy Editor.

INSPECT was designed specifically as a firewall language, and so it enables typical firewall actions (for example, accept, reject, log, *etc*.). An INSPECT script, or program, consists of a single source ASCII file, although this file may contain C preprocessor `#include` directives.

The INSPECT Compiler is a single phase compiler. It translates the Inspection Script in two stages:

- During the preprocessor stage, any C preprocessor directives are carried out.
- In the second stage, the INSPECT Script is transformed into low level Inspection Code.

The Inspection Code is transmitted on a secured control channel from the Management Station — the computer on which the Security Policy was defined — to the VPN-1/FireWall-1 daemons on the network objects that enforce the policy.

The VPN-1/FireWall-1 daemons load the Inspection Code into the VPN/FireWall Modules. The Inspection Code is run on a stack-based virtual machine. The virtual machine is placed in the FireWalled machine's kernel and inspects every IP packet passing through the machine.

To summarize, VPN-1/FireWall-1 inspection requires the following components:

TABLE 3-1 VPN-1/FireWall-1 Inspection Components

Component	Description
Inspection Script	An ASCII file (`*.pf`) in the INSPECT language. This file is generated from a Security Policy (`*.W` file), or can be written using a text editor. The file may be any combination of generated and manually written code.
Inspection Code	A file (`*.fc`) compiled from an Inspection Script (`*.pf`).
VPN/FireWall Module	A VPN-1/FireWall-1 software module running on a FireWalled host that executes Inspection Code.

INSPECT characteristics

To meet reliability and efficiency requirements, INSPECT has the following characteristics:

- There are no loops.
- Conditions are short circuits—if part of a conditional statement is evaluated as false, the rest of the statement is not evaluated.
- There is no explicit memory allocation.
- Function arguments are passed by value only.
- Functions do not support recursion.
- Functions return exactly one value.
- Source code is contained in a single file and there is no external linkage. However, the C preprocessor `#include` directive is allowed.
- The name space (that is, macros, functions, tables and formats) begins at the end of a name's definition and persists to the end of the file.

Using INSPECT

Administrators can tailor Inspection Scripts to their specialized security requirements in one of three ways:

1 By creating a User-Defined Service object using the GUI.

For more information, see Chapter 6, "Services," in *VPN-1/FireWall-1 Administration Guide*.

2. By including INSPECT scripts in `$FWDIR/lib/user.def` using the C preprocessor `#include` directive.

3. By creating an INSPECT Script named `$FWDIR/conf/defaultfilter` to be used as the default security policy.

   For more information, see Chapter 8, "Security Policy Rule Base," in *VPN-1/FireWall-1 Administration Guide*. Also see "Example" on page 104.

While you are learning INSPECT, you may find the following commands useful:

- `fwc`

  This command compiles an Inspection Script (`*.pf` file) but does *not* install the resulting Inspection Code (`*.fc` file).

- `fw load`

  This command compiles an Inspection Script and installs the resulting Inspection Code in a single step.

For additional information about these commands, see Chapter 1, "Command Line Interface."

Getting Started

The best way to get started with INSPECT is to write a simple program, compile it, load it to a FireWalled host and verify that the Inspection Code does what you expect it to do.

An Inspection Script corresponds to a Security Policy, and its most important elements are rule statements. The following script consists of a single rule statement:

```
accept [ 9 : 1] = 6;
```

This rule statement is read as "accept the packet if the value at byte 9 (for a length of 1 byte) is equal to 6." (In IP packets, byte 9 identifies the protocol, and a value of 6 indicates a TCP packet.) In short, this script accepts TCP packets.

To test this simple INSPECT script, proceed as follows:

1. Use the `fwc` command to compile the Inspection Script (`fwc name.pf`).

   The `fwc` command puts the Inspection code in `$FWDIR/tmp`. Your simple INSPECT script should compile successfully with no errors.

2. Verify that your host is FireWalled.

   Use the `fwstart` command to start the VPN/FireWall Module.

3. Use the `fw load` command to compile the Inspection Script and install the resulting Inspection Code in a single step. For example:

```
fw load my_code.pf
```

4 Verify that only TCP packets are allowed to access the current host.

For example, the Telnet protocol is accepted and the PING protocol is rejected.

Grammar and Syntax

An Inspection Script corresponds to a Security Policy, and its most important elements are rule statements.

In the Rule Base Editor, a rule is composed of six elements, as follows:

Source — where the packet is coming from

Destination — where the packet is going

Services — the type of application

Action — what is to be done with the packet

Track — whether to log the packet or generate an alert

Install On — the VPN/FireWall Module or Inspection Module that will enforce this rule

In INSPECT, a rule typically has the following form:

```
scope  action  condition;
```

This section describes each of these elements in greater detail.

Scope

This element is equivalent to Install On. It specifies the FireWalled objects that will enforce the rule, and uses the following syntax:

```
direction interfaces@hosts
```

TABLE 3-2 Scope Elements

element	meaning
direction	One of the following macros (defined in `$FWDIR/lib/fwui_head.def`): `inbound` (or `=>`) — incoming `outbound` (or `<=`) — outgoing `eitherbound` (or `<>`) — incoming and outgoing
interfaces	The network interface(s) on which packets are to be examined.
@	A required separator.
hosts	The host(s) on which packets are to be examined.

Consider the following example:

```
inbound all@natasha
accept tcp
```

This rule statement means, “Check all the inbound packets on all interfaces of the FireWalled host `natasha`. If they are TCP packets, accept them.”

Action

A rule’s Action element specifies what is to be done with a packet. An Action may be any of the following:

TABLE 3-3 Possible Actions

action	See...
`accept`	page 95
`drop`	page 98
`hold`	page 99
`reject`	page 102
`vanish`	page 103

Condition

The condition part of an INSPECT rule allows you to control access to a host based on the source and/or destination of each packet, as well as according to the type of service requested. A rule’s condition also allows you to specify if and how a packet should be tracked.

Extracting Information from a Packet

Information in a packet—such as its source, destination, and services—can be accessed in several ways:

- By specifying the location of the data within the packet.
- By using predefined INSPECT macros and functions.
- By writing your own INSPECT macros and functions.

Information can be directly extracted from a packet using the following syntax:

```
[offset : length , order]
```

TABLE 3-4 extracting data from packet

argument	meaning
`offset`	The offset within the packet, in bytes.
`length`	The number of bytes to be extracted: 1, 2, or 4. This argument is optional. The default is 4.
`order`	The protocol used: `b` (big endian) or `l` (little endian, or host order). This argument is optional. The default is little endian. For more information, see "Data Storage Conventions" on page 83.

For example,

```
[9 : 1]
```

means that the one byte beginning at byte 9 of the packet should be treated as a Little Endian integer.

The expression,

```
[12, b]
```

means that the 4 bytes beginning at byte 12 of the packet should be treated as a Big Endian integer.

INSPECT header files provide function and macro definitions that simplify extracting information about a packet's source and destination, and the requested service.

For example, in IP packets, byte 9 identifies the protocol. The file `$FWDIR/lib/tcpip.def` defines the macro `ip_p` as the value of byte 9 of the packet, for the length of 1 byte.

```
#define ip_p [9 : 1]
```

Based on this definition, you can now specify various services as follows:

```
define tcp { ip_p = 6 };
define udp { ip_p = 17 };
define icmp { ip_p = 1 };
```

Definitions for common services can be found in `$FWDIR/lib/base.def`.

For more information on using INSPECT macros and functions, see “Functions and Macros” on page 94.

Note – `#define` is a C preprocessor directive and `define` is an INSPECT statement. For further details, see “#define” on page 84, and “define” on page 96.

Track

Depending on a rule’s Track element, communications may be logged or an alert may be issued. You can define the format of a log entry by creating a format list, or you can use the log formats defined in the file `$FWDIR/lib/formats.def`.

The `log` command creates a log record in the specified format. However, it is often more convenient to use the `LOG` macro since it automatically handles rule numbers and takes into account certain parameters that the `log` command does not. For further information, see the topics listed in TABLE 3-5.

TABLE 3-5 Tracking

topic	See...
`LOG` macro	page 103
`log` command	page 100
format lists	page 92

Syntax of Conditional Expressions

The following rule statement illustrates the differences between conditional expressions in INSPECT and in C:

```
accept [ 9 : 1] = 6;
```

- The = operator means test for equality (rather than assignment).
- The conditional test (`[ 9 : 1] = 6`) is *not* preceded by the `if` keyword.

Logical Operators

INSPECT uses the following logical operators:

- The comma (`,`) is the logical AND operator.
- The keyword `or` is the logical OR operator.
- The logical OR operator takes precedence over the logical AND operator. This is the only difference in operator precedence between INSPECT and C.

For example, the rule statement below means, “Accept the packet if it is both TCP and Telnet.”

```
accept (tcp, telnet);
```

This next rule statement means, “Accept the packet if it is TCP and either Telnet or FTP.”

```
accept (tcp, telnet or ftp);
```

Operator Precedence

In C, the expression,

```
 X && Y || Z // read as "X AND Y OR Z"
```

is understood as ((X AND Y) OR Z). That is, AND is evaluated before OR.

In INSPECT, the expression

```
 X and Y or Z
```

is understood as (X AND (Y OR Z)). That is, OR takes precedence over AND.

Parentheses — “(”and “)” — can be used to force operator precedence. There is no penalty for superfluous parentheses.

The following rule statement illustrates the use of parentheses to force operator precedence:

```
accept (tcp, telnet or ftp) or (udp, snmp);
```

This means, “Accept the packet if either of the following conditions is true:”

- The packet is TCP and either Telnet or FTP.
- Or the packet is both UDP and SNMP.

Without parentheses, this statement would mean, “Accept the packet if both of the following conditions are true:”

- The packet is TCP and either Telnet, FTP, or UDP.
- And the packet is also SNMP.

Note – `telnet`, `tcp`, `udp`, `snmp` and `ftp` are defined in the file `$FWDIR/lib/base.def`.

Order of Evaluation

Each INSPECT condition is assigned a Boolean Value as follows:

- 0 if false.
- Nonzero if true.

Operators are evaluated only if necessary.

For example, in the expression below, B is evaluated only if A is false.

```
A or B
```

In the following case, B is evaluated only if A is true.

```
A , B
```

The compiler also optimizes expressions. For example:

```
(A , B) or (A , C)
```

is compiled to:

```
A , (B or C)
```

Comments

INSPECT comments have the same syntax as C++ comments.

The characters /* introduce a comment. The characters */ end a comment. For example:

```
accept [ 9 : 1] = 6; /* this is a comment
that spans two lines */
```

The characters // introduce a comment that is terminated by the end of line. For example:

```
accept [ 9 : 1] = 6;  // this is an end-of-line comment
```

Example

Consider the following INSPECT rule:

```
#include "fwui_head.def"
inbound all@natasha                     // Scope (Install On)
accept                                  // Action
 (tcp, telnet or ftp),                  // Services
 (ip_src = doors or ip_src = well),     // Source
 (ip_dst = natasha),                    // Destination
 LOG(long,LOG_NOALERT,1);               // Track
```

The scope of this rule is:

```
inbound all@natasha
```

The action is:

```
accept
```

The condition is:

```
 (tcp, telnet or ftp),                  // Services
 (ip_src = doors or ip_src = well),     // Source
 (ip_dst = natasha),                    // Destination
 LOG(long,LOG_NOALERT,1);               // Track
```

This rule statement means, “Check all the inbound packets on all interfaces of the FireWalled host `natasha`. Determine if the packets meet the following conditions:

- They are TCP and either Telnet or FTP packets.
- They originate from the machines `doors` or `well`.
- They are headed for `natasha`.

If the packets meet all of the above conditions, accept them and log them using the long log format.”

Constants

This section lists the constants available in INSPECT.

Numeric Constants

There are no floating point constants in INSPECT. INPSECT supports 32-bit integers.

Integer constants are expressed in the following standard formats:

hexadecimal integers	a number beginning with `0x`	e.g. `0x4f`
octal integers	a number beginning with `0`	e.g. `0777`
decimal integers	any other number	e.g. `23`

Time Constants

time of day	Three decimal integers separated by two colons.	e.g. `23:30:00`
day in month	Three character abbreviations or full month names, followed by a day number.	e.g. `Jan 22`
day in week	Three character abbreviations or full day names.	e.g. `sun`

Special FireWall constants

Since INSPECT is designed especially for FireWall purposes, it recognizes the following special communication entities:

networks and hosts, expressed as IP address constants	Four decimal integers separated by three periods.	e.g. `192.0.0.24`
	Three decimal integers separated by two periods.	e.g. `192.0.0`
	Two decimal integers separated by one period.	e.g. `192.0`
domains	A period, followed by two or more strings separated by periods.	e.g. `.checkpoint.com` `.tau.ac.il`
interfaces	These are identified depending on the context in which they are used.	e.g. `le0`, `hme0`

Network and communication entities may be used as special constants in both lists and expressions. Special purpose commands that handle some of these entities are described in “Packet Operators” on page 82.

Operators

The following operators are available in INSPECT:

Arithmetic Operators

+	addition
-	subtraction
/	division
*	multiplication
%	modular division

Bitwise Logical Operators

&	bitwise AND
|	bitwise OR
^	bitwise XOR
>>	shift right
<<	shift left

Relational Operators

<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to
= and is	equal
!= and is not	not equal

Logical Operators

,	logical AND
or	logical OR
xor	logical XOR

Date and Time Operators

Certain operators return information about the current date and time, as follows:

`date`	Returns the day of the month (0-30, where 0 is the first day of the month).
`day`	Returns the day of the week (0-6, where 0 is Sunday).
`tod`	Returns the time of day in seconds since midnight.

You can use these in conditional expressions, as follows:

```
accept tcp, tod < 28800 // equivalent to 08:00:00
```

Packet Operators

Certain operators return information about the current packet, as follows:

`direction`	Returns 0 (inbound) or 1 (outbound).
`host`	Returns the canonical IP address of the machine on which on the VPN/FireWall Module is running.
`ifaddr`	Returns the packet's interface address.
`interface`	Returns the packet's interface (le0, le1, etc.)
`packetid`	Returns a unique number assigned to the packet by VPN-1/FireWall-1 when the packet passed through the interface.

You can use these in conditional expressions, as follows:

```
drop direction=1
```

Other Operators

`[]`	One of the following: • an index to a table (e.g. `udp_tab[12,24]`)—for more information, see "Tables" on page 87. • the offset within the packet being examined (e.g. [3:1])—for more information, see "Extracting Information from a Packet" on page 74. • the offset within a segment register (e.g. `sr[12:2,b]`)—for more information, see "Segment Registers" on page 86.
`=>`	incoming—for more information, see "Scope" on page 73.
`<=`	outgoing—for more information, see "Scope" on page 73.

Data Storage Conventions

"Big Endian" and "Little Endian" describe two different hardware conventions for storing data.

Suppose the following data is stored in the following memory addresses:

data	1	2	3	4
address	1000	1001	1002	1003

In the Big Endian convention, the data types at memory address 1000 have these values:

long integer	1234
short integer	12
byte	1

Given the same data, in the Little Endian convention, the data types at memory address 1000 have these values:

long integer	4321
short integer	21
byte	1

In the expression,

```
[12,b]
```

the `b` indicates that the word (4 bytes, the default length when no length is specified) is to be treated as a Big Endian integer.

In the expression,

```
[12,l]
```

the `l` indicates that the word is to be treated as a Little Endian integer.

To ensure portability, always use `b` when referring to data in the header of any Big Endian protocol, for example, TCP/IP.

See also "Extracting Information from a Packet" on page 74.

C Preprocessor Directives

INSPECT uses the C preprocessor to preprocess the Inspection Script source file before compiling it. The following C preprocessor directives have no meaning in the context of a VPN-1/FireWall-1 Inspection Script:

- `#error`
- `#line`
- `#pragma`

The preprocessor directives most commonly used in INSPECT scripts are listed below.

#define

The `#define` directive enables macro replacement, e.g. *`expression1`* is replaced with *`expression2`* inline. These expressions may or may not have parameters.

```
#define expression1(a,b,c) expression2     // with parameters
```

or

```
#define expression1 expression2            // without parameters
```

#include

The `#include` directive specifies additional INSPECT source files to be included in your script. For example:

```
#include "fwui_trail.def"
```

#ifdef

You can use the preprocessor `#ifdef` directive to conditionally compile parts of an Inspection Script.

#include Files

In the `$FWDIR/lib` directory there are a number of files that contain various INSPECT functions and macro definitions. These files are always included by the Inspection Scripts generated by VPN-1/FireWall-1. You may find it useful to include some of these files in Inspection Scripts you write yourself.

TABLE 3-6 Some Useful include Files

file name	contents
`auth.def`	Authentication code.
`base.def`	Basic definitions and specific services (FTP, RealAudio, etc.)
`code.def`	The connection mechanism, Suspicious Activity Monitoring, etc.
`dcerpc.def`	DCE-RPC and DCOM support.
`init.def`	Initalization code.
`encrypt.def`	Encryption code.
`formats.def`	Formats for log entries, alerts, and traps that are used in the VPN-1/FireWall-1 User Interface. For example, the Long and Short log formats.
`fwui_head.def`	Many useful macro definitions. Also includes other `*.def` files. This file is generally included at the top of the Inspection Script generated by the GUI.
`fwui_trail.def`	The implicit drop rule — usually included at the end of an Inspection Scrip generated by the GUI.
`snmp.def`	SNMP support.
`table.def`	table definitions.
`tcpip.def`	Definitions for parsing packet headers (IP, TCP, UDP, etc.)
`traps.def`	Trap definitions.
`user.def`	Your user-defined INSPECT code.
`xtreme.def`	VXTREME protocol support.

Identifiers

Identifiers or names refer to variety of entities: functions, macros, tables, registers, etc. Identifier names are case-sensitive. INSPECT does not impose a limit to a name's length.

The compiler tries to resolve names using various external databases, such as `/etc/services`, based on the context in which the name is used. If a name is defined as an INSPECT identifier (for example, a table, macro, or function) then this definition hides the database value of the name.

Reserved Words

The following words (all in lower case only) are reserved words in INSPECT:

TABLE 3-7 INSPECT Reserved Words

accept	and	call	date
day	deffunc	define	delete
direction	domains	drop	dynamic
expcall	expires	export	format
from	fwline	fwrule	get
hold	host	hosts	if
ifaddr	ifid	in	interface
interfaces	keep	limit	log
modify	netof	nets	nexpires
not	or	packet	packetid
pass	record	refresh	reject
set	static	to	tod
vanish	xor		

In addition, the following are also reserved words:

- names of the days of the week (for example, `Sunday`, `sunday` and `sun`),
- names of the months (for example, `January`, `january` and `jan`)
- constructs of the form `[S|s][r|R]n` (where `n` is a decimal number)

It is recommended that you use service and protocol names for their original purpose. That is, when using Telnet, FTP, and so on, do not hide the system constants.

You should not use reserved words as object names, or use the character "&" in an object name.

Segment Registers

A segment register is defined as follows:

```
srn
```

where *n* is a decimal number between 0 and 15. For example, `sr6` and `sr13` are valid segment register names, but `sr16` is not.

A segment register can be used to store a value for later use.

For example, the code below means, “Accept the packet if byte 12 equals byte 14.”

```
set sr1 [12];
accept sr1 = [14,l];
```

For more information on extracting values from a packet, see “Extracting Information from a Packet” on page 74.

Tables

This section describes the two types of tables available in INSPECT: dynamic and static. Each table must be explicitly defined before it is used.

Dynamic Tables

A dynamic table is one whose entries change as the Inspection Code is being executed. That is, the number of entries in a dynamic table is not fixed at the time of creation, and entries can be added, deleted, or modified during execution.

Creating Dynamic Tables

To define a dynamic table, use the `dynamic` keyword as follows:

```
table-name = dynamic {};
```

`table-name` is the table’s name.

For example,

```
ThisTab = dynamic {};
```

creates a dynamic table named `ThisTab`. The table’s entries are not specified when the table is defined because they are added and deleted dynamically.

An optional list of attributes may be specified after the `{}`. These attributes are listed below:

TABLE 3-8 Dynamic Table Attributes

attribute name	meaning
`expcall`	Call the given (by number) kernel functions when a table entry expires (see “call” on page 95).
`expires` *n*	Entries not accessed for *n* seconds are removed from the table. If this attribute is specified for a table, it can be overridden for individual elements. For details, see “record” on page 101.
`free function` *x*	Call this function when an entry is deleted or expires.
`hashsize`	The size of the hash — should be close to table size.

TABLE 3-8 Dynamic Table Attributes (continued)

attribute name	meaning
`implies` *`table_name`*	An entry deleted from this table will be deleted from the table named *`table_name`* as well.
`kbuf` *x*	The x^{th} argument in the value section is a pointer to an internal data structure (primarily used in encryption).
`keep`	Keep the table's entries even if the Security Policy is reinstalled.
`limit` *x*	Limit this table to a maximum of *x* entries.
`nexpires`	Elements do not expire, but are removed only when explicitly deleted. This is the default setting.
`refresh`	Reset the expiration timer whenever an entry is accessed.
`synch`	Synchronize this table if using VPN-1/FireWall-1 synchronization.

For example, `my_table` is defined below as a dynamic table whose entries expire if they are not used for 60 seconds:

```
my_table = dynamic {} expires 60 refresh;
```

When a Security Policy is reinstalled on a FireWalled, the tables are all reset. A table will not be reset if both of the following conditions are true:

- The table was defined with the `keep` attribute.
- The table's name and sequential number are unchanged.

  A name and sequential number are assigned to each table internally by VPN-1/FireWall-1 in accordance with the table's position in the Inspection Script.

Dynamic Table Entries

Dynamic tables are associative. That is, an entry is identified by its keys rather than by its position in the table.

An entry contains keys and optional values. A semicolon (`;`) separates the list of keys from the list of values, as follows:

```
<key1, key2, ... ; value1, value2, ...>
```

An entry need not have the same number of keys and values.

For example,

```
<src, dst, dport>
```

is an entry that is not associated with a value.

However,

```
<src, dst, dport; ip_p>
```

has a value of `ip_p`. This value can be retrieved as follows.

```
my_table [src, dst, dport]
```

To retrieve multiple values, use the `get` command:

```
get <src, dst, dport> from my_table to sr1
```

This expression returns each of the values associated with the keys `<src, dst, dport>` in the segment registers `sr1`, `sr2` and so on. For example, if there are three values associated with these keys, the first is stored in `sr1`, the second in `sr2` and the third in `sr3`.

Dynamic Table Operations

INSPECT has special commands for modifying the contents of a table.

TABLE 3-9 Table Operations

action	See...
`delete`	page 97
`get`	page 98
`in`	page 99
`modify`	page 100
`record`	page 101

- To add an entry to a dynamic table:

```
record <src,sport,dst> in udp_out;
```

- To record an entry in the dynamic table with an expiration timer different from the default expiration timer for the table:

```
record <src,sport,dst @timeout> in my_table;
```

`timeout` is the number of seconds.

- To modify an entry in the dynamic table without resetting the expiration timer to zero:

```
modify <src,sport,dst> in my_table;
```

- To delete an entry from a dynamic table:

```
delete <src,sport,dst> from my_table;
```

- To check if an entry is in a dynamic table:

```
<src,sport,dst> in my_table
```

- To delete an entry from a dynamic table:

```
delete <src,sport,dst> from my_table;
```

Example

Here is a simple but powerful example of the use of dynamic tables:

```
#include "fwui_head.def"

udp_out = dynamic {} expires 60 refresh;

accept udp, direction = 1, record <src,sport,dst> in udp_out;
accept udp, direction = 0, <dst,dport,src> in udp_out;
```

`udp_out` is declared as a dynamic table. An entry is deleted after 60 seconds if no corresponding UDP packet is encountered.

The context tuple (source, port, and destination) of every outgoing UDP packet (`direction = 1`) is recorded in `udp_out`.

An incoming UDP packet (`direction = 0`) is accepted only if its context tuple is in `udp_out`. In other words, the incoming packet must be a reply to a previously registered outgoing packet.

Note – INSPECT short-circuits compound `AND` conditions as soon as it encounters a FALSE condition, so the `record` will only execute if `direction = 1` is TRUE. In other words, the order of the expressions is important here. For more information, see "Order of Evaluation" on page 78.

For an explanation of the `direction` operator, see "Packet Operators" on page 82.

Static Tables

A static table is one whose elements are fixed when the table is created, and cannot be modified. In other words, a static table stores a list of keys. During runtime, you can test whether certain values are in the static table, and based on this information, determine what action to take.

Creating Static Tables

Use the `static` keyword to define a static table, as follows:

```
table_name = static{list-of-entries};
```

table-name is the table's name. *list-of-entries* is a comma-delimited list of constant expressions.

For example,

```
GWList = static{gatekeeper, gatekeeper_le0, gatekeeper_le1 };
```

creates a static table called `GWList`, that contains three entries: `gatekeeper`, `gatekeeper_le0`, and `gatekeeper_le1`.

Lists

A list is a typed static table, and is defined as follows:

```
table-name = type{list-of-entries};
```

table-name is the table's name. *list-of-entries* is a comma-delimited list of constant expressions. *type* specifies the type of static table, and can be one of the following:

- domains
- nets
- hosts
- interfaces
- format

For example,

```
domain_list = domains {.security.com,.iconnet.com };
```

is a domain list, and the condition,

```
ip_src in domain_list
```

tests whether `ip_src` (the packet's source IP address) belongs to the domain defined by the given networks (`.security.com` and `.iconnet.com`).

Similarly, for a "proper" network (that is, a network whose netmask is the one implied by its class), you can write:

```
net_list = nets { 192.9.200.0, 10.0.0.0, 132.64.0.0 };
```

The statement,

```
ip_dst in net_list
```

tests whether `ip_dst` belongs to one of the networks in `net_list`.

Format Lists

A format list is a list used in creating log records.

For example, consider the following definition:

```
short = format {
    <"proto", proto, ip_p>,
    <"src", ipaddr, src>,
    <"dst", ipaddr, dst>,
    <"service", port, dport>
};
```

This format list defines a format named `short`, which consists of a list of four tuples. Each format list tuple is made up of three elements, as follows:

TABLE 3-10 Elements of a Format List

element	meaning	
label	The element's label, as written in the VPN-1/FireWall-1 Log, for example, "`src`". TABLE 3-11, at the end of this section, lists the labels VPN-1/FireWall-1 recognizes, as well as their meaning, and the Log Viewer column in which they are displayed. Elements whose labels are not recognized by VPN-1/FireWall-1 are displayed in the **Info.** column.	
type	One of the following predefined types:	
	type	**meaning**
	`int`	decimal 32 bit signed integer
	`uint`	decimal 32 bit unsigned integer
	`hex`	hexadecimal representation of a 32 bit unsigned integer
	`ipaddr`	an IP address
	`service, port`	TCP or UDP port number
	`proto`	IP protocol number
	`string`	an ASCII string (not used in the VPN/FireWall Module)
value	The actual value written to the log.	

Predefined log formats are in the file `$FWDIR/lib/formats.def`.

See also "LOG" on page 103 and "log" on page 100.

TABLE 3-11 Predefined Format Labels

format label	meaning	Log Viewer column displayed in
action	The action taken by VPN-1/FireWall-1.	**Action**
d_port	The connection's destination port.	**Info.**
direction	The direction of the connection.	**Inter.**
dst	The IP address of the connection's destination.	**Destination**
if_num	The interface index number: a positive integer.	**Inter.**
info_url	A URL containing third party information.	**Info.**
message	A description of the logged event (e.g. "Virus found and cured.").	**Info.**
operation	The action taken when this log message was generated.	**Info.**

TABLE 3-11 Predefined Format Labels

format label	meaning	Log Viewer column displayed in
originator	A string identifying the originator of the log message.	**Info.**
originator_ip	An IP address identifying the origin of the log message.	**Info.**
originator_port	The port number of the originator of the log message.	**Info.**
product	The product generating the log message.	**Info.**
proto	The connection's higher level protocol (the "real" IP protocol value).	**Proto.**
recovery_solution	The action to be taken in response to this message (e.g. "Restart the application").	**Info.**
s_port	The connection's source port.	**S_Port**
service	The connection's service (e.g. "Telnet").	**Service**
severity	One of "URGENT", "HIGH", "MEDIUM", "LOW", "WARNING".	**Info.**
src	The IP addressof the connection's source.	**Source**
status	The status of application.	**Info.**
url	The URL related to the log message.	**Info.**
user	The connection's user (e.g. "Monica")	**User**

Functions and Macros

Macros and functions enable you to break large computing tasks into smaller ones, making your code clearer and easier to read.

The differences between functions and macros include the following:

- Macros are faster than functions, since there is no pushing and popping of parameters on the stack.
- Using macros gives Inspection Scripts a simple linear structure.
- Using functions leads to smaller Inspection Scripts.

Both functions and macros must be defined before they are called. In addition, functions have the following characteristics:

- Functions return exactly one value.
- Function parameter passing is strictly by value.
- No recursion is allowed.
- There are no function prototypes.

For more information on macros, see "define" on page 96.

For more information on functions, see "deffunc" on page 96.

INSPECT Commands

The section lists available INSPECT commands.

TABLE 3-12 INSPECT commands

action	See...	action	See...
`accept`	page 95	`log`	page 100
`call`	page 95	`modify`	page 100
`deffunc`	page 96	`netof`	page 101
`define`	page 96	`record`	page 101
`delete`	page 97	`reject`	page 102
`drop`	page 98	`set`	page 102
`export`	page 98	`vanish`	page 103
`get`	page 98		
`hold`	page 99		
`in`	page 99		

accept

The `accept` action accepts a packet if `condition` is true.

Syntax

```
accept condition;
```

For more information on the syntax of `condition`, see “Condition” on page 74.

Example

```
accept tcp, telnet or ftp;
```

accepts the packet if the condition `(tcp, telnet or ftp)` is `TRUE`.

call

The `call` command enables an Inspection Script to call an externally defined function, identified by its kernel number.

Syntax

```
call (function-number, <tuple-of-arguments>)
```

Example

```
call (KFUNC_IPPOOL_ALLOCATE, <src, sport, dst, dport, ip_p>)
```

deffunc

The `deffunc` command defines an INSPECT function. For an overview of INSPECT functions, see “Functions and Macros” on page 94.

Syntax

```
deffunc function-name(argument-list) { body };
```

Function arguments are optional. To declare a function without arguments, use the following syntax:

```
deffunc function-name { body };
```

`body` consists of exactly one INSPECT statement. The function returns the value of this statement.

Example

The following function checks whether the current packet is a TCP packet:

```
deffunc tcp {ip_p=6};
```

define

The `define` command creates an INSPECT macro. For an overview of INSPECT macros, see “Functions and Macros” on page 94. For a list of predefined macros, see “INSPECT Macros” on page 103.

The `define` command is distinct from the `#define` directive (see “C Preprocessor Directives” on page 83).

The `define` operator is evaluated during compilation and assigns a meaning to an entity in a particular context.

Syntax

```
define macro-name(argument-list) { body };
```

Macro arguments are optional. To declare a macro without arguments, use the following syntax:

```
deffunc macro-name { body };
```

body consists of exactly one INSPECT statement.

Examples

The following script accepts TCP packets:

```
define tcp {ip_p = tcp};
accept tcp;
```

The use of the token tcp twice in the define statement is unambiguous because:

- The compiler knows the meaning of tcp from its internal tables.
- The compiler resolves {ip_p = tcp} first.

When evaluating the accept statement, the only meaning of tcp known to the compiler is that of the defined macro.

Now consider the following variation:

```
define tcp {ip_p = 6};
accept (ip_p = tcp);
```

Though tcp has two possible meanings in this script, the accept statement still compiles correctly because only one meaning is appropriate to the context.

delete

The delete command deletes the specified entry from a table.

Syntax

```
delete <entry keys> from table-name;
```

For more information on table entries, see “Dynamic Table Entries” on page 88.

Example

```
delete <src, sport, dat> from udp_out
```

This statement deletes the entry matching <src, sport, dat> from the table named udp_out.

drop

The `drop` action drops a packet without notifying the sender if `condition` is true.

Syntax

```
drop condition;
```

For more information on the syntax of `condition`, see “Condition” on page 74.

Example

```
drop (net_in(ip_src, cp_net_128, cp_net_128_netmask));
```

This statement drops the packet if the condition `(net_in(ip_src, cp_net_128, cp_net_128_netmask))` is `TRUE`.

export

The `export` command exports the specified chunk of INSPECT code to another file.

`export` is used to make data that is not part of the Inspection Code available to the VPN/FireWall Module. The `export` statement usually appears at the beginning of Inspection Scripts generated by VPN-1/FireWall-1.

Syntax

```
export {
   INSPECT-code
} .xxx ;
```

If the file being compiled is *name.pf,* the `export` command exports the *INSPECT-code* between the brackets (`{` and `}`) to the file *name*`.xxx`.

Example

```
export {
   INSPECT-code
} .set
```

If this statement is located in the file `abcdef.pf`, then *INSPECT-code* is exported to `abcdef.set`.

get

The `get` command retrieves multiple values corresponding to a table entry and stores them the segment registers beginning with the designated segment register.

Syntax

```
get <entry> from table-name to srn;
```

n may be between 0 and 15.

If (n + the number of values associated with the entry)is more than 16 (the available number of registers), the additional values will be lost. For example, if you try to retrieve three values and store them beginning with sr14, the third value will be lost.

For more information on segment registers, see “Segment Registers” on page 86.

For more information on table entries, see “Dynamic Table Entries” on page 88.

Example

Suppose the entry <src, sport, dport> in the table udp_out has 3 values associated with it. The statement below retrieves these three values and stores the first one in sr3, the second in sr4, and the third in sr5.

```
get <src, sport, dport> from udp_out to sr3;
```

hold

The hold action holds a packet in the kernel if *condition* is true. The packet is neither passed not rejected. Its status can only be changed by the VPN-1/FireWall-1 daemon.

Syntax

```
hold condition;
```

For more information on the syntax of condition, see “Condition” on page 74.

Example

The statement below holds the packet if the condition tcp, telnet or ftp is TRUE.

```
hold tcp, telnet or ftp;
```

in

The in operator tests whether a value is in a table.

Syntax

```
<entry> in table
```

For more information on table entries, see “Dynamic Table Entries” on page 88, and “Static Tables” on page 91.

Example

```
<dst,dport,src> in udp_out
```

This statement tests whether the specified entry is in the `udp_out` table.

log

The `log` command creates a log record in the specified format.

A format list consists of one or more tuples. Each tuple consists of three elements. The values of the expressions in the third element of the tuples are written to the log record. In addition, the following standard fields are also written to the log record:

- timestamp
- address of FireWalled host (or gateway) that created this log record
- interface
- direction
- action

See also “LOG” on page 103.

Syntax

```
log format;
```

format is a format list. For more information, see “Format Lists” on page 92.

Example

The following statement creates a log record in the `short` log format:

```
log short;
```

modify

The `modify` command adds an entry to a dynamic table. As opposed to the `record` command, if the entry already exists in the table, the `modify` command does *not* reset its expiration timer to zero. This is true even if the table is defined with the `refresh` attribute.

Syntax

```
modify <entry> in table-name;
```

For more information on table entries, see “Dynamic Table Entries” on page 88.

Example

```
modify <src,sport,dst> in udp_out;
```

netof

The `netof` operator tests whether an IP address is a part of a network.

Syntax

```
netof ip_address
```

ip_address is an IP address constant (see “Special FireWall constants” on page 80).

Example

```
(netof ip_src = big-net)
```

This statement tests whether `ip_src` is part of the network `big-net`, according to the network mask implied by `big-net`’s class.

record

The `record` command adds an entry to a dynamic table.

If the table was defined using the `refresh` attribute, and the entry already exists in the table, the expiration timer is reset to zero.

Syntax

```
record <entry> in table-name;
```

For more information, see “Dynamic Table Entries” on page 88.

The default expiration timer can be overridden using the *@timeout* parameter, as follows:

```
record <entry @timeout> in table-name;
```

`timeout` is the number of seconds.

Examples

Suppose the table `udp_out` is defined as follows:

```
udp_out = dynamic {} expires 60;
```

The statement below records the entry `<src, sport, dst>` in `udp_out`.

```
record <src,sport,dst> in udp_out;
```

The following statement overrides the value of the default expiration timer of 60 seconds. The entry recorded in the table will have an expiration timer of 120 seconds.

```
record <src,sport,dst @ 120> in udp_out;
```

reject

The `reject` action rejects a packet, and, for TCP and UDP, signals the originator that the attempt was forcibly denied.

Syntax

```
reject condition;
```

For more information on the syntax of `condition`, see “Condition” on page 74.

Example

```
reject tcp, ident;
```

If the condition `tcp, ident` is `TRUE`, then `reject` drops the packet and for TCP, signals the originator that the attempt was forcibly denied.

set

The `set` command assigns a value to a segment register.

Syntax

```
set srn value;
```

n is 0-15 (see “Segment Registers” on page 86).

value is any VPN-1/FireWall-1 constant (see “Constants” on page 79).

Example

```
set sr6 12;
```

This statement assigns the value 12 to the segment register sr6.

vanish

The vanish action drops a packet without a trace, and for packets of an established TCP connection, does not perform the mangling described in "Established TCP Connections" in Chapter 8 of *VPN-1/FireWall-1 Administration Guide*. The drop and reject actions do perform this mangling.

Syntax

```
vanish condition;
```

For more information on the syntax of condition, see "Condition" on page 74.

Example

```
vanish tcp, telnet or ftp;
```

This statement drops the packet without a trace if the condition (tcp, telnet or ftp) is TRUE.

INSPECT Macros

The following INSPECT macros are predefined.

LOG

The LOG macro is defined in the file fwui_head.def.

In addition to providing all the functionality of the log command, the LOG macro has the following advantages:

- The LOG macro automatically logs the rule number under the "rule" log field.
- The LOG macro checks the type of packet (e.g. ICMP, RPC, TCP) and logs the appropriate information (e.g. program number for RPC, type and sub-type for ICMP, port number for others).
- The LOG macro takes into account the **Excessive Log Grace Period** parameter in the **Control Properties/Logging and Alerting** window, so that only the first packet encountered within the grace period generates a log entry or an alert.

See also "Format Lists" on page 92 and "log" on page 100.

Syntax

```
LOG(format,alert,rule-number);
```

The LOG macro takes three arguments, as follows:

TABLE 3-13 LOG macro arguments

argument	meaning
format	the type of log, e.g. long or short.
alert	the type of alert to be issued. The value LOG_NOALERT is predefined.
rule-number	The number of the rule invoking the tracking.

TRAP

TRAP calls a routine in the VPN-1/FireWall-1 daemon which typically loads a value into a table.

Example

Here is an example of a Security Policy that allows routing updates from authorized routers, and allows outgoing TCP connections.

```
allowed_routers={ 184.145.23.5, 184.145.23.6, 184.154.23.8 };
connections = dynamic refresh expires 180;

//accept outgoing OSPF packets
outbound all@all
   accept ip_p = ospf;

//accept outgoing TCP packets and record in the table
outbound all@all
   accept tcp,
        record <src, sport, dst, dport, ip_p> in connections;

//accept incoming OSPF packets from allowed routers
inbound all@all
   accept <src> in allowed_routers, ip_p = ospf;

//accept incoming TCP reply packets for recorded connections
inbound all@all
   accept tcp, <dst, dport, src, sport, ip_p> in connections;

//drop all other packets
drop;
```

CHAPTER 4

Directories and Files

VPN-1/FireWall-1 directories

TABLE 4-1 VPN-1/FireWall-1 directories

Directory	Description	Described on
`bin`	executable files	page 106
`cisco`	Cisco routers' executable files (Unix only)	page 107
`conf`	GUI configuration and files	page 108
`database`	on FireWalled hosts, holds temporary copy of VPN-1/FireWall-1 database	page 110
`doc`	documentation and help files (Unix only)	page 110
`lib`	VPN-1/FireWall-1 language library files	page 111
`log`	log files	page 113
`man`	man pages (Unix only)	page 113
`modules`	Inspection Code module files	page 114
`state`	state files for hosts	page 114
`tmp`	temporary files (compilations and internal) and pid (process ID number)	page 115
`well`	Wellfleet routers' executable and configuration files (Unix only)	page 115

Note – In Windows, the Install application writes the `DeIsl1.isu` file in the `$FWDIR` directory, for use by the UnInstaller.

bin directory

Note – Executable files have the `.exe` suffix in NT only. For example, `fwinfo` in Unix corresponds to `fwinfo.exe` in NT.

TABLE 4-2 bin directory

File	Description
`alertf.exe`	(NT)
`cpconfig`	VPN-1/FireWall-1 configuration
`cpp`	C pre-processor
`cpsed.exe`	SDE execurable
`display.bat`	NT only
`ela_proxy.exe`	ELA proxy
`elaservice.exe`	ELA proxy
`fw`	command line executable
`fwalert`	executable for alert action
`fwav`	CVP server executable
`fwavstart`	start script for CVP server
`fwavstop`	stop script for CVP server
`fwc`	VPN-1/FireWall-1 compilation script
`fwinfo`	
`fwinfo.pmr`	NT Performance Monitor file
`fwinfo2`	
`fwcisco -> ../cisco/fwcisco`	link to router command line executable
`fwciscoload`	executable for downloading Access List to Cisco router
`fwcmsd.exe`	
`fwcomp`	VPN-1/FireWall-1 language compiler
`fwd`	daemon
`fwell`	executable for Wellfleet routers, using SNMP
`fwinfo`	generate debug information regarding the VPN-1/FireWall-1 configuration
`fwinstall`	software installation and configuration script
`fwlv`	GUI Log Viewer
`fwm`	Management Server for Unix and Windows GUI Clients
`fwsvc.exe`	
`fwsngui`	sample Session Authentication agent executable

TABLE 4-2 bin directory (continued)

File	Description
fwstart	start script: load module, start daemons, and install Inspection Code
fwstop	stop script: kill daemons, unload module
fwui	VPN-1/FireWall-1 GUI
fwuninstall	VPN-1/FireWall-1 software un-installation script
fwuninst	VPN-1/FireWall-1 software un-installation executable (NT)
fwxauth	Pops up an authentication window for use with the x11-verify service, but note that Session Authentication with **Contact Agent At** set to **Destination** (see "Session Authentication" in Chapter 15, "Authentication" of *VPN-1/FireWall-1 Administration Guide*) is recommended for this kind of authentication. (Unix only)
fwxlconf	Address Translation configuration executable
gunzip	GNU uncompression executable
in.aclientd	Client Authentication daemon
in.aftpd	FTP Security Server
in.ahttpd	HTTP Security Server
in.arlogind	RLOGIN Security Server
in.asmtpd	SMTP Security Server
in.atelentd.	TELNET Security Server
in.lhttpd	Security Server
load_agent	Load Measuring Agent ("Load Measuring" on page 586 of *VPN-1/FireWall-1 Administration Guide*)
router_load.exe	
sendmail.exe	
snmp_trap	SNMP trap executable
snmpd	VPN-1/FireWall-1 SNMP daemon
status_alert	status_alert executable
userconv.exe	
VIRSIG.DAT	Cheyenne virus signature file

cisco directory

TABLE 4-3 cisco directory

File	Description
fwciscoload	executable for downloading Access List to Cisco router

conf directory

TABLE 4-4 conf directory

File	Description
`*.C`	configuration file
`*.W`	Rule Base
`auth.C`	
`clients`	
`cpsed_config.conf`	SED configuration file
`cpsed_rulebase.conf`	SED configuration file
`cpsed_opsec.conf`	SED configuration file
`cp.macro`	
`default.W`	default rule-base in VPN-1/FireWall-1 GUI format
`dnsinfo`	DNS configuration file for SecuRemote (see "DNS" on page 153 of *Virtual Private Networking*)
`external.if`	used by the restricted versions of VPN-1/FireWall-1 — specifies the name of the external interface (for example, "le0" or "EPRO1") on which IP addresses should not be counted against the limit
`f2ht-bin-sfxs`	
`f2ht-msgs`	
`fwauth.NDB`	user database — not a text file
`fwauth.NDB7`	
`fwauth.NDBBKP`	user database backup file
`fwauthd.conf`	created during installation process; corresponds to `inetd.conf` (see "Security Server Configuration" in Chapter 11, "Security Servers and Content Security" of *VPN-1/FireWall-1 Administration Guide*). During the installation process, the original telnet and ftp are commented out in `inetd.conf`.
`fwauth.keys`	internal S/Key authentication file
`fwav.conf`	configuration file for Anti Virus CVP server included with VPN-1/FireWall-1
`fwmaddon`	

TABLE 4-4 conf directory (continued)

<table>
<tr><th>File</th><th>Description</th></tr>
<tr><td>fwmusers</td><td>list of VPN-1/FireWall-1 administrators
Each line is in the format:
name encrypted-password permission
where permission is one of the following:
<table>
<tr><th>value</th><th>meaning</th></tr>
<tr><td>40000000</td><td>monitor</td></tr>
<tr><td>00000000</td><td>read</td></tr>
<tr><td>01010101</td><td>read-write</td></tr>
<tr><td>00000100</td><td>user</td></tr>
</table></td></tr>
<tr><td>fwopsec.conf</td><td>OPSEC configuration file. See the VPN-1/FireWall-1 OPSEC documentation for more information.</td></tr>
<tr><td>fwrl.conf</td><td></td></tr>
<tr><td>gui-clients</td><td>A list of IP addresses (or network object names), one per line, from which GUI Clients may attach to the Management Server</td></tr>
<tr><td>serverkeys.*</td><td>internal S/Key authentication files</td></tr>
<tr><td>logviewer.C</td><td>Log Viewer GUI objects and layout file</td></tr>
<tr><td>masters</td><td>A list of IP addresses (or network object names), one per line. When the VPN/FireWall Module starts working, it reads this file to determine where to direct logging. The network objects listed in this file are also those which are allowed to load VPN/FireWall Modules to this machine. See also “Masters File” on page 72 of VPN-1/FireWall-1 Administration Guide.</td></tr>
<tr><td>objects.C</td><td>VPN-1/FireWall-1 GUI objects and layout file</td></tr>
<tr><td>omi.conf</td><td></td></tr>
<tr><td>options.conf</td><td>VPN-1/FireWall-1 product names (for installation)</td></tr>
<tr><td>product.conf</td><td>VPN-1/FireWall-1 installed product and options. You should not modify this file.</td></tr>
<tr><td>rulebases.fws</td><td>combined Rule Bases for Windows GUI</td></tr>
<tr><td>slapd.conf</td><td></td></tr>
<tr><td>snmp.C</td><td>VPN-1/FireWall-1 snmpd configuration file (see Chapter 18, “SNMP and Network Management Tools” of VPN-1/FireWall-1 Administration Guide)</td></tr>
<tr><td>smtp.conf</td><td>SMTP Security Server configuration file</td></tr>
<tr><td>smtp.conf.org</td><td></td></tr>
</table>

TABLE 4-4 conf directory (continued)

File	Description
Standard.W	
trapexec.conf	list of programs VPN-1/FireWall-1 kernel module can run
xlate.conf	Address Translation configuration file

conf/lists directory

This directory contains URL lists.

conf/ahclientd directory

This directory contains HTML files used by the Client Authentication daemon (aclientd).

database directory

This directory is on the FireWalled machine, and its files are part of the Security Policy.

TABLE 4-5 database directory

File	Description
authkeys.C	maintained by the local FireWall daemon
rules.C	Rule Base - authentication rules
fwauth.NDB	user database – not a text file
fwuserauth.NDB	user authentication user database – not a text file
fwd.h	
fwd.hosts	
objects.C	downloaded from Management Module

doc directory

This directory is for Unix only.

TABLE 4-6 doc directory

File	Description
fw.info	GUI VPN-1/FireWall-1 on-line help text
fwlv.info	GUI Log Viewer on-line help text

database/lists directory

This directory contains URL lists.

lib directory

TABLE 4-7 lib directory

File	Description
`*.def`	INSPECT include files
`*.h`	INSPECT include files
`auth.def`	Rule Base header definitions include file (authentication)
`base.def`	VPN-1/FireWall-1 language aliases, routines and macro definitions
`code.def`	header definitions include file
`control.map`	maps access privileges and authentication measures for VPN-1/FireWall-1's control link (see "Distributed Configurations" on page 69 of *VPN-1/FireWall-1 Administration Guide*)
`crypt.def`	encryption header definitions include file
`dcerpc.def`	
`defaultfilter.boot`	default "boot" Security Policy
`defaultfilter.drop`	default "drop" Security Policy
`default.pf`	default Security Policy
`dup.def`	debugging header definitions include file
`eht_set.C`	settings for HTML weeding
`formats.def`	log format header definitions include file
`fwconn.h`	structure of the connections table
`fwctrnm.h`	
`fwctrs.h`	(NT only) strings for NT Performance Monitor
`fwctrs.ini`	(NT only) strings for NT Performance Monitor
`fwf2htbin.gif`	
`fwf2htdir.gif`	
`fwf2htunknown.gif`	
`fwntperf.dll`	(NT only) VPN-1/FireWall-1 Performance Monitor DLL
`fwsnmp.dll`	(NT only) VPN-1/FireWall-1 SNMP agent DLL
`fwui_head.def`	Rule Base header definitions include file
`fwui_trail.def`	Rule Base trailing definitions — last "drop everything" rule
`gps.pro`	Postscript log printint prologue
`init.def`	

TABLE 4-7 lib directory (continued)

File	Description
kertabs.def	kernel table definitions
kerntabs.h	
libsun_av.so	Unix only
local.lg	
setup.C	GUI menus setup file
snmp	SNMP configuration files sub-directory
snmp.def	snmp definition headers
std.def	VPN-1/FireWall-1 command line aliases, routines and macros
table.def	table definitions include file
tcpip.def	VPN-1/FireWall-1 definitions of TCP/IP
traps.def	traps definitions include file
traps.h	traps include file
user.def	site specific INSPECT definitions
wellfleet.C	for Bay Networks routers
xtreme.def	protocol definitions include file

lib/ldap directory

This directory is for NT only.

TABLE 4-8 lib\ldap directory

File	Description
schema.ldif	VPN-1/FireWall-1 LDAP schema

lib/snmp directory

For additional information about the VPN-1/FireWall-1 MIB, see "VPN-1/FireWall-1 MIB Source" on page 600 of *VPN-1/FireWall-1 Administration Guide*.

TABLE 4-9 lib/snmp directory

File	Description
chkpnt.mib	VPN-1/FireWall-1 MIB — contains variable definitions for VPN-1/FireWall-1's SNMP daemon; can be used to incorporate the Check Point MIB into any MIB browser or network management system.
cmsapi32.dll	

TABLE 4-9 lib/snmp directory (continued)

File	Description
mib.txt	VPN-1/FireWall-1 MIB — accessed by the VPN-1/FireWall-1 SNMP daemon (snmpd).
mib.txt2	VPN-1/FireWall-1 MIB — compatible with SNMP managers such as SunNetManager.
wellfleet.mib	from Bay Networks

log directory

TABLE 4-10 log directory

File	Description
	VPN-1/FireWall-1 old Log File; name is date log was switched
*.pid	VPN-1/FireWall-1 processes process id number, used by fwstop and fw kill
aSERVERNAME.log	
fw.*alog	VPN-1/FireWall-1 current Accounting Log File
fw.*alogptr	pointers to fw.*alog
fw.*log	VPN-1/FireWall-1 current Log File
fw.*logptr	pointers to fw.log
fw.logtrack	a list of log files and unique identifying numbers (based on inode or timestamp)
fw.*vlog	VPN-1/FireWall-1 current Active (Live) Connections Log File
fw.*vlogptr	pointers to fw.*vlog
fwd.elg	
cpmgmt.aud	a text file log of VPN-1/FireWall-1 GUI Client events
manage.lock	lock file — This file is created by the Windows GUI Client or by fwm on behalf of a Unix GUI Client and is used to prevent two GUI Clients from simultaneously modifying a Security Policy. It contains the name of the locking process and other identifying information. The file is deleted by the process that created it when that process terminates normally.

In NT only, the files fw.log, fw.alog and fw.vlog are not the real Log Files, but only pointers to the real Log Files (fw.log0, fw.alog0 and fw.vlog0). This mechanism enables Log Files to be purged and renamed while they appear to be open.

man directory

This directory is present in Unix only, and holds the man pages.

modules directory

TABLE 4-11 modules directory

File	Description
`fw.conf`	kernel configuration file (Unix only)
`fw.mkdev`	(Unix only)
`fw.sys`	(NT only) the VPN-1/FireWall-1 driver which is copied to `..\System32\Drivers` (in NT 4.0 this file is copied to two different locations)
`fwmod.*`	kernel modules (Unix only)

state directory

The names of the files in this directory depend on whether the machine is a Master or a FireWalled host. If the machine is a Master, then there is a set of the files listed below for each of the managed hosts. In each set, the file names correspond to the host names.

If the machine is a managed host, then there is only one set of files, and the file names are `hostname.*`.

For example, if a Master named elvis manages hosts lisa and marie, then on elvis there would be two sets of files: `lisa.*` and `marie.*`. On lisa, there is a set of files named `lisa.*`, and on marie there is a set of files named `marie.*`.

TABLE 4-12 state directory

File	Description
`default.bin`	default filter
`fwrlconf`	loading configuration file
`hostname``.ctlver`	the Management Module version that created the current Security Policy
`hostname``.db`	users/encryption database
`hostname``.fc`	last filter code file for host *hostname*
`hostname``.ft`	last filter tables file for host *hostname*
`hostname``.ifs`	last state of “myhost”: filter name and interfaces
`hostname``.lg`	last filter log and alert formats for *hostname*
`hostname``.objects`	network objects database
`hostname``.set`	portions of Rule Base (.W)
`local.arp`	Establishes correspondence between IP addresses and MAC addresses for NT (see “From the Outside” on page 435 of *VPN-1/FireWall-1 Administration Guide* for an example of when this file is needed).

tmp directory

TABLE 4-13 tmp directory

File	Description
`default.fc`	filter code (assembler) compiled from `default.pf`
`default.ft`	tables file derived from `default.pf`
`default.lg`	filter log and alert formats derived from `default.pf`
`fwd.pid`	
`fwm.pid`	
`slapd.pid`	

well directory

This directory is for Unix only.

TABLE 4-14 well directory

File	Description
`fwell -> ../bin/fwell`	see “bin directory” on page 106
`wellfleet.C`	configuration file
`wellfleet.mib`	SNMP MIB describing interaction between VPN-1/FireWall-1 and Bay Networks routers

APPENDIX **A**

Glossary

A

Access Control List (ACL) A sequential list of permit and deny conditions that define the connections permitted to pass through a device, usually a *router. ACL syntax is arcane and specific to individual vendors, and a *security policy based on ACLs is difficult to maintain.

ActiveX A programming environment developed by Microsoft Corporation; a direct competitor to Sun Microsystems' *Java. ActiveX presents a security risk because its executable ActiveX control files run on the client and can be used to gain illicit access to its files.

ActiveX Stripping The ability to prevent *ActiveX programs from being executed on the client by removing all ActiveX programs from HTML pages as they are downloaded.

Address Resolution Protocol (ARP) The *protocol used inside networks to bind high level *IP addresses to low-level physical hardware addresses.

Advanced Encryption Standard (AES) A replacement proposed for *DES by the US Commerce Department's National Institute of Standards and Technology (NIST) in 1997. Each of the candidate algorithms supports key sizes of 128, 192 and 256 bits.

anti-spoofing A method used to protect a network against *IP spoofing attacks by verifying that a packet's source and destination *IP addresses are appropriate to the interface through which the packet passes, for example, that a packet entering the local network from the outside carries an external source IP address.

A simple precaution against IP spoofing attacks is to hide internal IP addresses (using the Network Address Translation feature) so that outside users cannot learn what they are.

anti-virus A mechanism that provides detection, inoculation, logging and alerting capabilities to disarm *viruses on a local disk or in files as they are transferred on the network.

API *see* "Application Programming Interface (API)"

application gateway A *firewall that uses *proxies to provide security.

Historically, application level gateways suited the Internet's common uses and needs. However, as the Internet has become a dynamic environment in which new protocols, services and applications appear almost daily, proxies are no longer able to cope with the diversity of the Internet, or to fulfill the new business needs, high bandwidth and security requirements of networks.

application layer The top network communication layer in a *protocol stack. The application layer is concerned with the semantics of work, such as how to format an e-mail message for display on the screen. A message's routing information is processed by lower layers of the network stack (*see* "layered communication model").

Application Programming Interface (API) A well-defined set of functions, syntax or languages that enable application programs to communicate with one another and exchange data.

ARP *see* "Address Resolution Protocol (ARP)"

Asynchronous Transfer Mode (ATM) A method for dynamically allocating bandwidth using a fixed packet size (called a cell). These cells can carry data, voice, and video at high speeds.

ATM *see* "Asynchronous Transfer Mode (ATM)"

audit In network security, examining and evaluating the relative security of a network.

authentication A method of verifying that an object is really what it appears to be: that a user or a computer is not being impersonated by another user or computer, or that a message received is the same message that was sent (that is has not been tampered with).

Users are authenticated by a challenge-response mechanism: the user is asked to provide information (for example, a *password or *token) presumably known to no one else. Computers may be authenticated in a similar way. In addition, human users can be authenticated by biometric means, such as verifying fingerprints or retinal images.

Authenticating a message verifies its integrity and verifying the sender's identity, usually by means of a *digital signature.

authentication algorithm An algorithm, such as MD5, used to calculate the *digital signature by which a message's integrity is verified.

B

B1, B2 level In the USA, the National Security Agency's rating system for network security. Ratings are certified by the National Computer Security Center. A B1 rating describes a basic level of enterprise-wide Internet security and is equivalent to the European E3 rating (*see* "E3"). A B2 rating describes a much higher level of security typically used to protect military systems.

bridge A device, with two interfaces connecting two networks, that replicates packets appearing on one interface and transmits them on the other interface.

broadcast A message sent to every destination on the network, in contrast to *multicast and *unicast.

C

certificate A *digital signature encrypted with the (for example, *RSA) private key of the *Certificate Authority (CA) who sent the message that includes the certificate, intended to generate confidence in the legitimacy of the public key contained in the message.

The recipient can verify that the message was indeed sent by the CA by computing the message's digital signature, decrypting the transmitted digital signature using the CA's public key (reliably available from an out-of-band source such as a printed directory) and comparing the two. If they are the same, then the message was sent by someone who knows the CA's private key; presumably this can only be the CA.[1]

Certificate Authority (CA) A trusted third party from which information (for example, a person's public key) can be reliably obtained, even over an insecure channel.

For example, if Alice and Bob obtain each other's public keys over an insecure channel such as the Internet, they must be certain that the keys are genuine. Alice cannot simply ask Bob for his public key, because there is the danger that Charlie might intercept Alice's request and send Alice his own key instead. Charlie would then be able to read all of Alice's encrypted messages to Bob.

The CA certifies the information it provides by generating a *certificate. Anyone receiving the information verifies the certificate as proof of the information's validity.

community In SNMP, a community is a logical group of managed devices and NMSs in the same administrative domain.

computationally unfeasible Impossible in practical terms though not theoretically so. For example, it is computationally unfeasible to compute the private part of a *public key pair from the public part, because the only known method — the "brute force" approach of trying all the possibilities one after the other — would take millions of years.

connectionless communication A scheme in which communication occurs outside of any context, that is, replies and requests are not distinguishable. Connectionless communication avoids the overhead inherent in maintaining a connection's context, but at the risk of allowing transmission errors to go undetected. Streaming services usually use connectionless communication protocols such as *UDP, because they must attain high transmission speeds and there is no advantage in sending a retransmitted packet out of sequence.

content security The ability to specify the content of a communication as an element of a security policy, in contrast to defining a security policy on the basis of header information only. Effective content security requires that a firewall understand the internal details of the protocols and services it monitors.

An example of content security is enforcing *anti-virus checking for downloaded files, disallowing emails from or to specified email addresses, or allowing access to Web pages containing certain words only during specified time periods.

Content Vectoring Protocol (CVP) An *OPSEC API that enables integration of third-party content security applications such as antivirus software into VPN-1/FireWall-1. The CVP API has been adopted by a wide variety of security vendors.

---

1. Purists would object to saying "encrypted with the private key" and "decrypted with the public key." The words "encrypted" and "decrypted" are used here in their common senses of hiding and revealing.

D

Data Encryption Standard (DES)
An widely-used *secret key *encryption algorithm endorsed as an official standard by the U.S. government in 1977. To address security concerns resulting from the relatively short (56 bit) key length, triple-DES (encrypting under three different DES keys in succession, believed to be equivalent to doubling the DES key length to 112 bits) is often employed.

data link layer (DLL)
see "layered communication model"

Demilitarized Zone (DMZ)
A computer or a network located outside the trusted or secure network but still protected from the insecure network (Internet). Network administrators often isolate public resources such as HTTP servers in a DMZ so that an intruder who succeeds in breaching security cannot continue on to the internal network.

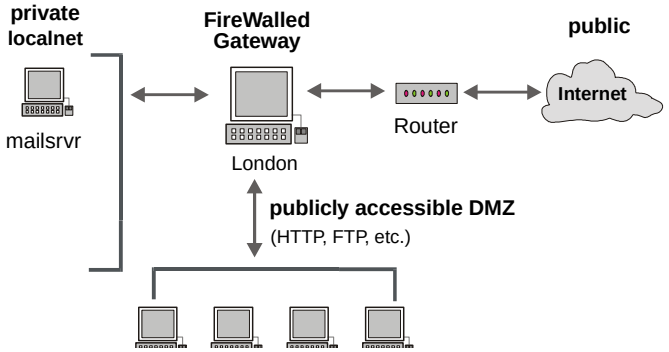


FIGURE A-1 A network with a Demilitarized Zone

In FIGURE A-1, the DMZ is protected by the FireWalled gateway but is at the same time isolated from the private network. There is no way of connecting from the DMZ to the private network without going through the *firewall.

denial of service attack
An attack with the purpose of overwhelming the target with spurious data to the point where it is no longer able to respond to legitimate service requests, in contrast to an attack whose purpose is to penetrate the target system. Examples of denial of service attacks are SYN and "ping of death."

dial-up line
A telecommunication line available only after a dialling procedure, such as an ordinary telephone line, in contrast to a *leased line.

Diffie-Hellman key exchange scheme

A public key scheme, invented by Whitfield Diffie and Martin Hellman, used for sharing a secret key without communicating any secret information, thus avoiding the need for a secure channel. Once the correspondents have computed the shared secret key, they can use it to encrypt communications between them.

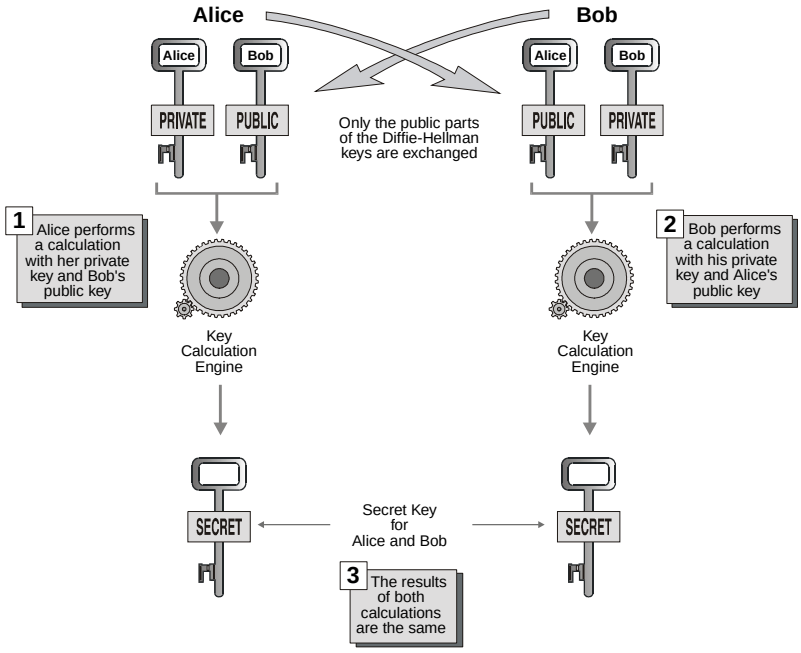


FIGURE A-2 Diffie-Hellman Key Exchange

Under the Diffie-Hellman scheme, each correspondent has a public-private key pair. They agree on a secret key as follows (FIGURE A-2):

- Bob gets Alice's public key (from a *Certificate Authority) and performs a calculation involving his own private key and Alice's public key.
- Alice gets Bob's public key (from a Certificate Authority) and performs a calculation involving her own private key and Bob's public key.

The result of both calculations is the same, and serves as the secret key. In this way, a secret key can be agreed on without any secret information being communicated. There is no opportunity for an eavesdropper to determine the secret key.

An additional advantage of this scheme is that only one key pair needs to be managed for each correspondent.

digital signature

The result of a complex calculation on the contents of a message. Changing even one bit in the message results in a completely different digital signature. Moreover, it is *computationally unfeasible to compose a message with a given digital signature. A digital signature is used to verify a message's integrity, that is, to ensure that it has not been tampered with. *See also* certificate.

directory service A standard database providing distributed, scalable, client/server-based repositories of data that are read much more frequently than modified (for example, user definitions, user profiles, and network resource definitions). Users and applications can access these directories through directory access protocols (DAPs). In network environments, example DAPs include the Novell Directory Services (NDS) and *X.500 directory access protocols. Another widely-used DAP is LDAP (*see* "Lightweight Directory Access Protocol (LDAP)").

DMZ *see* "Demilitarized Zone (DMZ)"

E

E3 A verifiable level of security required by European governments for any Internet firewalls employed over any of its networks. Products meeting this level of security (roughly equivalent to the U.S. B1 "Orange Book" level) are certified by the Information Technology Security Evaluation and Certification organization (ITSEC) in the United Kingdom and by the Logica Evaluation Defence Signals Directorate (DSD) in Australia. *See also* "B1, B2 level".

"E3" also refers to a high speed transmission line in Europe equivalent to the T3 transmission line in the United States.

encapsulated encryption An *encryption scheme in which an entire packet, including the header, is encrypted, and a new header appended to the packet. Encapsulated encryption hides the true source and destination but increases a packet's length, in contrast to *in-place encryption.

encryption The transformation of a message so that the encrypted message can only be read with the aid of some additional information (the *key) known to the sender and the intended recipient alone.

In *secret key (symmetric) encryption, the same key is used to both encrypt a message and then to decrypt it. In *public key (asymmetric) encryption, two mathematically-related keys are used: one to encrypt the message and the other to decrypt it.

encryption algorithm An algorithm, such as *DES, for encrypting and decrypting data. An encryption algorithm is one element of an *encryption scheme.

encryption domain The computers and networks on whose behalf a *gateway encrypts and decrypts communications.

encryption scheme A mechanism for encrypting and authenticating messages as well as managing and distributing keys, such as *FWZ, *IPsec, *SKIP and *IKE.

An encryption scheme consists of three elements:

- an *encryption algorithm that performs the actual encryption
- an *authentication algorithm for ensuring message integrity
- a *key management protocol for generating and exchanging keys

enterprise-wide security management The consistent application and management of a security policy in a complex, distributed network environment, usually including corporate *intranets and *extranets.

extranet In contrast to the Internet, which provides universal access to network-based information, and an *intranet, which is accessible only within an enterprise, an extranet enables a company and its partners or customers to collaborate, communicate and exchange documents in a secured network environment.

extranets typically utilize virtual private networks that allow authorized users to access specific information, such as technical documentation or inventory information (*see* “Virtual Private Network (VPN)”).

F

File Transfer Protocol (FTP) A widely-used TCP-based protocol for copying files between hosts. In security environments, FTP commands can be controlled via *authentication schemes, *content security schemes, file name restrictions, and *anti-virus programs.

firewall A combination of hardware and software resources positioned between the local (trusted) network and the Internet (see FIGURE A-3). The firewall ensures that all communication between an organization’s network and the Internet conform to the organization's security policy. Firewalls track and control communications, deciding whether to pass, reject, encrypt or log communications.

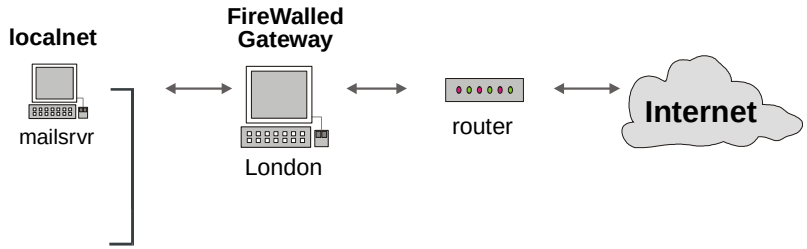


FIGURE A-3 A network protected by a firewalled gateway

FireWall Module A VPN-1/FireWall-1 security application, similar to an *Inspection Module, that provides the additional functionality of *user authentication, *content security, *encryption, *Network Address Translation, and *high availability.

Fortezza A family of security algorithms that ensure data integrity (Secure Hash Algorithm), authentication, non-repudiation (Digital Signature Algorithm), and confidentiality (Key Exchange Algorithm and Skipjack Algorithm). “Fortezza-enabled” and “Fortezza Certified” are terms applied to commercial hardware and software products that use one or more of these Fortezza security algorithms.

frame The packet transmitted by the *data link layer.

FTP *see* “File Transfer Protocol (FTP)”

FWDIR An environment variable specifying the directory in which VPN-1/FireWall-1 is installed.

FWZ Check Point’s domestic and worldwide exportable *encryption scheme, offering *Diffie-Hellman key exchange, multiple *encryption algorithms, *authentication, and *Certificate Authority capabilities.

G

gateway A device positioned between two networks through which all communications between the networks must pass. A gateway is a natural choice for enforcing a security policy and providing encryption and authentication services.

gateway stealthing Disallowing connections that originate or terminate on a *gateway while allowing connections to pass through the gateway, thereby making the gateway transparent (or “invisible”) to the networks which it connects.

H

header The portion of a packet, preceding the actual data, containing source and destination addresses, checksums and other fields. A header is analogous to the envelope of a letter sent by ordinary mail. In order to deliver the message (letter), it is only necessary to act on the information (address) in the header (envelope).

A communication can have several layers of headers. For example, a mail message includes an application layer header specifying, the message originator, date and time. At the lower layers, the packets in which the mail message is transmitted carry IP headers and TCP headers.

high availability A hardware and software configuration in which a device takes over the tasks of another device that has gone down.

host A computer connected to a network.

HTTP *see* "Hypertext Transfer Protocol (HTTP)"

hub A device that connects computers, servers and peripherals together in a local area network (LAN). Hubs typically repeat signals from one computer to the others on the *LAN. Hubs may be passive or intelligent and can be stacked together to form a single managed environment. *See also* "switch" and "router".

Hypertext Transfer Protocol (HTTP) A standard protocol for transferring files on the World Wide Web.

I

IETF *see* "Internet Engineering Task Force (IETF)"

in-place encryption A mechanism by which only the data in an IP packet is encrypted, while the header is not encrypted. In-place encryption leaves headers exposed, but preserves the packet's length, in contrast to *encapsulated encryption.

Information Technology Security Evaluation and Certification Scheme (ITSEC) An organization dedicated to evaluating the security features of information technology products and systems and to certifying the level of assurance that can be placed on them.

INSPECT Check Point's high-level scripting language for defining a *Security Policy. An INSPECT script is compiled into machine code and loaded into an *Inspection Module for execution.

INSPECT Script The ASCII file generated from the *Security Policy by VPN-1/FireWall-1 is known as an Inspection Script. An Inspection Script can also be written using a text editor.

Inspection Code Inspection Code compiled from an Inspection Script and loaded into a VPN-1/FireWall-1 FireWall Module for enforcement.

Inspection Module A VPN-1/FireWall-1 security application embedded in the operating system kernel, between the data link and network layers, that enforces a VPN-1/FireWall-1 *Security Policy. *See also* "FireWall Module".

Internet A public network connecting many thousands of computer networks in a three-level hierarchy including backbone networks (for example, NSFNET, MILNET), mid-level networks and stub networks. The Internet utilizes multiple communication protocols (especially TCP/IP) to create a worldwide communications medium.

Internet Key Exchange (IKE) A standard protocol for authentication and key exchange; part of the key management scheme used for negotiating virtual private networks (VPNs) as defined in the IETF IPSec working group. This key management scheme is mandated for deployment in IPv6. It was formerly known as *ISAKMP.

Internet Engineering Task Force (IETF) The principle body engaged in the development of new Internet standard specifications. IETF identifies solutions to technical problems and makes recommendations to the Internet Engineering Steering Group (IESG) regarding the standardization of protocols and protocol usage in the Internet, and facilitates the transfer of technology developed by the Internet Research Task Force (IRTF) to the wider Internet community. IETF also provides a forum for the exchange of information between vendors, users and researchers interested in improving various aspects of the Internet. The IETF meets three times a year and is comprised entirely of volunteers.

Internet Protocol (IP) The network layer for the TCP/IP protocol suite. IP is a connectionless, best-effort packet switching protocol designed to provide the most efficient delivery of packets across the Internet.

Internet Protocol Security Standard (IPSec) An encryption and authentication scheme supporting multiple encryption and authentication algorithms.

Internet Security Association Key Management Protocol (ISAKMP) A standard protocol for authentication and key exchange that is now known as IKE. *See* "Internet Key Exchange (IKE)".

Internet Service Provider (ISP) A provider of access to the Internet. In some cases, these providers own the network infrastructure, while other lease network capacity from a third party.

intranet An internal private network, managed according to Internet protocols, but accessible only inside the organization.

IP *see* "Internet Protocol (IP)"

IPSec *see* "Internet Protocol Security Standard (IPSec)"

IP address The 32-bit address defined by the Internet Protocol to uniquely identify Internet hosts and servers. A typical IP Address, shown here in conventional IP "dot" notation, consists of the following parts:

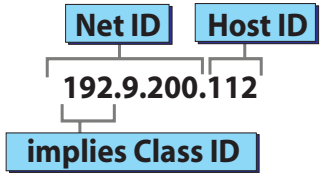


FIGURE A-4 IP Address

The first bits of the Class ID specify a network's class. Most local networks are of class C (Class ID byte = 110XXXXX; Class ID ≥ 192 in IP dot notation). Class C networks can have up to 254 hosts. Larger networks can be either class B or Class A.

The Net ID identifies the network. Because an IP address consists of both a network identifier (NetID) and a host identifier (HostID), it does not identify a host, but rather a network connection (interface). If a host or gateway is connected to several networks, it will have several IP addresses.

By convention, host ID 0 refers to the network itself; that is, a network's address ends in zeros. This scheme enables IP addresses to specify networks as well as hosts. A host identifier of all 1s is reserved for broadcast.

IP spoofing A technique whereby an intruder attempts to gain access by altering a packet's IP address to make it appear as though the packet originated in a part of the network with higher access privileges (for example, the IP address of a workstation in the local network). This form of attack is only possible if a network's internal IP addresses have been exposed (*see* "anti-spoofing").

ISP see "Internet Service Provider (ISP)"

ISAKMP *see* "Internet Security Association Key Management Protocol (ISAKMP)"

ITSEC *see* "Information Technology Security Evaluation and Certification Scheme (ITSEC)"

J

Java A platform-independent programming environment developed by Sun Microsystems and supported by numerous vendors, including Microsoft. Java presents a security risk because Java applets run on the client and can be used to gain illicit access to its files.

Java Stripping The ability to prevent *Java code from being executed on the client by removing all Java tags from HTML pages as they are downloaded.

K

Kerberos An authentication service developed by the Project Athena team at MIT. Kerberos uses secret keys for encryption and authentication. Unlike a public key authentication system, it does not produce digital signatures; Kerberos was designed to authenticate requests for network resources rather than to authenticate authorship of documents. Thus, Kerberos does not provide for third-party verification of documents.

key Information used to encrypt and decrypt data. There are two kinds of keys: *secret keys and *public keys.

key management A mechanism for distributing encryption keys in a public key scheme. Key management is performed by a *Management Station and includes key generation, certification (although this can also be performed by an external *Certificate Authority) and key distribution. Key management can either be manual or automated.

L

LAN *see* "Local Area Network (LAN)"

layered communication model The conceptual division of communication tasks into a "layered model." The fundamental characteristic of the layered model is that each layer processes the same object processed by the corresponding layer at the other end of the communication.

The X.25 protocols shown in FIGURE A-5 are based on the OSI model.

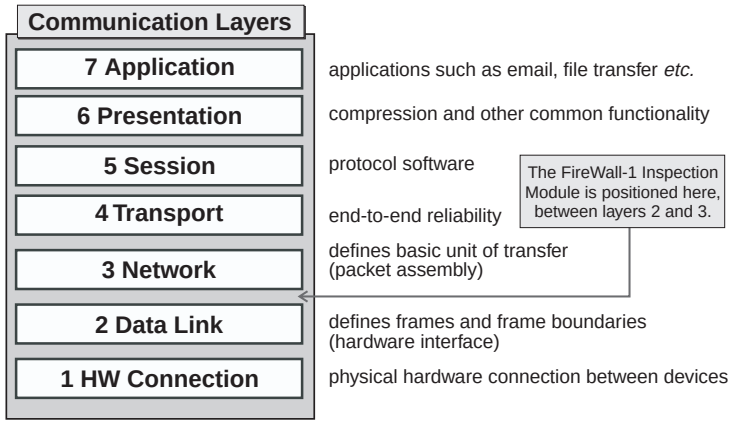


FIGURE A-5 OSI seven layer communication model

The TCP/IP model, consisting of four software layers and one hardware layer, is illustrated in FIGURE A-6.

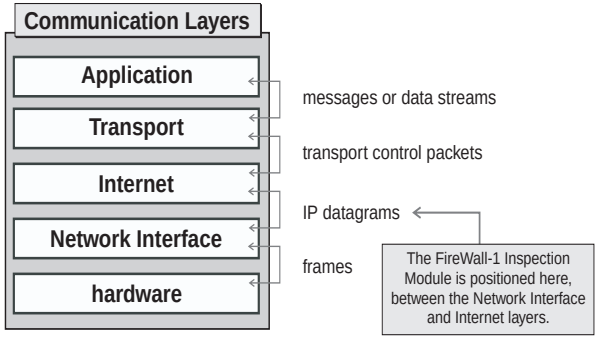


FIGURE A-6 TCP/IP communication model

leased line A dedicated telecommunications access line that is "leased" from a vendor, and thus always available, in contrast to a *dial-up line. The physical medium may be copper or fiber optic, providing a wide range of line speeds.

Lightweight Directory Access Protocol (LDAP) A mechanism for Internet clients to access and manage a database of directory services over a TCP/IP connection. A simplification of the X.500 directory access protocol, LDAP is gaining significant support from major Internet vendors.

load balancing The ability to distribute processing loads among multiple servers to improve performance and reduce access times. Load balancing is often transparent to the user and improves Internet security by reducing the risks associated with certain attacks and by applying greater resources to the task of monitoring and filtering network traffic. A variety of algorithms may be used to determine how best to distribute traffic over these servers.

Local Area Network (LAN) A data network intended to serve an area of only a few square kilometers or less (more typically, an individual organization). LANs consist of software and equipment such as cabling, hubs, switches and routers, enabling communication between computers and the sharing of local resources such as printers, databases, and file and video servers.

Logging and Event API (LEA) An *OPSEC API that enables an application to securely receive and process both real-time and historical logging and auditing events generated by VPN-1/FireWall-1. LEA can be used by a variety of applications to complement firewall management.

M

MAC address The physical hardware address of a device connected to a network.

Managed Internet Security Services Bundled security services, including secure *Internet, *intranet and *extranet, provided by an *ISP. Typically, the ISP handles management and support for the security services, which can be implemented as part of the Internet service implementation or customized to client needs.

Management Module The VPN-1/FireWall-1 module in which a VPN-1/FireWall-1 *Security Policy is defined. *See also* "Management Station".

Management Server The VPN-1/FireWall-1 application, controlled by a GUI on a client, that manages a VPN-1/FireWall-1 *Security Policy. *See also* "Management Station".

Management Station The workstation on which a VPN-1/FireWall-1 *Management Module runs. If the Management Module is deployed in Client/Server mode, then the Graphical User Interface (GUI) can be run on another workstation, while the Management Station runs the *Management Server that supports the GUI.

Manual IPsec *see* "IPSec".

Master In VPN-1/FireWall-1, the station to which logs and alerts are directed.

The Master also maintains the most recent Inspection Code for each of the FireWalled systems it controls. If a FireWalled system loses its Inspection Code for any reason, it can retrieve an up-to-date copy from the Master. In practice, the Master and Management Station are usually on the same system, but Failover Masters can be defined.

multicast A message sent to all the destinations in a specific group of hosts in a network, in contrast to *broadcast and *unicast.

multi-homed host A computer with two or more physical network connections is often referred to as a multi-homed host.

N

NAT *see* "Network Address Translation"

network address The network portion of an IP address. Depending on the class of network; this may comprise the first one to three bytes of an IP address, with the remainder being the host or server address.

netmask For a standard standard Class A, B, or C network, the netmask has no meaning. An explanation of the use of net masks with *nonstandard* network classes follows.

The standard IP addressing scheme can be extended by the use of net masks. For simple, unextended Class C networks, the net mask is 255.255.255.0; that is,

11111111 11111111 11111111 00000000

in binary notation. The 1s in the mask (the first 24 bits) indicate the bits that identify the network and the 0s (last 8 bits) indicate the bits that identify the host. By changing the interpretation of the IP address slightly, it is possible to extend the addressing scheme. If we "borrow" some of the bits from the HostID for the NetID portion of the address, we can extend the IP address to include subnets within one NetID. For instance, the net mask 255.255.255.192 (last byte is 11000000) indicates that 26 bits are being used for the network ID and only 6 bits for the HostID.

Network Address Translation Translating an internal network's real IP addresses to "false" IP addresses, either to prevent exposing the real addresses or to enable hosts with "invalid" addresses to communicate on the Internet, thus avoiding the need to change a network's IP addresses (a formidable, error-prone task).

NIC Network Interface Card; also Network Information Center, an organization that provides services to Internet networks and users.

O

Open Platform for Secure Enterprise Connectivity (OPSEC) An open, industry-wide alliance, driven by Check Point Software Technologies, to ensure interoperability at the policy level between security products. Interoperability is achieved through a combination of published APIs, industry-standard protocols, and a high-level scripting language. OPSEC encourages partnerships in the areas of infrastructure (network products and services), framework (security products), and passport (applications developers).

OPSEC *see* "Open Platform for Secure Enterprise Connectivity (OPSEC)"

overlapping encryption domains Encryption domains overlap when they have at least one host in common.

P

packet A unit of data as sent across a network.

packet filter A type of *firewall that examines only the network layer, typically implemented by *routers. This type of firewall cannot support dynamic protocols and cannot apply application intelligence to the data stream.

password A short string of characters, knowledge of which is required to gain access to some resource. Passwords are considered unreliable security devices because they are relatively easy to guess at, and people tend not to take strict precautions against their disclosure. *See also* "token".

Perfect Forward Secrecy In *IKE encryption, a method of assuring that if an intruder breaks into a system at a given point of time, and gains access to the entire state (all current Phase 1 and Phase 2 keys), he will not be able to decrypt future communications after the next Phase 2 exchange takes place.

PPP (Point-to-Point Protocol) A method for transmitting packets over serial point-to-point links, such as a *dial-up line.

PPTP (Point-to-Point Tunneling Protocol) An extension to PPP that encapsulates different protocols, including IPX and Appletalk, into an IP data stream so that they can be transmitted over the Internet.

protocol A formal description of message formats and the rules required to accomplish some task.

protocol stack A synonym (in practice if not in theory) for the *communication layers as supported by an operating system.

proxy An application-layer implementation of a service that provides additional functionality (for example, security or caching) that is not part of the original service.

Application gateways use proxies to implement firewalls. A proxy's primary advantage is its ability to provide partial communication-derived state, full application-derived state information and partial communication information.

The disadvantages of using proxies as firewalls are:

- **limited connectivity** — each service needs its own proxy, so the number of available services and their scalability are limited, and there is usually a significant delay before a new service can be implemented (a new proxy must be written)
- **limited technology** — application gateways cannot provide proxies for UDP, RPC and other services from common protocol families
- **performance** — application level implementation entails a discernible performance penalty

In addition, proxies are vulnerable to OS and application level bugs, overlook information contained in lower layers, and in the case of traditional proxies, are rarely transparent.

public key A scheme in which each correspondent has a pair of mathematically related keys: a public key known to everyone, and a private key known only to its owner.

- The *RSA public key scheme is used for encryption as follows: if Bob wants to send Alice an encrypted message, he encrypts the message with Alice's public key. The encrypted message can only be decrypted with Alice's private key, which only Alice knows.
- The *Diffie-Hellman public key scheme is used for sharing a secret key without communicating any secret information, thus avoiding the need for a secure channel.

The disadvantage of public key encryption is that it is much slower than *secret key encryption.

The terminology can be confusing, because "public key" is sometimes used to mean both keys together (in the context of schemes) and sometimes to mean only the public part of the key.

Public Key Infrastructure (PKI) A set of security services, usually provided by a *Certificate Authority, enabling *authentication, *encryption and certificate management using *public key encryption technology.

public network Any computer network, such as the Internet, that offers long-distance internetworking using open, publicly accessible telecommunications services, in contrast to a *WAN or *LAN.

R

RC2, RC4 A widely used *encryption method developed by Rivest Corporation for RSA Data Security.

Remote Authentication Dial In Service (RADIUS) A centralized network-authentication scheme developed by Livingston Enterprises and proposed as a standard to the IETF, which includes *authentication, authorization, and accounting features and may also include the ability to pass-through authentication to proxy servers.

Request For Comments (RFC) A numbered series of documents, available from *NIC, which are the primary means of technical discussion about the Internet. Some RFCs define standards.

Resource Reservation Protocol (RSVP) A *unicast and *multicast signaling *protocol, designed to install and maintain reservation state information at each router along the path of a stream of data. RSVP-enabled applications may improve the quality of service across IP networks. Networked multimedia applications, many of which benefit from a predictable end-to-end connection, are likely to be initial users of RSVP-signaled services.

RFC *see* "Request For Comments (RFC)"

Replay Protection A mechanism to prevent an intruder resending legitimate packets. The system detects that the packet was seen in the past in ignores it.

router A device providing network-to-network transmission capabilities, including routing, segmenting and filtering. Most routers support multiple communications protocols, such as ISDN and Ethernet. By examining only packet headers, routers can:

- pass the packets between networks running different protocols
- determine which network should receive the packet
- determine whether to block the transmission

Rule Base An ordered set of rules that defines a VPN-1/FireWall-1 *Security Policy. A rule describes a communication in terms of its source, destination and service, and specifies whether the communication should be accepted or rejected, as well as whether it is to be logged. Each communication is tested against the Rule Base; if it does not match any of the rules, it is dropped.

RSA A public key scheme used for *encryption and *digital signatures, invented in 1977 by Ron Rivest, Adi Shamir and Leonard Adelman; also a company founded by them to market products based on their inventions.

S

SAM see "Suspicious Activity Monitoring Protocol (SAM)"

secret key A symmetric key used to both encrypt and decrypt data.

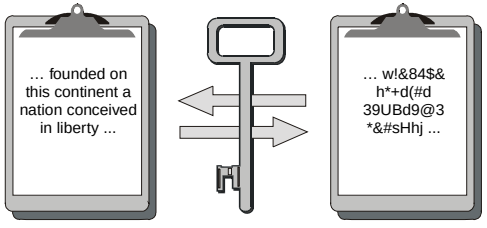


FIGURE A-7 encrypting and decrypting with a secret key

Ensuring the key's secrecy is critical, since anyone who knows the key can decrypt and read the message.

Secret key encryption is simple and fast, but has its disadvantages:

- A secure channel is required by which the correspondents can agree on a key before their first encrypted communication. Direct face-to-face negotiation may be impractical or unfeasible, and the correspondents may have to agree on a key by mail or telephone or some other insecure means.

- The number of keys required can quickly become unmanageable, since there must be a different key for each pair of possible correspondents.

Public (asymmetric) key systems, where each correspondent has a pair of keys, can solve both of these problems (*see* "public key").

Secure Hypertext Transfer Protocol (S-HTTP) A security-enhanced version of *HTTP providing a variety of mechanisms to enable confidentiality, *authentication and integrity. Unlike SSL, which layers security beneath application protocols like HTTP, NNTP, and Telnet, S-HTTP adds message-based security to HTTP. SSL and S-HTTP can co-exist by layering S-HTTP on top of SSL.

SecuRemote Client A software component installed on a desktop or mobile computer that enables secure encrypted communications with an enterprise network.

SecuRemote Server A FireWall Module or VPN Module with which a SecuRemote Client conducts encrypted communications.

Secure Socket Layer (SSL) A protocol combining *RSA *public key encryption and the services of a *Certificate Authority to provide a secure environment for electronic commerce and communications. SSL provides three levels of security server authentication:

- verification of the identity of the server using a *certificate
- *encryption, which ensures the privacy of client-server communications by encrypting the data stream
- integrity, which verifies that the contents of the message arrive at their destination in the same form as they were sent.

Security Policy A Security Policy is defined in terms of firewalls, services, users, and the rules that govern the interactions between them. Once these have been specified, an *Inspection Script is generated and then installed on the firewalled hosts or gateways. These gateways can enforce the Security Policy on a per-user basis, enabling verification not only of the communication's source, destination and service, but the authenticity of the user as well. A user-based Security Policy also allows control based on content. For example, mail to or from certain addresses can be rejected or redirected, access can be denied to specific URLs, and anti-virus checking of transferred files can be performed.

S-HTTP *see* "Secure Hypertext Transfer Protocol (S-HTTP)"

Simple Key Management for Internet Protocols (SKIP) An automated *key management system developed by Sun Microsystems and proposed to the IETF as a standard *IPSec key management scheme. SKIP adds key management functionality to IPSec. Several vendors have successful implementations of SKIP, and both SKIP and *IKE can be deployed/ implemented within the IPSec framework.

Simple Mail Transfer Protocol (SMTP) A *protocol used to transfer electronic mail between computers. Subsequently enhanced to support not only e-mails but file attachments as well, SMTP's flexibility poses a challenge to security systems.

Simple Network Management Protocol (SNMP) A *protocol for managing nodes on an IP network. In security environments, SNMP is used to communicate management information (monitoring, configuration and control) between the network management stations and network elements (for example, devices such as hosts, gateways and servers).

SKIP *see* "Simple Key Management for Internet Protocols (SKIP)"

SMTP *see* "Simple Mail Transfer Protocol (SMTP)"

SNMP *see* "Simple Network Management Protocol (SNMP)"

SSL *see* "Secure Socket Layer (SSL)"

state information Information describing the context of a communication. There are two types of state information: communication derived and application derived.

- Communication-derived state information is extracted from past communications and is compared against current attempts to access or manipulate information. For example, an outgoing PORT command of an *FTP session can be saved so that a later incoming FTP data connection can be verified against it.
- Application-derived state information is extracted from other applications to verify user access. For example, an *extranet application may be used to allow a previously authenticated access through the firewall for authorized services only.

Stateful Inspection A technology developed and patented by Check Point that provides the highest level of security currently available. A stateful *Inspection Module accesses and analyzes all the data derived from all communication layers. This state and context data is stored and updated dynamically, providing virtual session information for tracking connectionless protocols.

Cumulative data from the communication and application states, network configuration and security rules are all used to decide on an appropriate action, either accepting, rejecting or encrypting the communication (FIGURE A-8).

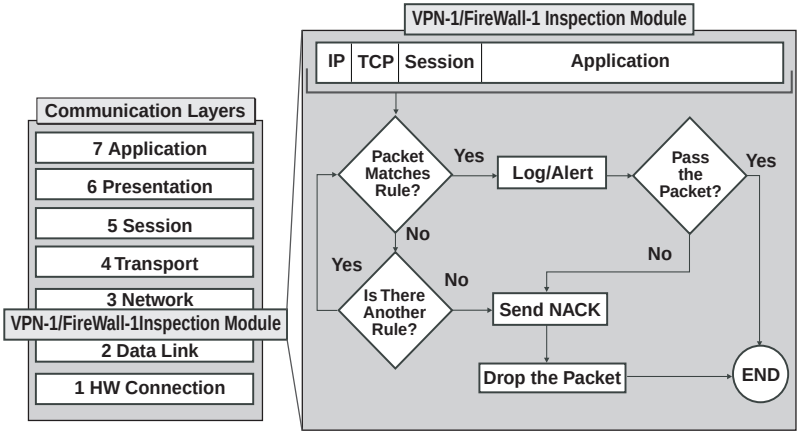


FIGURE A-8 Stateful Inspection

Any traffic not explicitly allowed by the *Security Policy is dropped.

TABLE G-15 Technology Comparison

firewall capability	routers	proxies	Stateful Inspection
communication information	Partial	Partial	Yes
communication-derived state	No	Partial	Yes
application-derived state	No	Yes	Yes
information manipulation	Partial	Yes	Yes

stub network A network that carries only packets to and from local hosts. Even if it has paths to more than one network, a stub network does not carry traffic for other networks. Stub networks are the third and last layer of the Internet network topography.

subnet A physically independent network segment, which shares a network address with other portions of the network. Subnets enable greater security from unauthorized internal access by dividing the intranet into discrete managed portions.

Suspicious Activity Monitoring Protocol (SAM) An *OPSEC API used to integrate third party intrusion detection applications into firewalls.

switch A hub-like device that maximizes the performance of a high-speed connection by providing a dedicated link between two devices via MAC-layer addresses.

symmetric key see "secret key"

T

TELNET (Telecommunications Network Protocol) A remote terminal protocol enabling any terminal to login to another host.

TCP *see* "Transmission Control Protocol"

TCP/IP *see* "Transmission Control Protocol over Internet Protocol (TCP/IP)"

token A *password that can be used only once, typically generated as needed by a hardware device. Tokens are considered to be secure because even if one is revealed, it cannot be misused because it is no longer valid after its first use.

Transmission Control Protocol An connection-oriented and stream-oriented Internet standard transport layer protocol, in contrast to the connectionless UDP protocol ("User Datagram Protocol (UDP)").

Transmission Control Protocol over Internet Protocol (TCP/IP) The common name for the suite of UNIX-based protocols developed by the U.S. Department of Defense in the 1970s. TCP/IP is the primary language of the Internet.

U

UDP *see* "User Datagram Protocol (UDP)"

unicast A message sent to a single destination, in contrast to *broadcast and *multicast.

Uniform Resource Locator (URL) An address format used by Internet communications protocols such as the *Hyper Text Transfer Protocol (HTTP) popularized by the World Wide Web. URLs typically identify the type of service required to access an item, its location on an Internet host and the file name or item name on that machine.

URL *see* "Uniform Resource Locator (URL)"

URL Filtering Protocol (UFP) An *OPSEC API that enables the integration of third-party application to categorize and control access to specific URL addresses.

user authentication The process of verifying that a user is actually who he or she claims to be. *See also* "authentication".

User Datagram Protocol (UDP) An Internet-standard transport layer protocol which adds a level of reliability and multiplexing to IP. UDP is a connectionless protocol, making no distinction between the originator of the request and the response to it.

Connectionless protocols are problematic in a security environment, but can be tracked and controlled using communication-derived state information (*see* "state information").

V

Virtual Private Network (VPN) A network with some public segments in which data passing over its public segments is encrypted to achieve secure communications. A VPN is significantly less expensive and more flexible than a dedicated private network.

virus A program that replicates itself on computer systems by incorporating itself into other programs which are shared among computer systems. Once in the new host, a virus may damage data in the host's memory, display unwanted messages, crash the host or, in some cases, simply lie dormant until a specified event occurs (for example, the turning of a new year).

VPN *see* "Virtual Private Network (VPN)"

W

WAN *see* "Wide Area Network (WAN)"

Web Server A network device that stores and serves up any kind of data file, including text, graphic images, video, or audio. Its stored information can be accessed via the Internet using standard protocols, most often *HTTP.

Wide Area Network (WAN) A (usually private) geographically large network. A WAN is typically constructed to span numerous locations within a single city.

World Wide Web (WWW) A hypertext-based information service providing access to multimedia, complex documents and databases via the Internet. Web application programs can access many other Internet services as well, including Gopher, Usenet news, file transfer, remote connectivity and even special access to data on the local network.

WWW *see* "World Wide Web (WWW)"

X

X.25 A widely-used set of *protocols based on the OSI model. *See also* "layered communication model".

X.500 A *protocol used for communication between a user and an X.500 directory services system. Multiple X.500 directory system agents may be responsible for the directory information for a single organization or organizational unit.

X.509 A certification methodology providing authenticated, encrypted access to private information, which establishes a trust model enabling certain transactions such as those involving money or funds. For example, X.509 certificates are used in the *IKE encryption scheme to obtain public keys and to verify the authenticity of the parties in an exchange.

Index

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X