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

Version 4.1

Part No.: 71300004410
July 1999

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

July 20, 1999

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 4.1?

Summary of Contents

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

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

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% uu` `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`)
`fwd`	`fw d` (space after `fw`)

Setup

cpconfig

Syntax

```
cpconfig
```

Windows NT

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

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).

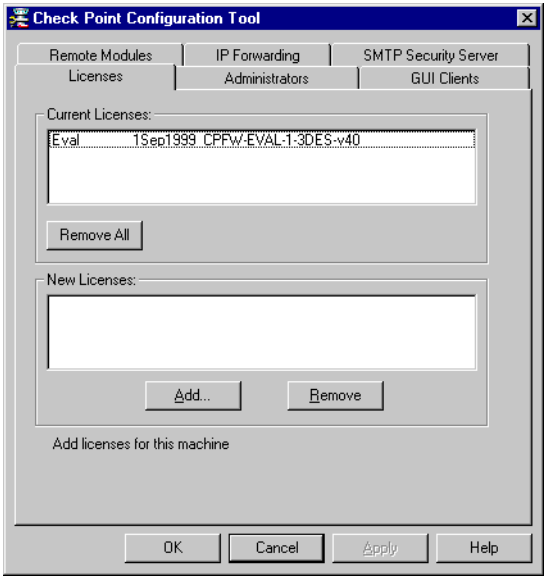

FIGURE 1-1 VPN-1/FireWall-1 Configuration window

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 “Configuring VPN-1/FireWall-1” on page 15 of *VPN-1/FireWall-1 Administration Guide*.

Unix

cpconfig displays the following screen (FIGURE 1-2). 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) Exit

Enter your choice (1-13) :
Thank You...
```

FIGURE 1-2 cpconfig Configuration Options

TABLE 1-2 cpconfig configuration options

option	description	see also ...
licenses	Update VPN-1/FireWall-1 licenses.	"fw putlic" on page 11
administrators	Update the list of administrators, users who are authorized to connect to a Management Server through the GUI.	"Access Control" on page 15" 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 15" of *VPN-1/ FireWall-1 Administration Guide*

TABLE 1-2 cpconfig configuration options(continued)

option	description	see also ...
remote modules	Update the list of remote FireWall and Inspection Modules managed by a Management Module.	"Access Control" on page 15 of *VPN-1/FireWall-1 Administration Guide*
Security Servers	Configure the Security Servers.	"Security Servers" on page 335 of *VPN-1/ FireWall-1 Administration Guide*
SMTP Server	Configure the SMTP Security Server.	"SMTP Security Server Configuration" on page 344" 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 26 of *VPN-1/FireWall-1 Administration Guide*
default filter	Configure the default Security Policy.	"Default Security Policy" on page 304 of *VPN-1/ FireWall-1 Administration Guide*
random pool	Configure RSA keys.	*VPN-1/FireWall-1 Virtual Private Networks*
CA keys	Configure Certificate Authority keys.	*VPN-1/FireWall-1 Virtual Private Networks*

fwstart

`fwstart` loads the VPN/FireWall Module, starts the VPN-1/FireWall-1 daemon (`fwd`), the VPN-1/FireWall-1 SNMP daemon (`snmpd`) and the authentication daemons, and starts `fwm`, the Management Server (see Chapter 1, "Configuring VPN-1/FireWall-1" of *VPN-1/FireWall-1 Administration Guide*).

fwstop

`fwstop` kills the VPN-1/FireWall-1 daemon (`fwd`) and the Management Server (`fwm`), the VPN-1/FireWall-1 SNMP daemon (`snmpd`), and the authentication daemons, and then unloads the FireWall Module.

fw

The `fw` program is used to manage the system. Its specific action is determined by the first command line argument, as described in the following sections. Commands may have a subject (target). There are three options to specify the targets (see "target" on page 6). If more than one is used, the command will execute on the combination of targets.

For each option, it is sufficient to type the first character. Commands and options are described below.

If the first argument is "`-d`" then debug information is generated as `fw` runs.

Control

fw load

`fw load` compiles and installs a Security Policy to the target's FireWall Modules.

Syntax

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

`fw load` compiles and installs an Inspection Script (`*.pf`) file to the FireWall Modules specified by `target`. It converts a Rule Base (`*.W`) file created by the GUI into an Inspection Script (`*.pf`) file and perform the above operations. For more information, see "fw gen" on page 28.

target

If `target` is not specified, the Inspection Code is installed on the local host.

TABLE 1-3 targets

parameter	meaning
`-conf confile`	The command is executed on targets specified in `confile`. Each line in `confile` has the syntax of a target in a target list (see "If target is not specified, the Inspection Code is installed on the local host." on page 6).
`-all`	The command is executed on all targets specified in the default system configuration file (`$FWDIR/conf/sys.conf`).
`targets`	The command is executed on the specific named `target`. See below for an explanation of the syntax of this argument. The dot (`.`) and the at-sign (`@`) are part of the format; spaces around them are not allowed.

Formats

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

Examples

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

Options

parameter	meaning
`interface`	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`. The reference point for the direction is the target host.
`host`	target host is specified using the host's name (the name returned by the `hostname` command) or its IP address.
`all`	has different meaning according to its place. It may specify: both directions, all interfaces or both directions on all interfaces.

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.

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 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.

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 imbedded FireWall Modules. 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.

Syntax

```
fw bload [inspect-file | rule-base] target
```

fw bload compiles and installs an Inspection Script (*.pf) file to the FireWall imbedded system specified by target. It converts a Rule Base (*.W) file created by the GUI into an Inspection Script (*.pf) file and performs the above operations. For more information, see "fw gen" on page 28.

TABLE 1-4 Options

parameter	meaning
rule-base	file containing the rule-base
inspect-file	file containing the compiled version of the rule-base
targets	The command is executed on the specific named target.

fw unload

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

Syntax

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

Examples

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

fw fetch

`fw fetch` fetches the Inspection Code that was last installed on the local host. You must specify the Master where the Inspection Code is found. Use "localhost" in case 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.

Syntax

```
fw fetch  targets
```

Examples

```
fw fetch gateway1
```

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`.

In addition, 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 411 of *VPN-1/FireWall-1 Administration Guide*.

Syntax

```
fw logswitch [-h target] [+|-][""|old_log]
```

Options

parameter	meaning
target	The resolvable name or IP address of the remote machine (running either a FireWall Module or a Management Module) 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 22 of *VPN-1/FireWall-1 Administration Guide*.
+	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 TABLE 1-3 on page 6 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 1-5 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 1-5 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, but 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 *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's name (*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 *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 1 above).

Example

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

```
fw logswitch old.log
```

See also "How can I switch my Log File on a periodic basis?" on page 593 of *VPN-1/FireWall-1 Administration Guide*.

fw putkey

`fw putkey` installs a VPN-1/FireWall-1 authentication password on a host, thus enabling control connections between the host on which the fw putkey command is run and a second host.

Syntax

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

This password is used to authenticate internal communications between VPN/FireWall Modules and between a VPN/FireWall Module and its Management Station/Server. 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 22 of *VPN-1/FireWall-1 Administration Guide*.

The password can be entered on the command line (using the `-p` argument), or interactively. If neither `-opsec` nor `-no_opsec` is specified, then both VPN-1/FireWall-1 and OPSEC connections are enabled.

Options

parameter	meaning
`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"
`-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.

Options (continued)

parameter	meaning
`-k num`	The length of the first S/Key password chain for fwa1 authentication. The default is 7. When less 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 exhausted relatively quickly.
`-n name`	The IP address (in dot notation) to use in identifying this host to the other host, instead of the resolution of the `hostname` command.
`-p password`	The key (password). You will be prompted for this if you do not enter it in the command line.

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 2).

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 17).

Syntax

```
fw putlic [-overwrite]
    [-check-only] [-check-one]  [-f licensefile]
    [-n netmask] [-kernelonly]
    hostname|ip-addr|hostid|"eval"
    licensekey features certificatekey
```

Options

<table>
<tr><th>parameter</th><th>meaning</th></tr>
<tr><td>-overwrite</td><td>overwrite (delete) all existing licenses with the new license</td></tr>
<tr><td>-check-only</td><td>verify the license</td></tr>
<tr><td>-check-one</td><td></td></tr>
<tr><td>-f licensefile</td><td>the name of the file with the license text</td></tr>
<tr><td>-kernelonly</td><td>copy the user level license to the kernel — takes no parameters</td></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><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>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 (beginning and ending with 00)</td></tr>
</table></td></tr>
<tr><td>"eval"</td><td>evaluation license</td></tr>
<tr><td>licensekey</td><td>This is the License Key string you received from the License Distribution Center, for example:
aa6uwknDc-CE6CRtjhv-zipoVWSnm-z98N7Ck3m</td></tr>
<tr><td>features</td><td>This is a string listing the features included in the license, for example:
CPSUITE-EVAL-3DES-v41</td></tr>
<tr><td>certificatekey</td><td>This is 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 “CK0123456789ab”.

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 [targets]
```

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 <old objects.C file> <new objects.C file>
```

fw ca putkey

`fw ca putkey` is used to distribute the Certificate Authority (CA) Key to a FireWalled gateway. For more information see “Generating CA Keys” on page 349 of *VPN-1/ FireWall-1 Administration Guide*.

fw ca genkey

`fw ca genkey` is used to generate the CA Key on a Management Station. For more information see “Generating CA Keys” on page 349 of *VPN-1/FireWall-1 Administration Guide*.

fw certify ssl

`fw certify ssl` is used to generate a CA certificate on a FireWalled gateway. For more information see “Generating CA Keys” on page 349 of *VPN-1/FireWall-1 Administration Guide*.

Monitor

fw stat

`fw stat` displays the status of target hosts in various formats.

Syntax

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

The default format displays the following information for each host: host name, Rule Base (or FireWall Module) file name, date and time loaded, and the interface and direction loaded.

If `target` is not specified, the status of localhost is shown.

Options

parameter	meaning
`-short`	use short format; for each direction and interface, displays: host name, direction, interface, Rule Base file name and loading date. This is the default format.
`-long`	use long format: in addition to short format, displays number of packets in each of the following categories: total, rejected, dropped, accepted, and logged.
`-inactive`	display 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.

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

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

fw log

`fw log` displays the content of Log Files.

Syntax

```
fw log [-f] [-c action] [-l] [-s start time] [-e end time]
    [-b stime etime]][-h hostname] [log-file] [-n]
```

The default Log file is `$FWDIR/log/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.
`-s start time`	Display only events that were logged after time. `time` may be a date, a time, or both. If date is omitted, then today's date is assumed.
`-e end time`	Display only events that were logged before time. time 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.
`-l`	Display the date for each record.
`-h hostname`	Display only log entries sent by the FireWalled machine `hostname`.
`-n`	don't perform DNS resolution of the IP addresses in the Log File (this option significantly speeds up the processing)
`logfile`	Use `logfile` instead of the default Log file.

Examples

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

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]
```

Options

parameter	meaning
`-d delimiter`	output fields will be separated by this character — default is semicolon (`;`)
`-i inputfile`	name of the input Log FIle
`-o outputfile`	name of the output ASCII file
`-r record_chunk_size`	determines how many records should be read (during a single access to the Log File) into the internal buffer for processing
`-n`	don't perform DNS resolution of the IP addresses in the Log File (this option significantly speeds the processing)
`-f`	export forever to the ASCII output file

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

parameter	meaning
`-k`	print the version number in the Kernel Module

fw printlic

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

Syntax

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

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 “CK0123456789ab”.

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.

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

parameter	meaning
`-product <product-name>`	product name for which license information is requested
`-version <product-version>`	product version for which license information is requested
`-kernel`	check the license in the Kernel Module
`-quiet`	no output is printed
`-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

Example

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

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] crtierion <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>
<tr><th>parameter</th><th>meaning</th></tr>
<tr><td>sam_server</td><td>The IP address (in dot format) or the resolvable name of the FireWall that will enforce the command. The default is <code>localhost</code>, the machine on which the <code>fw sam</code> command is executed. See "Configuration Files" on page 20 for more information.</td></tr>
<tr><td>fwm</td><td>Specifies the FireWall Modules on which to enforce the action. Can be the name of a FireWall object, group or one of the following (default is "<code>All</code>"):
<table>
<tr><th>value</th><th>the action will be enforced on ...</th></tr>
<tr><td><code>All</code></td><td>all the FireWalls which are defined as gateways or hosts on the machine on which the <code>fw sam</code> command is executed</td></tr>
<tr><td><code>Gateways</code></td><td>all the FireWalls which are defined as gateways on the machine on which the <code>fw sam</code> command is executed</td></tr>
</table>
See "Configuration Files" on page 20 for more information.</td></tr>
<tr><td>-v</td><td>Verbose crtierion — writes one message (describing whether the command was successful or not) to <code>stderror</code> for each FireWall on which the command is enforced.</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>

Options (continued)

<table>
<tr><th>parameter</th><th>meaning</th></tr>
<tr><td>timeout</td><td>Specifies the time period (in seconds) for which the action will be enforced. The default is forever.</td></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>-D</td><td>Cancel all inhibit (-i and -I) and notify (-n) commands.</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>crtierion</td><td>One of the following:
<table>
<tr><th>value</th><th>match</th></tr>
<tr><td>src</td><td>ip_address of the source IP address</td></tr>
<tr><td>dst</td><td>ip_address of the destination IP address</td></tr>
<tr><td>any</td><td>ip_address of either the source IP address or the destination IP address</td></tr>
<tr><td>srv</td><td>service</td></tr>
</table></td></tr>
<tr><td>ip_address</td><td>IP address (in dot format or resolvable name) to be matched according to crtierion.</td></tr>
<tr><td>source</td><td>source IP address (in dot format or resolvable name)</td></tr>
<tr><td>dest</td><td>destination IP address (in dot format or resolvable name)</td></tr>
<tr><td>dport</td><td>destination port (integer or name, for example, “telnet”)</td></tr>
<tr><td>ip_protocol</td><td>protocol (integer or name, for example, “tcp”)</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 Modules (Servers) and to `0` on VPN/FireWall and Inspection Modules.

- `FireWall`

  When VPN-1/FireWall-1 is installed, this parameter is set to `0` on Management Modules (Servers) and to `1` on FireWall and Inspection Modules. On machines which are both Management Stations and 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 FireWall Module to inhibit connections through other FireWalled Modules, 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 for 10 minutes all connections originating on louvre.

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 `crtierion` does not match `crtierion` in the original command.

The command:

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

cancels the command in the first example.

Utilities

fwciscoload

`fwciscoload` downloads a Security Policy to a Cisco router.

Syntax

If only a password and an enable-password are required, then the syntax is:

```
fwciscoload machine-name  conf-file LoginPassword EnablePassword
```

Options

parameter	meaning
machine-name	router name
conf-file	Security Policy file (must be in $FWDIR/conf)
LoginPassword	login password for the Cisco router
EnablePassword	enable password for the Cisco router

If the Cisco router uses the TACACS protocol, or if a user name is required in addition to the password and enable-password, then the syntax is:

```
fwciscoload machine-name conf-file UserName LoginPassword
    EnableName EnablePassword
```

Options

parameter	meaning
machine-name	router name
conf-file	Security Policy file (must be in $FWDIR/conf)
UserName	user name
LoginPassword	login password for the Cisco router
EnableName	enable name
EnablePassword	enable password for the Cisco router

Each of the last four parameters can be XXX to indicate that it is unneeded, or PROMPT to indicate that the user should be prompted for the parameter. Use PROMPT when you do not want a password to appear on the command line, or if the password is not fixed (for example, with SecurID).

Note – XXX and PROMPT are case–insensitive and cannot be used as either name or password.

Alternatively, you can download the Security Policy using a TELNET-like interactive session. You should use this option when the enable-login is not covered by the above options. In this case, type:

```
fwciscoload machine-name conf-file -t
```

The interactive session will begin. Enter enable mode manually (type `Ctrl-C` to exit `fwciscoload`).

Type `Ctrl-]` to return to `fwciscoload`, which will then download the Security Policy and exit.

Note – If the TACACS authentication connection between the Cisco router and the TACACS server passes through a FireWalled machine, you must enable the connection in the Rule Base.

Example

The command:

```
fwciscoload cis cis.cl XXX 1234 abcd PROMPT
```

downloads the policy file `cis.cl` (in `$FWDIR/conf`) to the router `cis` with the following parameters:

- The login password is `1234`.
- There is no UserName.
- EnableName is `abcd`.
- The user is prompted for a password (for example, a SecurID password).

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 | xipsec
```

Options

parameter	meaning
`ip_forwarding never`	Specify that VPN-1/FireWall-1 does not control (and thus never changes) the status of IP Forwarding.
`ip_forwarding always`	Specify that VPN-1/FireWall-1 controls the status of IP Forwarding as described below.
`ip_forwarding default`	Specify that 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.
`pstat`	Display VPN-1/FireWall-1 internal statistics.
`install`	Install the VPN-1/FireWall-1 kernel.
`uninstall`	Uninstall the VPN-1/FireWall-1 kernel.
`iflist`	Displays the IP interfaces known to the kernel by name and internal number
`arp`	Displays ARP proxy table, which is a mapping of IP and MAC addresses, and utilizes local.arp file.

IP Forwarding

```
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 there is never a time (after VPN-1/FireWall-1 has been started for the first time) that the host is forwarding packets without the VPN/FireWall Module being loaded with a Security Policy.

It is recommended that IP Forwarding be disabled in the kernel. See “Enabling and Disabling IP Forwarding” on page 26 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

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 62 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. The command takes a Rule Base file as an argument and the Inspection Script is printed to the standard output. Rule Base (`*.W`) files are created by the Graphical User Interface, but you may edit them and use this command to generate Inspection Scripts (though this is *not* recommended).

Syntax

```
fw gen <RuleBase_filename>
```

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 [-sig_no] proc-name
```

Options

parameter	meaning
`[-t sig_no] proc-name`	If the file `$FWDIR/log/<proc-name>.pid` exists, send `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 FireWall Modules.

`fwc` takes an Inspection Script (`*.pf`) file as an argument and produces several files: Inspection Code (`*.fc`) file, FireWall Module tables (`*.ft`) file, log format (`*.lg`) file and `*.set`,`*.db` and `*.objects` files. Those files are created in `$FWDIR/tmp`.

fwm

`fwm` is the VPN-1/FireWall-1 Management Server in the Client/Server implementation of the Management Module, 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.

Syntax

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

Options

parameter	meaning
`-a name`	add or update an administrator
`-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)

Options (continued)

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

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

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

parameter	meaning
`load`	loads the Access List to the router
`unload`	unloads the Access List
`-s`	generate summary output
`stat`	show statistics
`-u`	specifies list of interfaces

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.

For example, 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.

Examples

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

The user might wish to define (manually, in `SSSSSSSSS`) two "virtual" routers, `SSSS1` and `SSSS2`, 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

For example, the command:

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

performs the following actions:

- unloads `E21`, `S21`, `S22` (all the interfaces of the real router `SSSS` — this is because the `SSS` parameter was not specified)
- loads `E21` (all the interfaces of the virtual router `SSSS1`)

In practice, specifying E21 in the command line had no effect. All the interfaces were loaded, but as it happens, SSSS1 has only one interface.

Note – p.W is the name of the Rule Base file.

The command:

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

performs the following actions:

- unloads S21 and S22 (all SSSS2 interfaces — this is because the SSS parameter was specified)
- load S21 and S22 (all SSSS2 interfaces)

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.

Syntax

```
fw tab [-a] [-s] [-u | -m number] [-t <table_name>] targets
```

The default format displays for each host: host name and a list of all tables with their elements.

Options

parameter	meaning
-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 number of elements (default is 16).
-t table_name	Display only table_name table.

Examples

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

fwxlconf

fwxlconf is the VPN-1/FireWall-1 Address Translation configuration utility. For information about using fwxlconf, see "Configuring Address Translation — Command Line Interface" on page 449 of *VPN-1/FireWall-1 Administration Guide*.

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).

- host — the name of the host that should receive the trap
- message — the message sent to host.

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

    -v var: an optional object id to bind with the message
     -g generic_trap: One of the values:
        0 coldStart
        1 warmStart
        2 linkDown
        3 linkUp
        4 authenticationFailure
        5 egpNeighborLoss
        6 enterpriseSpecific (default value)
     -s specific_trap: a unique number specifying
    the trap type; valid only if generic trap
    value is enterpriseSpecific (default value is 0)
```

`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 578 of *VPN-1/FireWall-1 Administration Guide*).

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 375 of *VPN-1/FireWall-1 Administration Guide*).

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 156 of *VPN-1/FireWall-1 Virtual Private Networks*.

Syntax

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

Example

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

fw ldapsearch

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

Syntax

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

Options

parameter	meaning
`-A`	Retrieve attribute names only (without values).
`-B`	Do not suppress printing of non-ASCII values
`-D bindDN`	DN to be used for binding to the LDAP Server
`-F separator`	Print `separator` between attribute name and value instead of "=".
`-h host`	LDAP server identified by IP address or resolvable name
`-l timelimit`	Server side time limit (in seconds) for search
`-p portnum`	port number (The default port is the 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, whose name has the same pattern as (for the `fw1color` attribute) `/tmp/ldapsearch-fw1color-a00188`.
`-T timeout`	Client side timeout (in milliseconds) for all operations
`-u`	Show "user friendly" entry names in the output, for example, "`cn=Babs Jensen, users, omi`," instead of "`cn=Babs Jensen,cn=users,cn=omi.`"
`-w password`	password
`-Z`	Encrypt using SSL.
`-z sizelimit`	Server side size limit (in entries) for search

Filter

This is an RFC-1558 compliant LDAP search filer, for example:

```
objectclass=fw1host
```

Attributes

This is a list of attributes to be retrieved. If no attributes are given, all the attributes are retrieved.

Example

```
fw ldapsearch -p 18185 -b cn=omi objectclass=fw1host objectclass
```

parameter	value
options	-p 18185 -b cn=omi
filter	objectclass=fw1host
attributes	objectclass

User Database - Importing and Exporting

Importing a User Database

To import users into the VPN-1/FireWall-1 User Database from an external source, you must create an ASCII (text) file with the required information and import the file into VPN-1/FireWall-1 using the `fw dbimport` utility.

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 1-6 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.

After preparing the import file, execute `fw dbimport` to import the users into the VPN-1/FireWall-1 User Database.

Syntax

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

Options

parameter	meaning
-m	Indicates that 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	`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	Specifies the name of the import file. The default import file is `$FWDIR/conf/user_def_file`.
-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.

Exporting a User Database

You can export your User Database to a file using `fw dbexport`.

The generated file can be in either of two syntaxes:

- the same syntax as the import file for `fw dbimport` (see "Importing a User Database" on page 36)
- LDIF syntax, which can be imported into an LDAP server

Warning – If you use the `-a` parameter (see below) 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.

Exporting a User Database — dbimport syntax

Syntax (dbimport syntax)

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

Options (dbimport syntax)

parameter	meaning
`-g`	Specifies a group of users to be exported; users are not exported.
`-u`	Specifies that only one user (`user`) be exported.
`-d`	Specifies a delimiter different from the default value (";").
`-a`	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`	Specifies the name of the output file. The default output file is `$FWDIR/conf/user_def_file`.

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.

Exporting a User Database — LDIF syntax

Syntax (LDIF syntax)

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

Options (LDIF syntax)

parameter	meaning
`-l`	Create an LDIF format file for importation by an LDAP server.
`-s`	Specifies the branch under which the users are to be added.
`-a`	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`	Specifies the name of the output file. The default output file is `$FWDIR/conf/user_def_file`.
`-k`	provide the isakmp shared secret for LDIF

Example (LDIF syntax)

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

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

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

Notes

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” the login name.

Another issue is that you may wish to import different groups of users into different branches. In this case, you should run `fw dbexport` more than once, 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.

VPN-1 Accelerator Card

By default, the VPN-1 Accelerator Card is enabled in when VPN-1/FireWall-1 starts. You can also enable or disable it manually as well as obtain status.

In This Section

fw accel

Syntax

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

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 Options parameter: -l — meaning: long format

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

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 1-7.

TABLE 1-7 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

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

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

To view FireWall-1 statistics, proceed as follows:

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

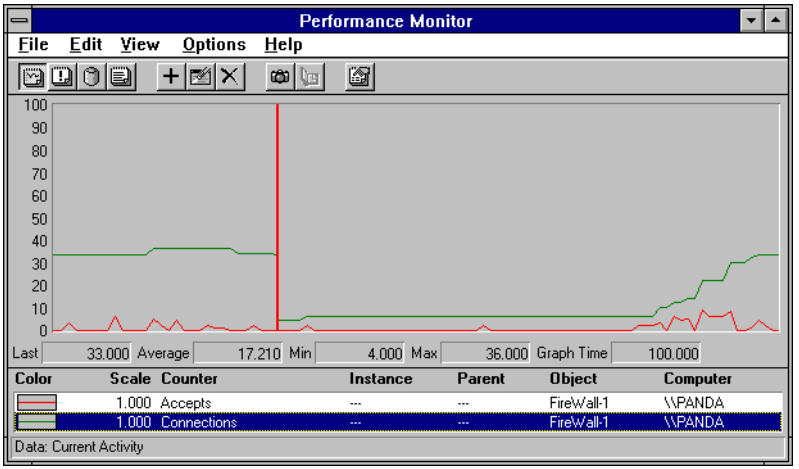

FIGURE 2-1 Performance Monitor window

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

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

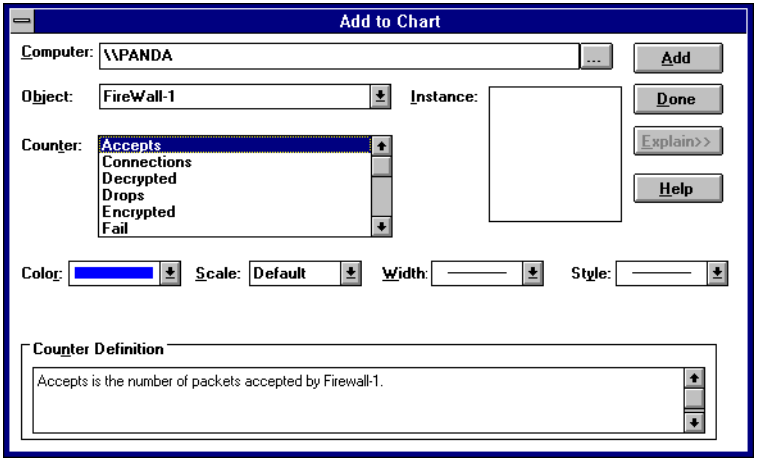

FIGURE 2-2 Add to Chart window

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

The **Counters** listbox shows all the 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 FireWall-1 statistics (see FIGURE 2-1). For the most part, these are the same statistics that are available in the FireWall-1 System Status View.

Windows NT Event Viewer

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.

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

- FW1
- FireWall-1
- FW1SVC

CHAPTER 3

The INSPECT Language

In This Chapter

Introduction

A FireWall-1 Security Policy is defined by a Rule Base and the properties of the objects (networks, services, hosts, and users) used in the Rule Base. Typically, the system administrator defines a Security Policy using the FireWall-1 GUI (Graphical User Interface). From the Security Policy, FireWall-1 generates an Inspection Script written in the FireWall-1 Language (INSPECT). Inspection Scripts are ASCII files and can also be written using a text editor.

Inspection Code, compiled from the Inspection Script, is then transmitted on a secured control channel from the FireWall-1 Management Center — the computer on which the Security Policy was defined — to the FireWall-1 daemons on the network objects that will enforce the policy. The FireWall-1 daemon loads the Inspection Code into the FireWall-1 FireWall Module.

INSPECT was designed specifically as a firewall language, and so it enables typical firewall actions (for example, accept, reject, log, *etc*.). To meet reliability and efficiency requirements, INSPECT has the following characteristics:

- There are no loops.
- Functions do not support recursion.
- Only a limited form of indirect access is allowed.
- Conditions are short circuits.
- There is no explicit memory allocation.
- Function argument passing is by value only.

- A function returns exactly one value.
- Source code is in a single file (except that the C-preprocessor `#include` directive is allowed), and there is no external linkage.
- The name space (that is, macros, functions, tables and formats) begins at the end of its definition and persists to the end of the file.

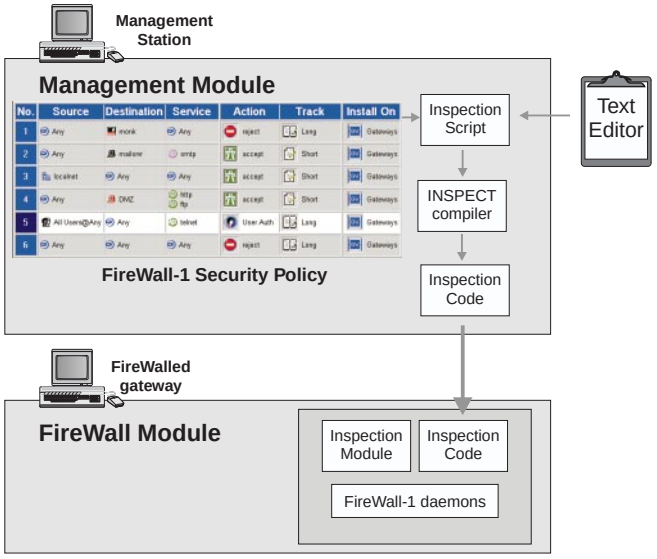

FIGURE 3-1 FireWall Inspection Components - flow of information

The ability to directly edit Inspection Scripts facilitates debugging and enables administrators to tailor Inspection Scripts to their specialized requirements. The rest of this chapter describes the INSPECT language.

TABLE 3-1 FireWall Inspection Components

Component	Description
Inspection Script	ASCII file (`*.pf`) in the INSPECT language which is either generated from a Security Policy (`*.W` file), hand written or some combination of the two
Inspection Code	a file (`*.fc`) compiled from an Inspection Script (`*.pf`)
FireWall Module	a FireWall-1 software module running on a FireWalled host that executes Inspection Code

Writing an Inspection Script

The only way to learn a new language is by writing programs in the language. INSPECT is a firewall language, so testing your program requires that you create the program text, successfully compile it, load it to a FireWalled host and verify that the Inspection Code does what you expect it to do. Once you have mastered the details of these mechanical steps, everything else is relatively straightforward and simple.

A Simple Script

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, the script accepts TCP packets.

Testing the Script

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.

Note – For additional information about the `fwc`, `fwstart` and `fw load` commands, see Chapter 1, "Command Line Interface."

2. Verify that your host is FireWalled.

   Use the `fwstart` command to start the FireWall module.

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

INSPECT Syntax

INSPECT's syntax is similar to that of C, but there are differences between the two languages. The rule statement shown above illustrates some of them:

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

Since INSPECT uses the C preprocessor (see “Preprocessor” on page 77), the script shown above might be rewritten as follows:

```
#define tcp ([ 9 : 1] = 6)
accept tcp;
```

In this script, a pre-processor macro named `tcp` is defined (using the pre-process `#define` directive), and then used in the `accept` statement. This version of the rule statement is simpler and more readable than the first version.

Taking the idea of using macros one step further, the script might again be rewritten, as follows:

```
#define ip_p [ 9 : 1]
define tcp { ip_p = 6 };
accept tcp;
```

In this version, a macro (`ip_p`) representing the byte that specifies the protocol is defined, and then `tcp` is defined in terms of `ip_p`.

Note – `#define` is a C preprocessor directive and `define` is an INSPECT statement. The difference between them is discussed under “define” on page 72.

This two stage definition is useful because it simplifies defining additional protocols, as follows:

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

Compound Conditions

A rule statement’s condition may be more complicated. For example,

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

means: accept the packet if it is both TCP and TELNET. The comma (`,`) is the logical AND operator.

Note – `telnet` and `ftp` are defined in the file `base.def`.

Here is another example:

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

This statement means: accept the packet if it is TCP and either TELNET or FTP.

This statement illustrates the only difference between operator precedence in C and INSPECT. 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 takes precedence over (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.

Here is a rule statement that illustrates the use of parentheses to force operator precedence:

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

This statement would have quite a different meaning without the parentheses.

The next rule statement looks almost like a rule in the Rule Base Editor:

```
accept                                      // Action
 (tcp, telnet or ftp),                      // Services
 ((ip_src = doors) or (ip_src = well)),     // Source
 (ip_dst = natasha);                        // Destination
```

The first four elements of a rule in the Rule Base (Action, Source, Destination and Services) are expressed in the above rule statement.

Note – The definitions of `ip_src` and `ip_dst` are not shown here. INSPECT comments have the same syntax as C++ comments.

Elements of a Rule

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 FireWall Module or Inspection Module that will enforce this rule

You have already seen how the first four elements are expressed in an INSPECT rule statement.

From the point of view of INSPECT's syntax, none of the elements in a rule statement is required. Even a rule statement without an Action can "do something" as a side effect of a condition.

Track

A rule's Track element is often set to one of the log options. Though there is a `log` operator in INSPECT, it's more convenient to use the `LOG` macro, as follows:

```
#include "fwui_head.def"
SRV_tcp(telnet, 23)
SRV_tcp(ftp, 21)
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 `SRV_tcp(telnet, 23)` statement defines `telnet`, and `SRV_tcp(ftp, 21)` defines `ftp`.

For a description of the LOG macro, see "LOG" on page 76.

The `#include` statement in the script shown above includes the standard macro definitions. The script is complete and will compile without errors, if the names `doors`, `well`, and `natasha` can all be resolved. (For more information about `#include`, see "#include" on page 77).

Scope (Install On)

The last element in a rule is Install On, the FireWalled objects that will enforce the rule. This element is known as the rule's scope, and its syntax is:

```
direction interfaces@hosts
```

TABLE 3-2 Scope Elements

element	meaning
direction	`inbound` (or =>) — incoming `outbound` (or <=) — outgoing `eitherbound` (or <>) — incoming and outgoing
interfaces	on which network interface(s) to examine packets
@	required separator
hosts	on which host(s) to examine packets

The scope is specified before the action, as follows:

```
#include "fwui_head.def"
SRV_tcp(telnet, 23)
SRV_tcp(ftp, 21)
inbound all@natasha                    // Install On (scope)
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 shown above specifies the rule's scope as inbound packets on all interfaces of the FireWalled host `natasha`.

`inbound`, `outbound` and `eitherbound` are all macros defined in `fwui_head.def`.

include Files

The `$FWDIR/lib` directory contains a number of files that are always included by Inspection Scripts generated by FireWall-1. You may find it useful to include some of these files in Inspection Scripts you write yourself.

TABLE 3-3 Some Useful include Files

file name	meaning
`fwui_head.def`	contains many useful macro definitions — also includes other `*.def` files
`formats.def`	contains definitions of log formats that are used in FireWall-1 User Interface, for example, the definitions of the Long format and the Short format
`code.def` `base.def`	contain the core logic of the Inspection Module
`fwui_trail.def`	contains the implicit drop rule — usually included at the end of an Inspection Script

INSPECT Reference Manual

Introduction

An INSPECT script is compiled into low level code which is run on a stack-based virtual machine.

The virtual machine is placed in the FireWalled machine's kernel. The virtual machine inspects every IP packet passing through the machine by running the code compiled from the Security Policy.

Since FireWall-1 runs on machines with a 32-bit word length, each stack cell is 32-bits wide. INSPECT uses the stack for storing intermediate values. Other storage areas used by INSPECT are registers and tables (as described later).

INSPECT supports 32-bit integers, as well as other special constants (for example, IP-addresses, interfaces, etc.).

Lexical Conventions

The INSPECT compiler is a single phase compiler. A program consists of a single source file (except for C pre-processor #include file) and is translated in two stages. The first stage is a preprocessor stage, during which the C-preprocessor directives are carried out. In the second stage, the INSPECT Script is transformed into Inspection Code.

INSPECT comments have the same syntax as C++ comments. The characters `//` introduce a comment.

Reserved Words

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

TABLE 3-4 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`

TABLE 3-4 INSPECT Reserved Words

accept	and	call	date
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
- 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, etc, do not hide the system constants (for example, avoid using these names for network objects).

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

Constants

Numeric Constants

There are no floating point constants in INSPECT.

Integer constants can be expressed in the standard formats:

- A number beginning with `0x` is understood as a hexadecimal integer, for example `0x4f`.
- A number beginning with `0` is understood as an octal integer, for example `0777`.
- Otherwise, numbers are understood as decimal integers.

Time Specification

Three decimal integers separated by two colons (for example, `23:30:00`) are understood as a time constant.

Day in Month Specification

Three character abbreviations or full month names, followed by a day number (for example, `Jan 22`) are understood as a time constant.

Day in week Specification

Three character abbreviations or full day names (for example, `sun`) are understood as a time constant.

Special FireWall constants

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

- domains
- nets and hosts (expressed as IP-address constants)
- interfaces

Network and communication entities may be used as special constants in either lists or expressions. Special purpose commands that handle some of these entities are described later.

IP address constant

Four decimal integers separated by three periods (for example, `192.0.0.24`), or three decimal integers separated by two periods (for example `192.0.0`), or two decimal integers separated by one period (for example, `192.0`) are considered constants and are understood as IP addresses.

Identifiers

Types

There are five types of identifiers in INSPECT:

- segment registers
- special purpose registers
- dynamic and static tables
- macros
- functions

Meaning of Identifiers

Identifiers or names refer to variety entities: functions, macros, tables, registers, *etc*. In contrast to function and macro identifiers, tables and registers are storage area designed for the programmer. The compiler is responsible for handling (that is, allocating) these storage areas and the programmer may only store, remove and retrieve data.

Indirect access is limited to storing relative addresses in segment registers (see “Segment Registers” on page 63). The value of a segment register may be interpreted as a relative address in a packet or as a table number (the `set` command may be used to store these values in a segment register). Some of the INSPECT expressions use segment registers for indirect access.

For example,

```
sr6.[12:1]
```

refers to byte (12 + the contents of segment register sr6).

Names

Identifier names are case-sensitive. INSPECT does not impose a limit to a name's length.

Name Resolution

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.

Segment Registers

A segment register is defined as follows:

```
segment-register:
    s[r|R]n
```

where n is a decimal number between 0 and 15. For example, sr6 and sR13 are valid segment register names, but sr16 is not.

Unlike other INSPECT identifiers, segment register names are not case sensitive. So, Sr3, sr3 and sR3 all refer to the same segment register.

A segment register is used to store a base value from which offsets in an expression are calculated. For example, the expression:

```
[12:2]
```

refers to 2 bytes beginning at byte 12 of the packet (the first byte is at byte zero), while:

```
sr6.[12:2]
```

refers to 2 bytes beginning at byte (12 + the contents of segment register sr6).

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

```
set sr1 [12];
accept sr1 = [14,1];
```

This means: accept the packet if byte 12 equals byte 14.

Functions

INSPECT uses functions for two primary reasons:

- to break large computing tasks into smaller ones, clarifying the code
- to decrease the size of the Inspect code

The INSPECT `deffunc` operator is evaluated similarly to `define` but outputs a code rather than identifying inline code.

Consider the following script:

```
#define ip_p [ 9 : 1]
deffunc tcp {ip_p = 6};
accept tcp;
```

The generated code contains a function that checks if the current packet is a TCP packet.

In INSPECT, every function returns exactly one value. The Inspection Code starts by calling the function `tcp` and then applies the `accept` statement on the value returned from the function.

Function parameter passing is strictly by value; so, for example, formats cannot be passed as arguments to a function. No recursion is allowed. Moreover, functions must be defined before they are called and there are no function prototypes.

Tables

There are two types of tables: dynamic (see “Dynamic Tables” on page 66) and static (see “Static Tables” on page 68).

A table consists of attributes and entries.

Attributes

TABLE 3-5 lists the possible attributes of a FireWall-1 table.

TABLE 3-5 FireWall-1 Table Attributes

attribute name	meaning
expires	An entry is removed from the table if it is not accessed in this period of time.
refresh	reset the expiration timeout when accessed
free function *x*	Call this function when an entry is deleted or expires.
hashsize	size of the hash — should be close to table size.
modtrap *x*	Trap the FireWall-1 daemon when the table is modified.
intrap *x*	Trap the FireWall-1 daemon when adding an entry.
outtrap *x*	Trap the FireWall-1 daemon when deleting an entry.
keep	Keep the table’s entries after a Security Policy is installed.
kbuf *x*	The x^{th} argument in the value section is a pointer to an internal data structure (primarily used in encryption).

TABLE 3-5 FireWall-1 Table Attributes (continued)

attribute name	meaning
implies *table_name*	A entry deleted from this table will be deleted from *table_name* table as well.
limit *x*	Limit this table to *x* entries.
synch	Synchronize this table if using FireWall-1 synchronization.

Entries

An entry can be in the form of a tuple, or of a tuple and a value. Tables are associative — that is, an entry is identified by its value rather than by its position in the table.

For example,

```
<12,24,36>
```

is a tuple with 3 elements, while:

```
<12,24;36>
```

is a tuple with 3 elements, of which the last (36) is the value.

A value may itself be a tuple, for example <12,24;36,48>.

The action:

```
record <12,24;36> in udp_tab;
```

puts the 3 element tuple <12,24;36> in the table udp_tab (where 36 is the value and <12,24> is the key.

The expression:

```
udp_tab [12,24]
```

returns the table value (in this case the singleton <36>) whose key is <12,24>.

When the value is a tuple, the get statement is used. The expression:

```
get <12,24,36> from udp_tab to sr1
```

returns each of the values from the <12,24,36> entry in udp_tab into the segment registers starting with sr1, sr2 and so on. For example, if the value is a tuple with three elements, the first is stored in sr1, the second in sr2 and the third in sr3.

A table must be explicitly defined before it is used (see examples below). Only dynamic tables support values.

INSPECT has special operators for testing the content of tables and for extracting table values. The expression:

```
udp_tab [12,24]
```

returns the table value whose key is `<12,24>` and the expression:

```
<12,24> in udp_tab
```

tests if the table contains an entry whose key is `<12,24>`.

A 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. The number of entries (or tuples) in a dynamic table is not fixed when the table is created, and tuples can be freely added and deleted.

To create a dynamic table, define one using the `dynamic` keyword. For example, the expression:

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

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

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

`expires n` — tuples not updated or written for n seconds are deleted

If this attribute is specified for a table, it can be overridden for individual elements (see "Adding an Element to a Dynamic Table" on page 67).

`refresh` — reset the expiration timer on each use (read or write)

`limit` — sets the maximum number of elements the table can contain

`nexpires` — elements do not expire, but are removed only when explicitly deleted

`nexpires` is the default setting.

`keep` — do not reset this table when the Security Policy is re-installed

When a Security Policy is installed on a FireWalled host on which a Security Policy is already installed, 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 (internally assigned by FireWall-1 to each table in accordance with its position in the Inspection Script) are unchanged

`expcall` — call the given (by number) kernel functions when a table element expires (see "call" on page 72)

Defining a Dynamic Table

For example, to define a dynamic table whose tuples expire if they are not used for 60 seconds:

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

Table Operations

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

Adding an Element to a Dynamic Table

To record a tuple in the dynamic table:

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

To record a tuple in the dynamic table with an expiration timer different from the default expiration timer for the table:

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

where `timeout` is the number of seconds.

To modify a tuple in the dynamic table without resetting the expiration timer to zero:

```
record <12,24;36> in udp_tab;
```

Checking if an Element is in a Dynamic Table

To check if a tuple is in a dynamic table:

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

Deleting an Element from a Dynamic Table

To delete a tuple from the table:

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

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;
```

- The context tuple (source, port, and destination) of every outgoing UDP packet (`direction = 1`) is recorded in the dynamic table `udp_out`.
- A context tuple is deleted after 60 seconds if no corresponding UDP packet is encountered.
- An incoming UDP packet (`direction = 0`) is accepted only if its context tuple is in `udp_out` (in other words, if it is 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. The order of the expressions is important here.

For an explanation of `direction`, see "'Current Packet'" on page 71.

Static Tables

A static table is one whose elements are fixed when the table is created, and cannot be added and deleted.

To create a static table, define one using the `static` keyword. A comma-separated list of constant expressions must be specified between the brackets.

For example:

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

Lists

A list is a typed static table. The available types are:

- domains
- nets
- hosts
- interfaces
- format

For example:

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

is a domain list. Having defined the list, 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 };
```

Then 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, the format list (defined in $FWDIR/lib/formats.def):

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

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:

label (for example "`src`")

type (from a list of predefined types):

- `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 FireWall Module)

The first two elements of each tuple are used in creating a log dictionary — a record which describes log records. The last element is the actual value written to the log.

The pre-defined format lists are in the file `$FWDIR/lib/formats.def`.

See also“LOG” on page 76 and “log” on page 74.

Operators

The following operators are available in INSPECT:

TABLE 3-6 FireWall-1 Language Operators

operator	Meaning	operator	Meaning
+	addition	>>	shift right
-	subtraction	<<	shift left
/	division		
*	multiplication	()	function call
%	modular division	[]	table indexing (for example, `udp_tab[12,24]`)
&	bitwise AND	<> and ><	in-out
|	bitwise OR	= and `is`	equal
^	bitwise XOR	!= and `is not`	not equal
,	logical AND	=>	incoming
<	less than	or	logical OR
>	greater than	xor	logical XOR
<=	outgoing or less than or equal to (depending on context)		
>=	greater than or equal to		

Date and Time

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

- `date`— day of the month (0 – 30, where 0 is the first day of the month)
- `day` — day of the week (0 – 6, where 0 is Sunday)
- `tod` — time of day (for example, in the statement `tod < 08:00:00`)

'Current Packet'

Rule statements operate on the current packet, that is, the packet being inspected. Certain operators return information about the current packet, as follows:

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

INSPECT Commands

INSPECT enables you to define functions in an INSPECT script. The differences between functions and macros are:

- 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 macros avoids recursion problems.
- Using functions leads to smaller Inspection Scripts.

For information on defining macros, see "define" on page 72.

For information on defining functions, see "deffunc" on page 72.

accept

The `accept` action accepts a packet. For example:

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

accepts the packet if the condition `(tcp, telnet or ftp)` is `TRUE`. See also the descriptions of the `vanish` and `drop` actions in this section.

A more complex use of `accept` is this:

```
acccept ((condition-list)
or       (TRAP(parameter-list),drop));
```

If the conditions in `condition-list` are all `TRUE`, then the packet is accepted, otherwise the `TRAP` macro is executed and the packet is dropped. This technique (putting the `drop` statement in the `accept` statement) is used when the `TRAP` macro can be expected to modify a table in such a way that when the dropped packet is re-transmitted, the conditions in `condition-list` will all be `TRUE` and the packet will be accepted.

call

The `call` command enables an Inspection Script to call an externally defined function, identified by the first argument. The `call` command is used extensively to support encryption.

The syntax of the `call` command is:

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

deffunc

An INSPECT function is defined in exactly the same way an INSPECT macro (as opposed to a preprocessor macro) is defined, except that `deffunc` is used instead of `define`. The difference is that `define` is expanded inline and `deffunc` is expanded out of line.

Compatibility

FireWall-1 supports the `deffunc` statement starting with Version 2.1, but earlier versions do not. This means that:

- Inspection Scripts generated by FireWall-1 Version 2.1 and higher will not compile under earlier versions.

  To compile your Version 2.1 and higher Inspection Scripts (under Version 2.1 and higher) so that the compiled code will run under earlier versions, add the following pre-processor directive in `$FWDIR/lib/fwui_head.def`:

  ```
  #define deffunc define
  ```

  This directive changes all the function definitions in the script to macro definitions.

- Inspection Code compiled from Inspection Scripts generated by Version 2.1 and higher will not run under earlier versions.

define

The pre-processor `#define` directive removes the `#defined` entity from the compiler's input before compilation. In contrast, the INSPECT `define` operator is evaluated during compilation and assigns a meaning to an entity in a particular context.

Consider the following script:

```
#include "fwui_head.def"
#define ip_p [ 9 : 1]
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.

The compiler understands the `accept` statement because the only meaning of `tcp` known to the compiler when it encounters the statement is appropriate to the context in which `tcp` is used.

Consider the following script:

```
#include "fwui_head.def"
#define ip_p [ 9 : 1]
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.

drop

The `drop` action drops a packet without notifying the sender. For example:

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

drops the packet if the condition `(net_in(ip_src, cp_net_128, cp_net_128_netmask))` is `TRUE`. See also the descriptions of the `vanish` and `accept` actions in this section.

export

The statement:

```
export {} .xxx ;
```

copies everything in the block — between the `{` and the `}` — to the file `<name>.xxx`, where `name` is the name of the file being compiled (without the `.pf` suffix).

For example, the statement:

```
export {} .set
```

in the file `abcdef.pf` copies the block to `abcdef.set`.

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

hold

The `hold` action holds a packet in the kernel. The packet is neither passed not rejected. Its status can only be changed by the FireWall-1 daemon.

in

The `in` operator tests whether a value is in a table. For example, the statement:

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

tests whether the tuple is in the `udp_out` table.

log

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

```
log short;
```

creates a log record in the `short` format. The values of the expressions in the third elements of the format 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

The first two elements in each format tuple are used in creating a log dictionary, that is, a record which describes log records. A single log file may contain many dictionaries, each of which describes a different set of log records in the file.

See also “Format Lists” on page 69 and “LOG” on page 76.

modify

The `modify` command adds a tuple to a dynamic table (as the `record` operator does). If the tuple already exists, the expiration timer is not reset to zero (in contrast to the record operator). This is true even if the table is defined with the refresh attribute.

For example:

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

netof

The `netof` operator tests whether an IP address is a part of a network. For example, the statement:

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

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

set

The `set` command assigns a value to a segment register. For example:

```
set sr6 12;
```

assigns the value `12` to the segment register `sr6`.

record

The `record` command adds a tuple to a dynamic table. If the tuple already exists, the expiration timer is reset to zero (in contrast to the `modify` operator). For example:

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

adds the tuple `<src,sport,dst>` to the dynamic table `udp_out`.

reject

The `reject` action rejects a packet. For 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.

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" on page 307." The `drop` and `reject` actions do perform this mangling for packets of an established TCP connection.

Big Endian and Little Endian

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

Big Endian

Given the following data at these 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

Little Endian

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

When Used

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.

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

Macros

LOG

The `LOG` macro (defined in the file `fwui_head.def`) takes three arguments:

- format — the type of log: `long` or `short`
- alert — the type of alert to issue

  The value `LOG_NOALERT` is predefined.
- rule number — the number of the rule in the Rule Base invoking the tracking

The LOG macro has the following advantages over the log operator:

- The LOG macro automatically handles the rule number.
- The format takes into account the type of packet and writes the appropriate information (program number for RPC, type and sub-type for ICMP and port number for others) to the log.
- LOG takes into account the **Excessive Log Grace Period** parameter in the **Control Properties/ Logging and Alerting** window.

See also "Format Lists" on page 69 and "LOG" on page 76.

TRAP

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

Preprocessor

Pre-processor statements

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

#error

#line

#pragma

The pre-processor directives most commonly used in a INSPECT script are given below:

#define

```
#define XYZ(a,b,c) expression // with parameters
```

or

```
#define XYZ expression  // without parameters
```

#include

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

Conditional Compilation

You can use the pre-processor #ifdef directive to conditionally compile parts of an Inspection Script. The following symbolic constants are defined in each environment:

Compiling and Installing

To compile an Inspection Script (`*.pf` file), use the `fwc` command, which compiles an Inspection Script but does *not* install the resulting Inspection Code (`*.fc` file).

To compile an Inspection Script and install the resulting Inspection Code in a single step, use the `fw load` command.

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

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 80
cisco	Cisco routers' executable files (Unix only)	page 81
conf	GUI configuration and files	page 82
database	on FireWalled hosts, holds temporary copy of VPN-1/FireWall-1 database	page 84
doc	documentation and help files (Unix only)	page 84
lib	VPN-1/FireWall-1 language library files	page 84
log	log files	page 87
man	man pages (Unix only)	page 87
modules	Inspection Code module files	page 87
state	state files for hosts	page 88
tmp	temporary files (compilations and internal) and pid (process ID number)	page 89
well	Wellfleet routers' executable and configuration files (Unix only)	page 89

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

bin directory

TABLE 4-2 bin directory

File	Description
alertf.exe	(NT)
cpconfig	VPN-1/FireWall-1 configuration
cpp	C pre-processor
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
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

TABLE 4-2 bin directory (continued)

File	Description
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 565 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

<table>
<tr><th>File</th><th>Description</th></tr>
<tr><td>*.C</td><td>configuration file</td></tr>
<tr><td>*.W</td><td>Rule Base</td></tr>
<tr><td>auth.C</td><td></td></tr>
<tr><td>clients</td><td></td></tr>
<tr><td>cp.macro</td><td></td></tr>
<tr><td>default.W</td><td>default rule-base in VPN-1/FireWall-1 GUI format</td></tr>
<tr><td>dnsinfo</td><td>DNS configuration file for SecuRemote (see “DNS” on page 156 of Virtual Private Networking)</td></tr>
<tr><td>external.if</td><td>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</td></tr>
<tr><td>f2ht-bin-sfxs</td><td></td></tr>
<tr><td>f2ht-msgs</td><td></td></tr>
<tr><td>fwauth.NDB</td><td>user database — not a text file</td></tr>
<tr><td>fwauth.NDB7</td><td></td></tr>
<tr><td>fwauth.NDBBKP</td><td>user database backup file</td></tr>
<tr><td>fwauthd.conf</td><td>created during installation process; corresponds to <code>inetd.conf</code> (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 <code>inetd.conf</code>.</td></tr>
<tr><td>fwauth.keys</td><td>internal S/Key authentication file</td></tr>
<tr><td>fwav.conf</td><td>configuration file for Anti Virus CVP server included with VPN-1/FireWall-1</td></tr>
<tr><td>fwmaddon</td><td></td></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>
</table>

TABLE 4-4 conf directory (continued)

File	Description
fwopsec.conf	OPSEC configuration file. See the VPN-1/FireWall-1 OPSEC documentation for more information.
fwrl.conf	
gui-clients	A list of IP addresses (or network object names), one per line, from which GUI Clients may attach to the Management Server
serverkeys.*	internal S/Key authentication files
logviewer.C	Log Viewer GUI objects and layout file
masters	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 25 of *VPN-1/FireWall-1 Administration Guide*.
objects.C	VPN-1/FireWall-1 GUI objects and layout file
omi.conf	
options.conf	VPN-1/FireWall-1 product names (for installation)
product.conf	VPN-1/FireWall-1 installed product and options. You should not modify this file.
rulebases.fws	combined Rule Bases for Windows GUI
slapd.conf	
snmp.C	VPN-1/FireWall-1 snmpd configuration file (see Chapter 18, "SNMP and Network Management Tools" of *VPN-1/FireWall-1 Administration Guide*)
smtp.conf	SMTP Security Server configuration file (see FIGURE 2-5 on page 109 of *VPN-1/FireWall-1 Administration Guide*)
smtp.conf.org	
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 22 of *VPN-1/FireWall-1 Administration Guide*)

TABLE 4-7 lib directory (continued)

File	Description
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	
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

TABLE 4-7 lib directory (continued)

File	Description
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 578 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	
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	
fwui.log	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)

TABLE 4-11 modules directory (continued)

File	Description
`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 427 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 80
`wellfleet.C`	configuration file
`wellfleet.mib`	SNMP MIB describing interaction between VPN-1/FireWall-1 and Bay Networks routers

APPENDIX **A**

Glossary

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.

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 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, 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 U.S., 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

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.

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 anti-virus software into FireWall-1. The CVP API has been adopted by a wide variety of security vendors.

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 unsecure 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.

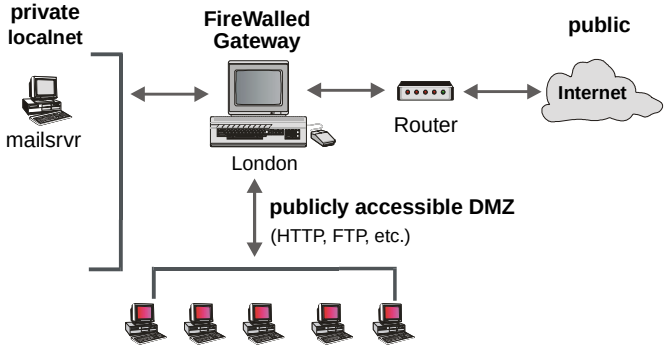

FIGURE A-1 A network with a Demilitarized Zone

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.

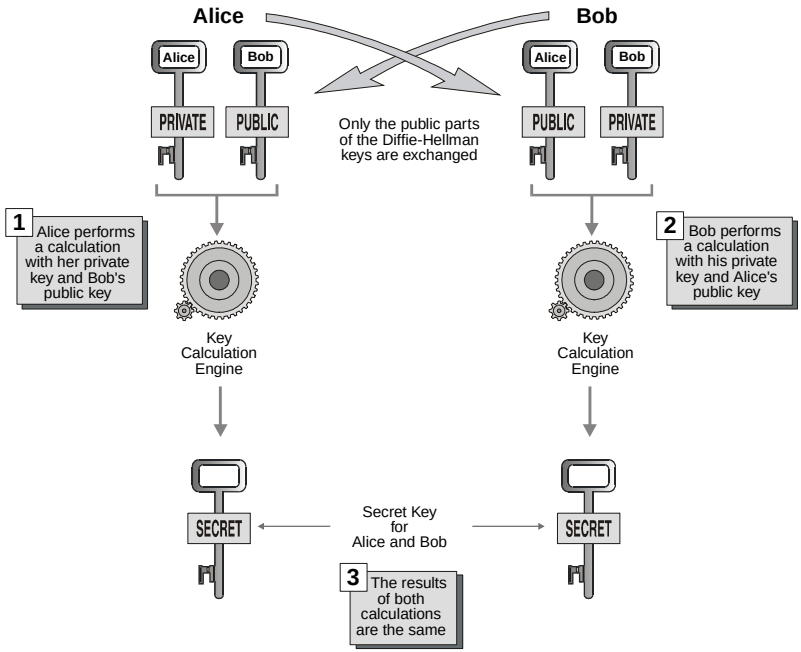

FIGURE A-2 Diffie-Hellman Key Exchange

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 results of both calculations are 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 *ISAKMP.

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.

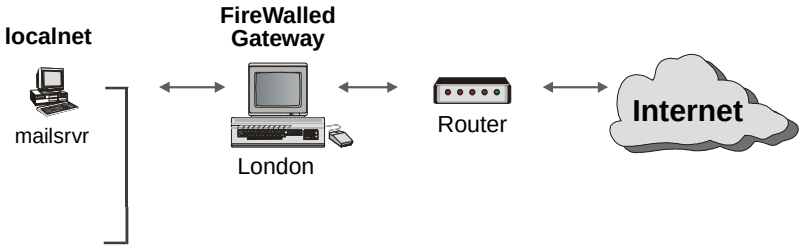

FIGURE A-3 A network protected by a firewalled gateway

FireWall Module A 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 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 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 FireWall-1 FireWall Module for enforcement.

Inspection Module A FireWall-1 security application embedded in the operating system kernel, between the data link and network layers, that enforces a 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) *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:

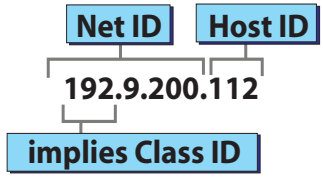

FIGURE A-4 IP Address

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.

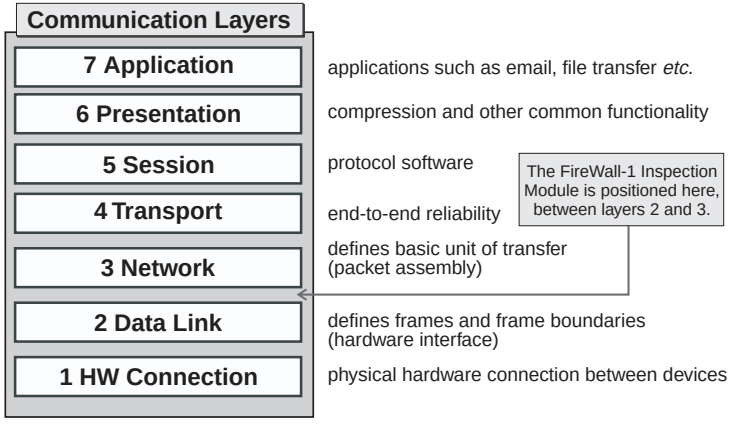

FIGURE A-5 OSI seven layer communication model

The TCP/IP model, consisting of four software layers and one hardware layer, is illustrated in FIGURE A-6.

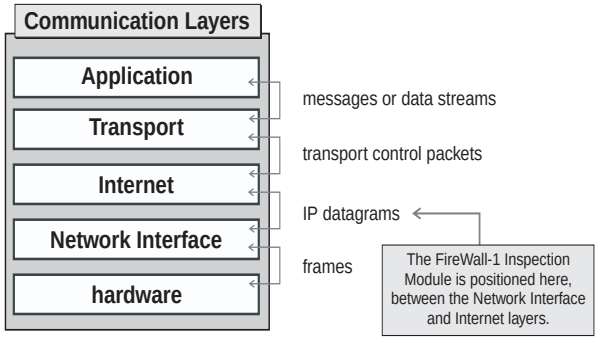

FIGURE A-6 TCP/IP communication model

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 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 FireWall-1 module in which a FireWall-1 *Security Policy is defined. *See also* "Management Station".

Management Server The FireWall-1 application, controlled by a GUI on a client, that manages a FireWall-1 *Security Policy. *See also* "Management Station".

Management Station The workstation on which a 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 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.

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 inter-networking 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 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.

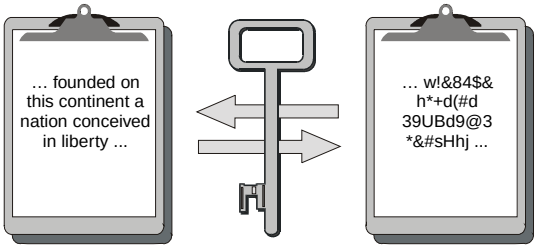

FIGURE A-7 encrypting and decrypting with a secret key

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 *ISAKMP 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).

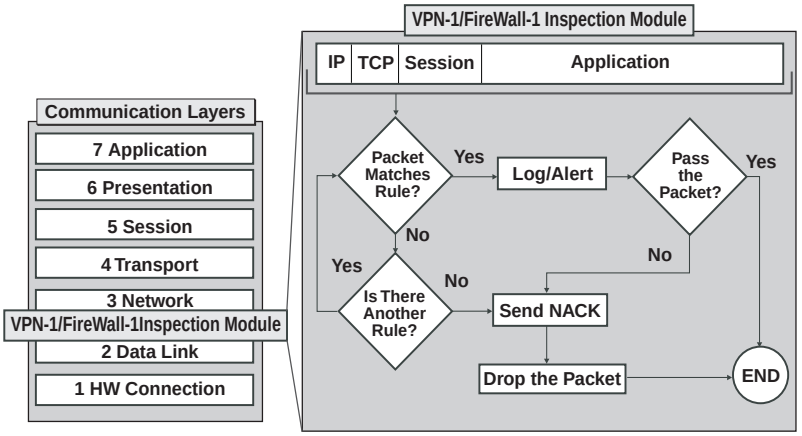

FIGURE A-8 Stateful Inspection

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 *ISAKMP encryption scheme to obtain public keys and to verify the authenticity of the parties in an exchange.

Index

E

F

R

S

T

U

V

W

X