

# Check Point Firewall-1 on Linux, Part Two

David Del Elson 2001-03-05

## Check Point Firewall-1 on Linux, Part Two

by David "Del" Elson

last updated March 5, 2001

---

Check Point Firewall-1 has been the market-leading firewall system since its introduction in 1994. The main advantage of Firewall-1 is its comprehensive and easy to understand GUI, which has made it a firewall system of choice for many corporate IT managers. This is the second in a series of three articles that will examine Check Point Firewall-1 for Linux. The first article consisted of a brief introductory overview of Firewall-1, and a discussion of installation, post-installation tasks, as well as single and multi-system installations. This installment will cover Firewall-1 concepts such as network objects, firewall rules, address translation rules, and NAT, as well as features and limitations of Firewall-1. The final article will then discuss aspects of Firewall-1 such as file and directory layout, rulesets, migrating existing Firewall-1 installation to Linux, and back-up and standby configurations.

### Firewall-1 Concepts

#### Network Objects

A network object is the basic "unit of information" for Firewall-1. The "Network Objects" table is where all of the information about anything with an IP address (or group of IP addresses) is stored.

To manage your network objects, start the Firewall-1 Policy Editor, and choose "Network Objects" from the manage menu. You will need to enter details of the various objects you have on your network, including:

- The firewall itself!
- Servers, both inside and outside of your network, that you want to connect to; and,
- Workstations, inside and outside, that connections are coming from.

For example, you might want to define the following simple set of network objects:

- A firewall;
- The network range inside your network;
- A few servers on a DMZ, for example a web server or FTP server, which you allow public access to; and,
- Some servers on the Internet that you allow / disallow internal users to access.

## Services

Firewall-1 comes with a pre-defined set of services, including most of the well-known services in use on the Internet. It is possible, however, that you may want to include more services that are not predefined. To do this, choose "Services" from the Manage menu in the Firewall-1 Policy Editor.

## Groups

You can group network objects or services together into an easy-to-manage group. For example, you may have a number of web servers in a DMZ to which you want to allow public access. To avoid having to define a large number of rules, one for each server, or to define a rule containing a large list of servers, you could group all of your web servers together into a single group called "Web-Servers". Use this group in a rule, rather than each individual server.

## Firewall Rules

Firewall-1 comes pre-installed with an empty rule set.

Once you have a basic set of network objects and groups defined, you will want use the Policy Editor to begin programming your firewall with a set of rules that define your security policy. Rules can be added to the end of the rule set, inserted at the top or anywhere in the middle. The Firewall-1 policy editor allows you to cut and paste rules, move rules up and down in the rule set, or delete rules at will.

Experimenting with an existing rule set, reading the Check Point "Getting Started Guide" and the Firewall-1 manual (contained in PDF format on the installation CD, in the Docs directory) is the best way to learn how to use the GUI management tool.

## Address Translation Rules

Firewall-1 supports a wide range of Network Address Translation (NAT) rules. The Policy Editor's main tab, "Security Policy", lists accept/deny rules only. The second tab, "Address Translation", is where the address translation rules are shown.

In most cases, you will create address translation rules when defining the network objects. To do this, in the network object properties window, choose the "NAT" tab. There are two basic types of NAT: These are Static and Hide NAT.

## 1. Static NAT

Static NAT is where a single IP address outside your network gets mapped to a single IP address inside your network, in other words a 1:1 mapping. For Static NAT to work, your firewall must have more than one externally visible IP address (more on that under "ARP" shortly), and your ISP must provide routing for all of your externally visible IP addresses.

Static NAT is most useful in cases where you have a server inside your network (or on a DMZ) to which you want to provide access from the Internet. In order for this to happen, you must have an externally visible IP address for that server to use, and hence you must use Static NAT.

To use Static NAT for a network object, you must first ensure that no other network object is using the external IP address that you want to use. Open the network object to which you want to apply NAT (e.g.: an internal server of some kind). In the NAT tab, select the checkbox labelled "Add Automatic Address Translation Rules", choose "Static" as the translation method, and enter the externally visible IP address of the network object in the "Valid IP Address" field. You should now have created a network object that has an internal IP address as well as an externally visible IP address.

## 2. Hide NAT

Hide NAT (or dynamic NAT) is similar to IP Masquerading or other similar functions. This is where a number of network objects, or your entire network, gets hidden behind a single externally visible IP address.

To use Hide NAT, you only need to use a single IP address connected to the Internet (although you can have more.) You can use Hide NAT to hide your network behind the external IP address of your firewall, this is exactly the same as IP Masquerading. To use Hide NAT for a network

object, open the network object that you want to apply NAT to (this could be a workstation, group, or network). In the NAT tab, select the checkbox labelled "Add Automatic Address Translation Rules", choose "Hide" as the translation method, and enter the externally visible IP address that you wish to use in the "Valid IP address" field.

## Editing NAT rules

When you create static or hide NAT rules in a network object, this creates address translation rules within the "Address Translation" tab of the Policy Editor. These rules cannot be edited, but new rules can be added above and below the automatically generated rules.

For example, after setting up NAT for a pair of hidden networks, you may wish to disable NAT for traffic going between those networks. You can do this by inserting an address translation rule at the top of the list, showing that traffic from and to these networks are to be left untranslated. To do this, you might want to first create a group object containing all of your networks (the Address Translation rules can only take a single object in each field of a rule), and leave the "translated packet" source and destination fields reading "= (Original)".

NAT rules can also be used for more complex packet mangling. For example, you might want a host to appear as one IP address when access is requested from the Internet, but a different IP address when it is accessed from a DMZ. (The discussion of why and how to do this is complex and beyond the scope of this article.)

## ARP and routes

Firewall-1 relies on a fairly complex system of ARP and route table entries to assist with static NAT.

In order for static NAT to operate correctly, you must define an ARP rule to make the externally visible IP address of the NAT-ed server appear at the MAC address of your firewall. To do this, you need to know the MAC address of your firewall's external interface (this can be obtained by running "ifconfig" and looking at the "HWaddr" value).

Publishing the ARP entry is done using the /sbin/arp program, as follows:

```
/sbin/arp -s <ip address> xx:yy:zz:aa:bb:cc
```

... where xx:yy:zz:aa:bb:cc is the address that you want to publish, and <MAC address> is the MAC address of the external interface of the firewall (NOT that of the host!).

Surprisingly enough (and confusing even for TCP/IP experts), you also need to create a route, from the external IP address to the internal IP address. For example, if you have a web server which has an internal IP address of 192.168.1.1, and you have set up Static NAT for this server to appear on the internet as 211.11.22.33, then you need to add a route table entry with the following command:

```
/sbin/route add -host 211.11.22.33 gw 192.168.1.1
```

Needless to say, adding all of these arp and route entries can be somewhat boring each time your firewall is restarted, so you had probably best put these in one of your init scripts (e.g.: /etc/rc.d/rc.local).

## FEATURES AND LIMITATIONS

### Features

Firewall-1's main feature, in comparison to other firewall systems, is the relatively easy to use and intuitive GUI.

Firewall-1 supports many types of address translation, that (for example) are not supported by the ipchains module in the Linux 2.2 kernel. For a complex firewall that protects many different types of network devices, this is a must-have feature. This type of address translation is now a feature of the Linux 2.4 kernel with the ipfilter package, and so support for complex address translation is no longer a feature that differentiates Firewall-1 from the Linux kernel.

### Comparison between Firewall-1, ipchains, and netfilter

Ipchains is the firewalling system built into the Linux 2.2 kernel. The ipchains home page is here: <http://netfilter.kernelnotes.org/ipchains/>

Netfilter, the firewalling system built into the Linux 2.4 kernel, is designed as a replacement for ipchains. The home page for netfilter is here: <http://netfilter.kernelnotes.org/>

Note that netfilter is, correctly speaking, a conglomerate of components. At the bottom layer in

the kernel there is "netfilter" which is the system built into the 2.4 kernel for mangling packets. On top of this is a layer for network address translation (NAT), a layer for packet filtering (IPtables), and a user-space tool called "iptables" for managing the entire process.

## Rule chains and trees

Both ipchains and iptables set up a number of "rule chains" in a type of tree structure. At the top of the tree are some pre-installed chains, called input, output, and forward (or for iptables, INPUT, OUTPUT and FORWARD). Each of these chains can branch to another chain for evaluating rules, performing NAT, or other functions (such as logging) based on certain conditions.

A common procedure with ipchains or iptables is to set up a rule chain based on the input and output interface of the packet. The pre-installed chains then branch to these chains based on some rules defined at the top of the chain.

Firewall-1 supports a single chain of rules. Each packet passes the rule chain from the top to the bottom, until it meets a rule that either accepts or rejects the packet.

## Performance Issues

The branching-chain of rules is more efficient than the single chain of rules, because each packet has to pass through fewer rules, on average, before meeting an accept/reject match. Because of this, Firewall-1 is, on a complex rule set, measurably slower on average than using netfilter on the same hardware.

On the other hand, establishing the branching chain is more complex, and more error-prone. Many GUI-based rule-set generators for ipchains do not use the branching-chain mechanisms but instead put all rules into the input, output and/or forward chains as appropriate. This negates most of the performance advantage that would be obtained from not using Firewall-1.

Both Firewall-1 and ipchains / netfilter are implemented as kernel modules. I expect that any measurement of the performance difference between Firewall-1 and iptables using a similar (non-branched) rule set would be very slight. Note that Firewall-1 for Windows NT is not implemented as a kernel module (I expect that this is because Check Point does not have access to Microsoft's source code) and is therefore measurably slower than Firewall-1 for Linux

## Management, Usability, and Support

Generally speaking, I have found Firewall-1 support to be quite good, although somewhat slow. There are many individuals and organisations world-wide that are well trained in Firewall-1 management, and so obtaining staff or an outsourcer to look after a Firewall-1 system should not be a major headache. In comparison, the ipfilter code in Linux is relatively new, and rather complex. I would expect that the learning curve for this would be fairly significant, and finding staff trained in its use would be noticeably more difficult.

To a certain extent, however, a firewall is a firewall, and an experienced security consultant with a good understanding of firewalling concepts should have no difficulty learning either the Firewall-1 or iptables systems. Firewall-1 does appear to have the edge in terms of usability at the moment, however. The Firewall-1 GUI is excellent, and I hope that there will be a port to KDE and/or GNOME in the not too distant future!

## Price

Firewall-1 does not compete with either iptables or ipchains on price. Both ipchains and iptables are free products. Firewall-1 is not-free, and not even particularly cheap.

## Proxy services

Ipchains and netfilter are packet-filtering and packet-mangling modules only. Firewall-1 is not a proxy server system, and does not provide some features found in other packages available on Linux (e.g.: the Squid proxy cache server). It does, however, provide some limited content filtering through proxy "resources" which are defined as Firewall-1 objects.

Because Firewall-1's content filtering features are rather limited, and, in any case, slow down the operation of the firewall ruleset, I am inclined not to use it. Content filtering in HTTP requests is probably best done using a separate proxy server (e.g.: Squid), while SMTP filtering is probably best done using a combination of mail relay system (e.g.: sendmail) and attachment filtering / virus scanning package. There are many of these available on the market, some of which are reasonably priced or free.

Generally speaking, a packet filtering system is much faster than a proxy firewall. Firewall-1 is

built for speed, and is therefore mostly implemented as a packet filtering system only. Although it would be possible to run Squid on the same Linux server as being used for Firewall-1, I would be inclined not to do so, for performance reasons.

## The Next Article...

This installment of our look at Check Point Firewall-1 for Linux has covered Firewall-1 concepts such as network objects, firewall rules, address translation rules, and NAT, as well as features and limitations of Firewall-1. The third and final article will discuss aspects of Firewall-1 such as file and directory layout, rulesets, migrating existing Firewall-1 installation to Linux, and back-up and standby configurations.

To read **Check Point Firewall-1 for Linux, Part Three**, click [here](#).

### Relevant Links

[Check Point Firewall-1 for Linux, Part One](#)

*David "Del" Elson, SecurityFocus.com*

[Check Point Firewall-1](#)

*Check Point*

[Packet Manging for Linux 2.3+](#)

*The Netfilter Project*

[Privacy Statement](#)

Copyright 2006, SecurityFocus