Global Information Assurance Certification Paper

Building an Iptables Firewall with RedHat 8.0

Guideline for building a firewall and hardening the operating system.

The Practical Assignment for GIAC Security Essentials Certification (GSEC)
Version 1.4b

December 2003

By
Graeme Boyle

Table of Contents

Abstract

Firewalls are generally perceived as the first layer in network security in the attempt to provide “Defense in Depth”, as taught in the SANS GIAC training. There are many hardware and software commercial firewall products available, from products that protect a single computer, to products that support large corporate networks. As firewall technology matures, overall prices for firewalls decreases and more vendors offer firewall appliances which are usually lower in price as compared to large corporate solutions. Administrative methods and overall throughput performance may vary, however, all commercial firewalls provide some form of network traffic filtering and monitoring.

On the flip side to commercial products are the shareware, publicly licensed software solutions. Generally, this applies to software developed for the Linux operating system. As with the commercial products, the maturity of firewall products supported under Linux has increased substantially and today, there are numerous free software firewalling products available to the public from various sites on the Internet.

The following paper describes the tools and methodology for hardening an operating system and implementing an IP Tables firewall solution running on an Intel platform with RedHat Linux 8.0. This firewall solution offers both flexibility and security to the user community, as well as a cost effective, easy to administer system.

Assumptions

This document assumes that the reader has some prior Linux/Unix knowledge and is able to download, compile and install software packages as well as build a Linux kernel from the sources. Some understanding of network terminology is required however, there is an abundance of excellent reference material available for reference (see the *references* section).

Background

The corporate firewall is a Shiva LAN Rover VPN Gateway device with two network interfaces. Due to the age of this system, both hardware and software support is no longer available from the vendor. The user community is experiencing an increasing amount of dropped network connections, especially when the support staff connects to remote clients’ networks using various Microsoft Windows based VPN software tools, which is impacting our customer service ability. The system is also susceptible to failure during high network loads and overall reliability has declined. This firewall requires that both incoming and outgoing network definitions are explicitly defined in the rule base which makes the system more complex than is necessary, as well as reducing system performance and slowing down overall Internet access. There is no facility

available to analyze the system logs to see if the company is the target of any attacks or to understand trends and/or overall Internet usage.

The decision was made to replace the current firewall with a modern system that was easier to administer, fast, reliable and supported open standards. While there are many commercial firewall solutions available, the company cannot afford to spend a lot of money on a new system so, a Linux and Iptables based solution was implemented.

Requirements

The company is a system integrator, providing both field service support and Internet support systems to its customers. Specifically, there are two web servers that clients access via the Internet which provide tracking and ordering services to the client base. We also have the usual corporate services, a system providing Microsoft Exchange mail service, DNS and Microsoft Outlook Web support, located inside the corporate network.

Customer service provides support to various customers throughout the United States, utilizing VPN technology between our internal systems and the customers' network being supported. One of the drawbacks with the current firewall is that every connection has to be explicitly defined in the firewall. This requires the IT staff to be constantly available to create these rules, especially on the exception basis. The company's customer service department has frequently failed to perform due to these restrictions.

Firewall logs need to be accessible so that analyze of these logs can be performed, in the hope that we will be able to identify possible risks, attacks or probes, and take the appropriate action proactively. The firewall also needs to be able to track IP sessions so that the amount of required rules are reduced, which will simplify the overall complexity of the firewall as well as provide flexibility to the user community to better service our clients. The firewall has to support NAT (Network Address Translation) as well as proxy services for servers that will be relocated in the DMZ and finally, the system has to be fast!

Assessment

There are two mainstream applications for Linux operating system firewall support, Ipchains and IPTables, which are both supported by RedHat Linux version 8.0. The Ipchains package has been available since kernel version 2.2 whereas IPTables has only been supported from kernel version 2.4. Both of these packages are well tested and there is a large amount of information to be found, both on the Internet and written documentation. My first task was to establish which package would suit my company's requirements. To do this in the shortest amount of time possible, I utilized the Google search engine and searched for "IPTables vs. Ipchains" which produced many articles posted on

Internet News groups as well as information from the Linux.org web site which has a wealth of knowledge regarding Linux in general.

There are many differences between the Ipchains and IPTables packages as demonstrated by the following table, which may be found at http://testweb.oofle.com/iptables/comparison.htm.

	IPTables	IPChains
Match Packets based on:		
Src/Dest Address	*	*
Src/Dest Port	* (Single Port, Port Range, Multiple Port+)	* (Single Port, Port Range++)
TCP Flags	*	*
Input Interface	*	*
Output Interface	*	
Protocol (TCP,UDP, ICMP)	*	*
ICMP Type	*	*
Fragmented Packets	*	
Packet Match Limits	*	
Packet Marking	*	*
Output source†	*	
Connection State††	*	
Type of Service bit	*	*
Packet Destinations:		
ACCEPT (Pass)	*	*
DROP	*	*
REJECT with †††	*	*
ICMP Net Unreachable	*	
Host Unreachable	*	
Port Unreachable	*	
Protocol Unreachable	*	
Network Prohibited	*	
Host Prohibited	*	
Echo Reply	*	
TCP Reset	*	
LOG	*	*††††
on Log Level x	*	
prefixed by text	*	
include TCP sequence	*	

include TCP options	*	
include IP options	*	
MARK	*	*††††
MIRROR•	*	
REDIRECT	*	*
Many other targets not available in ipchains	*	

+ = Multiple Ports: A set of up to 15 nonconsecutive ports (i.e. 1,3,5)
++ = Port Range: A set of consecutive ports (i.e. 1 to 10)
† = Valid only on the output chain from the machine running iptables.
†† = States: Established, Related, New, and Invalid
††† = Valid Rejection types in Iptables: Network Unreachable, Host Unreachable, Port Unreachable, Protocol Unreachable, Network Prohibited, Host Prohibited, Echo Reply, TCP Reset
†††† = Available as a command line switch, not as a packet target.
• = Experimental - inverts source and destination in packet and retransmits.

I chose to implement my firewall with IPTables for the following reasons:

- Development for IPTables is current
- Stateful application level and protocol level network traffic inspection for ICMP, UDP and TCP traffic
- There are a lot new application modules that are inherently supported
- Support for port forwarding for internal and external traffic
- Full policy based routing
- Source MAC address filtering
- Denial of Service (DOS) packet rate limiting
- Variable levels of logging

The Netfilter web site is the project web site for IPTables and includes a number of FAQ's and links to various sites offering tutorials on Iptables. The tutorial that I selected is the one written by Andreasson, Oskar IPTables Tutorial 1.1.19, Version 1.1.19, dated 5/18/2003 and may be found at http://iptables-tutorial.frozentux.net/.

After working through this tutorial for a few days, it was apparent that the complexity of writing IPTables rules will make ongoing firewall maintenance and rule changes difficult, unless you are writing rules on a very regular basis. Something else I considered is that there are no official training courses that I know of or could find, for Iptables. This makes knowledge transfer to co-workers an extra challenge. It was very clear that I would need to search for a front-end interface to IPTables that would make training and ongoing support and maintenance easier and therefore less prone to mistakes.

One of my favorite sites to search for software is http://sourceforge.net. I connected to this site and entered “firewall” as my search criteria. This produced a plethora of projects and software available for download. I decided that I would scan through the project pages and download a number of packages for testing based on the maturity of the software and the current level of development. In this way, I could decide which package would suite my environment. After reading and accessing many project sites within Sourceforge, I downloaded the following packages: Firewall Builder, jwall, firehol, Shorewall, Kmyfirewall and Bastille Linux.

Before installing any of the above packages, I drew the network topology that I was going to implement which assisted me in evaluating the different software packages in a consistent manner. My evaluation criterion included the following features:

- support for Network Address Translation on outgoing traffic
- multiple system support in a DMZ
- proxy address support
- secure access from the Internet to servers located in the LAN
- ability to restrict access by source and destination ports and IP Addresses

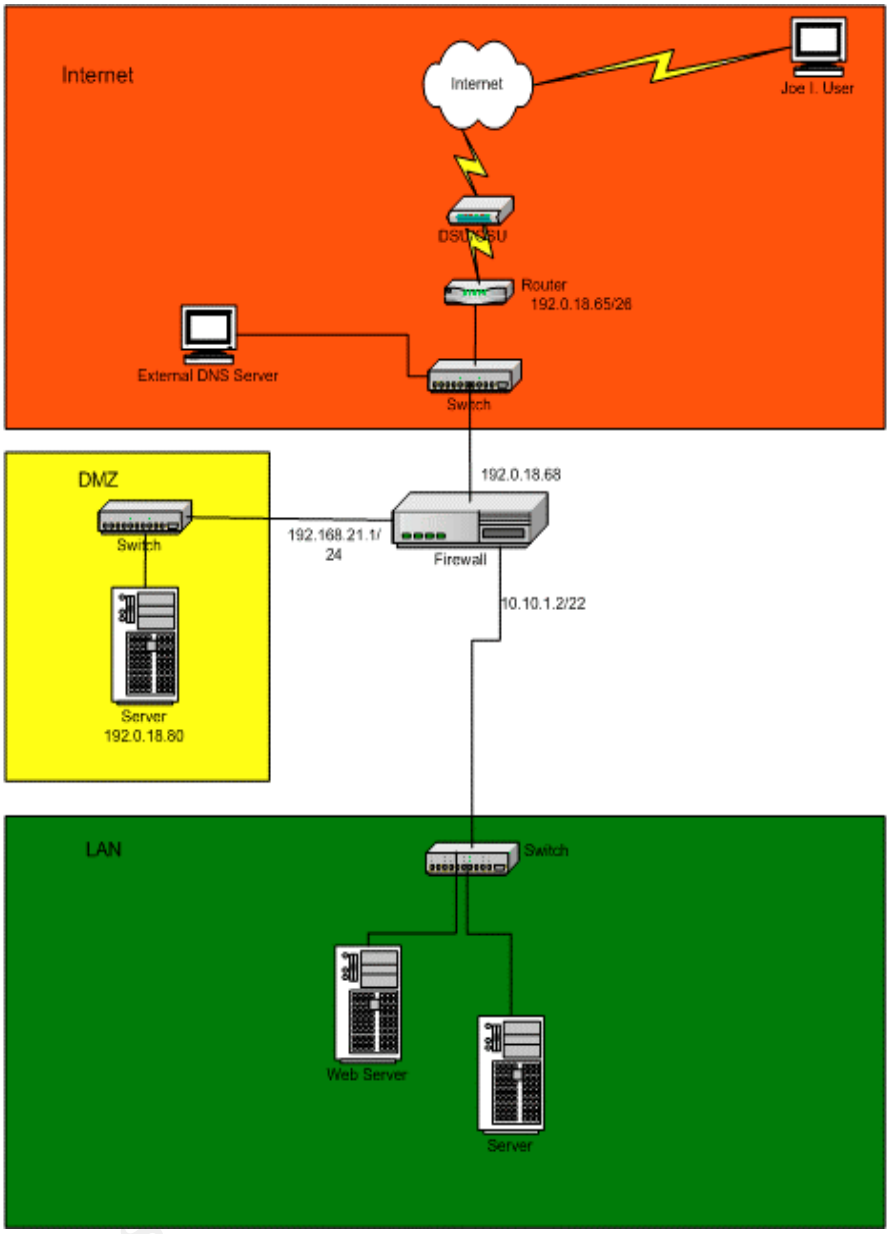

Figure 1.1

Testing Phase

I built a Dell Pentium III, 400 MHz system with three interfaces and 768Mb Ram, and loaded RedHat 8.0 using the custom installation mode which allows for individual selection of software components to be installed. I confirmed that the kernel source software was included in my installation. After installing the operating system, I ran the “up2date” facility to make sure that I had the latest software updates and security fixes for the system. The RedHat operating system comes with a very nice command called “*chkconfig*”. Using this command, I disabled all the services that I did not want to run including, telnet, ftp, nfs, kudzu, Sendmail, autofs since this is going to be a firewall and the only

access that I allow is via the SSH protocol. Once I had disabled all the unnecessary services, I rebooted the system to confirm that the system was still functional and I had not disabled any program that I would need and the system would boot without user intervention.

The next step was to rebuild the system kernel to make sure that support for the Net Filter options was included as without these options enabled, IPTables will not work. RedHat Linux source files are located in the directory /usr/src/linux2.4. To build a kernel, you need to have a valid configuration file which reflects the hardware and software options that you are enabling on your system. Kernel building has to be performed as the root user as you are going to be updating system files. Once you have logged in, if your login session does not automatically start an X-Windows session, type the command "startx" at the prompt. Open a shell window and change to the kernel source directory. Now type the command "*make xconfig*". This will bring up a GUI window with all the sections of the kernel listed. I strongly suggest that you read each component and sub-component to make sure you (a) remove unwanted hardware support and (b) add the networking support that you will need as this will make your kernel more efficient. Since each kernel is system dependant, it does not make sense for me to list all the options however, the Net Filter options that should be include are listed in Appendix C.

Once you have completed editing the configuration, click on the "Save and Exit" button. This will write a ".config" file in the kernel source directory and prompt you to run the "*make dep*" command. After successfully running this command, you need to build the boot image as well as the kernel loadable modules. To complete these tasks and install the kernel and modules in the correct directories, run the following commands in order: "*make bzImage*"," *make modules*", "*make modules_install*" and finally "*make install*". Depending on the system boot loader you are using, check the configuration file to make sure that the correct updates have been inserted and the system will boot to your newly created kernel. For the lilo system, this can be found in the /etc/lilo.conf file and for the GRUB loader, the configuration file is /boot/grub/grub.conf. For more information about how to compile, build and install a new Kernel, see http://www.tldp.org/HOWTO/Kernel-HOWTO/index.html.

Now that I had a working system and newly built kernel with Net Filter support, I installed each firewall package individually, making sure that the package compiled and installed correctly and was completely operational. During this process, I found that the GUI packages are generally harder to install and configure than the command line based packages mainly due to the variety of GUI libraries used, as well as, in certain cases, dependency on specific versions of these libraries

Using a combination of two hubs and laptops, I was able to create a simulation of the proposed network topology, connecting one laptop to the DMZ interface and

the other on the LAN interface, and accessing the internal LAN servers to simulate the internet. I found that, generally, the GUI programs' logic was difficult to understand in the way they created Net Filter rules and how they displayed the rules. Overall, I found this process to be extremely time consuming as I needed to learn each package to be able to test its functionality.

After a few days of installing and removing packages, I decided on the Shorewall package, written by Tom Eastep. My decision was based on the following:

- ease of use
- fast learning curve
- the package builds IPTables rules not a replacement
- is not GUI based
- has excellent documentation and examples within the configuration files
- informative web site
- excellent support through the Shorewall mailing list
- portable code to other systems for backup

My next step was to extend the test environment to work in parallel with the current firewall so that I could test "real world" examples and widen the test environment to a few selected users. This proved to be a valuable training and testing tool.

During my out-bound testing, I set up systems on the LAN to utilize the firewall, which worked as expected. When testing NAT connections from the internet to internal systems, the network connections would continuously fail. After a considerable amount of frustration, head scratching and requests for help on the Shorewall email list, I changed the default route on the internal system that I was testing access to, to point to the new test firewall as its default gateway and then everything worked. While this did slow down my deployment, I was able to gain valuable knowledge and advice on my deployment and on a positive note, I did read the manual.

Building the Final System

After running the test environment for a week with three to four users, the test system proved to be fast and stable. During my testing phase, I had purchased new hardware, a dual Intel 2.8GHz Xeon system with three 10/100/1000 Network interfaces, 2Gb Ram, a 3ware EIDE RAID controller and two EIDE 40Gb hard drives in a RAID 0+1 configuration.

I followed the same procedure as with the test system to load RedHat 8.0 onto the system, disabling unnecessary processes and updating the system to minimize any operating system security risks and building a new kernel. I installed the Shorewall software and started to configure the new system while using the test system files as a reference. Once I had completed the Shorewall configuration I connected the firewall into the network in place of the test system

but, still in parallel to the original firewall. This allowed me to bring the system on-line and test without disruption to the user community.

Once this configuration and testing phase was completed, I started to configure the firewall to provide service to the internal systems which were being accessed from the Internet. As each system was configured into the Shorewall firewall, I removed the corresponding rules from the Shiva firewall. Since servers have fixed IP addresses in my environment, I was able to change the default route in network settings, without rebooting the server and therefore no noticeable disruption of service to the user community. The transition of all the servers to utilize the new firewall took no more that a few hours to perform and after completing the server additions, I did not make any more changes to the network and firewall. This was so that I could monitor the new firewall logs and server performance.

After two successful days of service with no noticeable problems connected to the changes to the network, I updated the DHCP server settings to reflect IP address of the new firewall as the default gateway. I sent an email to all the users in the company, requesting that people reboot their system so that their computers would load the new settings. What caught me by surprise is that 90% of the company rebooted immediately, totally unusual for general users! I continued to monitor the logs on the new firewall for any anomalies as well as address any networking issues raised by the user community. It was only after a further two days that I was confident to shut down the old firewall and remove it completely from the network.

Creating a VPN Tunnel

The company, looking to expand its presence in the marketplace, looked overseas for an acquisition. After finding a company that we wanted to acquire, the due diligence process started however, it was apparent early in this process that the need to communicate securely, both with email as well as file sharing, was of paramount importance. I contacted the overseas IT manager and found out that their firewall is running SUSE Linux IPTables with FreeSwan providing IPSec secure tunneling. Since building a secure tunnel requires that each site share certificates or in my case, host based keys, there is a certain amount of trust required by both parties before setting up any tunnels. This trust relationship is the basis for the secure communications to work correctly.

I downloaded the FreeSwan, version 2.0, source software from the FreeSwan project web site and installed the package in the default location, /usr/local/sbin. To fully support FreeSwan IPSec tunneling, the kernel has to include module support. The basic idea when working with an IPSec tunnel is that there is a “left” and “right” side of the tunnel which should be consistent on both hosts. Authentication between each side is via host key verification which is generated on each host involved in the tunnel and inserted into the ipsec.conf file.

Once the software was installed, I generated a host key for my firewall using the command "*ipsec newhostkey –output /etc/ipsec.secrets –hostname myhost.mydomain.com*". Next was to configure the */etc/ipsec.conf* (see appendix B for an example) file with the correct IP addresses and keys which we were generated on each system with the command previously.

Incorporating the IPSec tunnel interface into the Shorewall installation is relatively simple. You need to edit the *zone* file and add a zone for the IPSec tunnel which in my case is called VPN1. I included a numeric value to this tunnel as I may have to incorporate more sites later and this will give me the flexibility to do this with the minimum amount of reconfiguration. I then edited the Shorewall *tunnels* file which requires the type of tunnel, in this case IPSec, a zone and gateway address which is the IP address of the participating firewall or system. In the *interfaces* file, you associate the zone with the interface and then add specific rules to the *policy* file. The final files to edit are the *start*, *stop* and *init* files which enable you to start the IPSec tunnel after Shorewall has configured all the interfaces on the firewall. Once this is all completed, a restart of the firewall should verify that the tunnel will restart correctly on a reboot.

Good Security Practices

There are some well documented, simple but effective security practices to follow, that will enable you to secure the operating system of your firewall, which is the first line of defense to your network in the "Defense in-depth" practice. As with most security practices, you need to balance security with functionality and practicality. There are ways that you can make the system so secure that it is nearly impossible to administer the system without rebooting into single user mode however, does this justify the disruption that your use community is going to experience?

A good practice is to disable any process(es) that is(are) not required by the system to function, remember this is a firewall, so consider the system an appliance e.g. printing support. If you have to run a service, is there a secure alternative and if so, select this software service instead of the insecure one.

Make sure that your server is up-to-date on all current software patches, whether or not the software is enabled. If you do not want to continually update the software installed on the system, consider removing the software package altogether however, this may produce some strange results or behavior in the system.

Ensure that all local system account passwords follow the guidelines posted on the SANS Institute web site and may be found at http://www.sans.org/top20/#u4 . Passwords should be changed on a regular basis and this is especially true for privileged or system accounts.

There are numerous tools available to harden an operating system but my favorite is called Titan. This is a set of scripts, written by people well known in computer security circles, has been available for many years and works on a variety of different versions of UNIX. Since Titan is script based, you are able to either run the full set of scripts or a subset, depending on your requirements.

You should perform security audits against your firewall on a regular basis. This can either be performed by a third party company or yourself from outside the firewall. Outside company's that perform these services can be costly. Two software tools that I used to scan and test my firewall on a regular basis are CIS Scan and Nessus. The CIS tool scans the local host for know vulnerabilities and Nessus scans the firewall ports from an outside source. While these tools are not completely fool proof, they do provide a good snapshot of the overall security of your system and firewall. There are also web sites on the internet that you can use to probe your firewall with port scans as well as known vulnerabilities, these include http://www.grc.com and http://www.pcflank.com/stealth_test1.htm .

An important point to note is that these tests must be run on a regular basis as well as checking the software package home sites for updates and patches.

Summary

The GSEC course has been a great help to me throughout this project by solidifying my overall security knowledge in both the Windows and UNIX environments as well as introducing tools to monitor and test the overall security of the network and firewall. I have successfully updated the corporate security policy with stronger rules and policy decisions, as well as including the reasoning and explanations behind these rules that are now in place.

The new firewall is an improvement on the old system, both from a performance perspective as well as increased productivity for our customer support personnel. We now have a secure DMZ zone and are moving servers that require access from the Internet into this zone as quickly as possible, providing more security to the LAN. Ongoing maintenance and updates to the firewall configuration is easy, reducing the amount of training required as well as any mistakes.

As with all security solutions, there will always be vulnerabilities from Trojans or viruses coming through the firewall via a system that is tunneled into an infected client network or users loading infected software, however, this is where the "Defense in-depth" approach helped to identify small steps required to reduce the overall risk to the company. As part of this "Defense in-depth" practice, we now update anti-virus software definitions on all systems every 3 days and these updates are pushed out rather than relying on the users to remember to run the updates manually. We have installed a Microsoft software management server that provides operating system updates, also in a push scenario, with little or no

user intervention. This service is performed on an as-needed basis, relying on security updates from SANS and Microsoft.

There is still more work to be done to the firewall especially in monitoring the log file and parsing the large amount of data into a more meaningful format but, this is another discussion.

Appendix A

Shorewall.conf

```
#############################################################################
# /etc/shorewall/shorewall.conf V1.4 - Change the following variables to
# match your setup
#
# This program is under GPL [http://www.gnu.org/copyleft/gpl.htm]
#
# This file should be placed in /etc/shorewall
#
# (c) 1999,2000,2001,2002,2003 - Tom Eastep (teastep@shorewall.net)
#############################################################################
# L O G G I N G
#############################################################################
LOGFILE=/var/log/messages
LOGFORMAT="Shorewall:%s:%s:"
LOGRATE=
LOGBURST=
LOGUNCLEAN=info
BLACKLIST_LOGLEVEL=
LOGNEWNOTSYN=
MACLIST_LOG_LEVEL=info
TCP_FLAGS_LOG_LEVEL=debug
RFC1918_LOG_LEVEL=debug
PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/bin:/usr/local/sbin
SUBSYSLOCK=/var/lock/subsys/shorewall
STATEDIR=/var/lib/shorewall
MODULESDIR=
FW=fw
NAT_ENABLED=Yes
MANGLE_ENABLED=Yes
IP_FORWARDING=On
ADD_IP_ALIASES=Yes
ADD_SNAT_ALIASES=Yes
TC_ENABLED=Yes
CLEAR_TC=No
MARK_IN_FORWARD_CHAIN=No
CLAMPMSS=No
ROUTE_FILTER=Yes
NAT_BEFORE_RULES=No
MULTIPORT=Yes
DETECT_DNAT_IPADDRS=Yes
MUTEX_TIMEOUT=60
NEWNOTSYN=Yes
BLACKLIST_DISPOSITION=DROP
```

MACLIST_DISPOSITION=REJECT
TCP_FLAGS_DISPOSITION=DROP
#LAST LINE -- DO NOT REMOVE

Zones File

```
#
# Shorewall 1.4 -- Sample Zone File For Two Interfaces
# /etc/shorewall/zones
#
# This file determines your network zones. Columns are:
#
# ZONE Short name of the zone
# DISPLAY Display name of the zone
# COMMENTS Comments about the zone
#
#ZONE DISPLAY COMMENTS
net Net Internet
loc Local Local Networks
dmz DMZ Demilitarized Zone
vpn1 VPN1 VPN to Germany
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

Interfaces File

```
###############################################################################
#ZONE INTERFACE BROADCAST OPTIONS
net eth0 62.123.106.127 routefilter,norfc1918,blacklist,tcpflags
loc eth1 detect dhcp,routefilter
dmz eth2 detect
vpn1 ipsec0
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
Routestopped File:
#INTERFACE HOST(S)
eth1 -
eth2 -
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Policy File

```
###############################################################################
#SOURCE DEST POLICY LOG LEVEL LIMIT:BURST
loc net ACCEPT
loc fw ACCEPT
loc dmz ACCEPT
# If you want open access to the Internet from your Firewall
# remove the comment from the following line.
fw net ACCEPT
fw loc ACCEPT
```

```
fw dmz ACCEPT
dmz fw ACCEPT
dmz loc ACCEPT
dmz net ACCEPT
#
# Adding VPN Access
loc vpn1 ACCEPT
dmz vpn1 ACCEPT
fw vpn1 ACCEPT
vpn1 loc ACCEPT
vpn1 dmz ACCEPT
vpn1 fw ACCEPT
#
net all DROP info
all all REJECT info
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

Masq File

```
#INTERFACE SUBNET ADDRESS
eth0 eth1 1192.0.18.126
#
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

NAT File

```
#EXTERNAL INTERFACE INTERNAL ALL INTERFACES LOCAL
#
# Intranet Web Server
192.0.18.115 eth0:0 10.10.1.60 No No
#
# Project Web Server
192.0.18.84 eth0:1 10.10.1.55 No No
#
# Blackberry Server
192.0.18.97 eth0:2 10.10.1.55 No No
#
# Corporate Mail Server
192.0.18.93 eth0:3 10.10.1.252 No No
#
# Second Corp Mail Server
192.0.18.70 eth0:4 10.10.1.8 No No
#
# Sims Server
192.0.18.75 eth0:5 10.10.1.56 No No
#
#LAST LINE -- ADD YOUR ENTRIES ABOVE THIS LINE -- DO NOT REMOVE
```

Proxy ARP

```
#ADDRESS INTERFACE EXTERNAL HAVEROUTE
#
# The Corporate email server in the DMZ
192.0.18.80 eth2 eth0 No
#
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Tunnels File

```
# TYPE ZONE GATEWAY GATEWAY ZONE PORT
ipsec net 134.147.129.82
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Rules File

```
#############################################################################
#ACTION SOURCE DEST PROTO DEST SOURCE ORIGINAL
# PORT PORT(S) DEST
#
# Accept DNS connections from the firewall to the network
#
ACCEPT fw net tcp 53
ACCEPT fw net udp 53
#
# Accept SSH from internet interface from kaos only
#
ACCEPT net:192.0.18.98 fw tcp 22
#
# Accept connections from the local network for administration
#
ACCEPT loc fw tcp 20:22
ACCEPT loc net tcp 22
ACCEPT loc fw tcp 53
ACCEPT loc fw udp 53
ACCEPT loc net tcp 53
ACCEPT loc net udp 53
#
# Allow Ping To And From Firewall
#
ACCEPT loc fw icmp 8
ACCEPT loc dmz icmp 8
ACCEPT loc net icmp 8
ACCEPT dmz fw icmp 8
ACCEPT dmz loc icmp 8
ACCEPT dmz net icmp 8
DROP net fw icmp 8
DROP net loc icmp 8
```

```
DROP net dmz icmp 8
ACCEPT fw loc icmp 8
ACCEPT fw dmz icmp 8
DROP fw net icmp 8
#
# Accept proxy web connections from the inside
#
ACCEPT loc fw tcp 8118
#
# Forward PcAnywhere, Oracle and Web traffic from outside to the Demo
systems
# From a specific IP Address on the Internet.
#
# ACCEPT net:207.65.110.10 loc:10.10.3.151 tcp 1521,http
# ACCEPT net:207.65.110.10 loc:10.10.2.32 tcp 5631:5632
#
# Intranet web server
ACCEPT net loc:10.10.1.60 tcp 443
ACCEPT dmz loc:10.10.1.60 tcp 443
#
# Projects web server
ACCEPT net loc:10.10.1.55 tcp 80
ACCEPT dmz loc:10.10.1.55 tcp 80
#
# Blackberry Server
ACCEPT net loc:10.10.1.230 tcp 3101
#
# Corporate Email Server
ACCEPT net loc:10.10.1.252 tcp 25,53,110,143,443
#
# Corporate #2 Email Server
ACCEPT net loc:10.10.1.8 tcp 25,80,110,443
#
# Sims Server
ACCEPT net loc:10.10.1.56 tcp 80,443
ACCEPT net loc:10.10.1.56 tcp 7001:7002
ACCEPT net:63.83.198.0/24 loc:10.10.1.56 tcp 5631:5632
#
# Access to DMZ
ACCEPT loc dmz udp 53,177
ACCEPT loc dmz tcp 80,25,53,22,143,443,993,20,110 -
ACCEPT net dmz udp 53
ACCEPT net dmz tcp 25,53,22,21,123
ACCEPT dmz net tcp 25,53,80,123,443,21,22
ACCEPT dmz net udp 53
```

```
#
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Start File

```
#############################################################################
# Shorewall 1.4 -- /etc/shorewall/start
#
# Add commands below that you want to be executed after shorewall has
# been started or restarted.
#
qt service ipsec start
```

Stop File

```
#############################################################################
# Shorewall 1.4 -- /etc/shorewall/stop
#
# Add commands below that you want to be executed at the beginning of a
# "shorewall stop" command.
#
qt service ipsec stop
```

Init File

```
#############################################################################
# Shorewall 1.4 -- /etc/shorewall/init
#
# Add commands below that you want to be executed at the beginning of
# a "shorewall start" or "shorewall restart" command.
#
qt service ipsec stop
```

Appendix B

Sample /etc/ipsec.conf file

```
# /etc/ipsec.conf - FreeS/WAN IPsec configuration file
# RCSID $Id: ipsec.conf.in,v 1.10 2003/04/23 05:02:24 sam Exp $

# This file:  /usr/local/share/doc/freeswan/ipsec.conf-sample
#
# Manual:     ipsec.conf.5
#
# Help:
# http://www.freeswan.org/freeswan_trees/freeswan-2.00/doc/quickstart.html
# http://www.freeswan.org/freeswan_trees/freeswan-2.00/doc/config.html
# http://www.freeswan.org/freeswan_trees/freeswan-2.00/doc/adv_config.html
#
# Policy groups are enabled by default. See:
# http://www.freeswan.org/freeswan_trees/freeswan-2.00/doc/policygroups.html
#
# Examples:
# http://www.freeswan.org/freeswan_trees/freeswan-2.00/doc/examples


version 2.0     # conforms to second version of ipsec.conf specification

# basic configuration
config setup
        # Debug-logging controls:  "none" for (almost) none, "all" for lots.
        # klipsdebug=all
        # plutodebug=dns
        interfaces="ipsec0=eth0"
        # Debug-logging controls:  "none" for (almost) none, "all" for lots.
        klipsdebug=none
        plutodebug=none
        # Close down old connection when new one using same ID shows up.
        uniqueids=yes

# Add connections here.
# defaults for subsequent connection descriptions
conn %default
        # How persistent to be in (re)keying negotiations (0 means very).
        keyingtries=0
        type=transport
        left=192.0.18.66
        leftid=192.0.18.66
        esp=3des-md5-96
```

```
        leftnexthop=192.0.18.65
        leftfirewall=yes
        # RSA 2192 bits   achilles.viisage.net   Mon Aug  4 14:22:12 2003

leftrsasigkey=0sAQN4wjQLoYinl0139KNjUr2RHbmO5EFGJb+Uuk0V9VD3//LZB
yuFDldOc+nFcBCyoh8WFMcH1exJMhsYRem66lX90vJnJZ5AcJlY7xL4kEYmnQj
Z8NYmQKovE7p9hzfh0k4/bGrnLIZTO+6QRiEpcr2wBfOc5p+MbfnOoqAy2824yA
yjl+QDlgVgPzv0R6Ei+4gw1/OB/waKcc3vne8ZaYlVGxV3rkKsgQx6/n+bfJ6kSzS5
94Nv6FpmKDrCgF9EvWzuu0zz3qMdacwxt1e8rwBi5tuP4cCvke4LzLe9sEQAEa
3lHI/N12cSi21oek5q0clntAMVp/jaK5EAHWdBQlobg3qa9bQ5tkN2aHMquOi+tLA
Z


conn clear
        auto=ignore
conn clear-or-private
        auto=ignore
conn private-or-clear
        auto=ignore
conn private
        auto=ignore
conn block
        auto=ignore
conn packetdefault
        auto=ignore

conn xyzdmz-to-acmecorp
        leftsubnet=192.168.12.80/28
        leftid=@ipsec1.xyzdmz.com
        left=192.168.28.43
        leftnexthop=192.168.28.18
        right=192.0.18.65
        rightid=@ipsec1.acmecorp.com
        rightsubnet=10.10.0.0/22
        rightnexthop=192.0.18.65
        type=tunnel
        authby=rsasig
        auto=start
        pfs=yes

leftrsasigkey=0sAQNJu0JRaLnqUqnNC4YzvhR3zsDE6/NVQylSI/wXi2dSNCsT1
++oiDKkJvYtZZFCHXG5SWLYFyAgarKNt3MPdLI7mgBnlM3uYSzh3cBD4X+Uxc
FBuTD3EUeHkvRMSYCwXQa1RrEIsD8d41W1VTrCOaSlHxQAcRn8ICCgsv0n+
WfnyPDjWUu8M7RBLoEzZMJmdPhveoEetB1RWNUu2kxAsHUtDvAZw8SSGdig
Y/sWwSzzt1LYQPjHJQcacBGB4xDce3pqW4MLE5HxJFrpvdfvOUELCIpYwvnhm
dQE8zmUzEuKvugV/E5v018ZIxvWkVTKQ/e4Kg0Qacn8pcmnyhC0eTZf
```

Appendix C

Kernel Netfilter components.

```
#
#   IP: Netfilter Configuration
#
CONFIG_IP_NF_CONNTRACK=m
CONFIG_IP_NF_FTP=m
CONFIG_IP_NF_AMANDA=m
CONFIG_IP_NF_TFTP=m
CONFIG_IP_NF_IRC=m
CONFIG_IP_NF_IPTABLES=y
CONFIG_IP_NF_MATCH_LIMIT=m
CONFIG_IP_NF_MATCH_MAC=m
CONFIG_IP_NF_MATCH_PKTTYPE=m
CONFIG_IP_NF_MATCH_MARK=m
CONFIG_IP_NF_MATCH_MULTIPORT=m
CONFIG_IP_NF_MATCH_TOS=m
CONFIG_IP_NF_MATCH_ECN=m
CONFIG_IP_NF_MATCH_DSCP=m
CONFIG_IP_NF_MATCH_AH_ESP=m
CONFIG_IP_NF_MATCH_LENGTH=m
CONFIG_IP_NF_MATCH_TTL=m
CONFIG_IP_NF_MATCH_TCPMSS=m
CONFIG_IP_NF_MATCH_HELPER=m
CONFIG_IP_NF_MATCH_STATE=m
CONFIG_IP_NF_MATCH_CONNTRACK=m
CONFIG_IP_NF_FILTER=y
CONFIG_IP_NF_TARGET_REJECT=m
CONFIG_IP_NF_NAT=m
CONFIG_IP_NF_NAT_NEEDED=y
CONFIG_IP_NF_TARGET_MASQUERADE=m
CONFIG_IP_NF_TARGET_REDIRECT=m
CONFIG_IP_NF_NAT_AMANDA=m
CONFIG_IP_NF_NAT_LOCAL=m
CONFIG_IP_NF_NAT_IRC=m
CONFIG_IP_NF_NAT_FTP=m
CONFIG_IP_NF_NAT_TFTP=m
CONFIG_IP_NF_MANGLE=y
CONFIG_IP_NF_TARGET_TOS=m
CONFIG_IP_NF_TARGET_ECN=m
CONFIG_IP_NF_TARGET_DSCP=m
CONFIG_IP_NF_TARGET_MARK=m
CONFIG_IP_NF_TARGET_LOG=m
CONFIG_IP_NF_TARGET_ULOG=m
```

```
CONFIG_IP_NF_TARGET_TCPMSS=m
CONFIG_IP_NF_ARPTABLES=y
CONFIG_IP_NF_ARPFILTER=m
```

References

IPTables vs. Ipchains: http://testweb.oofle.com/iptables/comparison.htm

Eastep, Tom. “Shorewall – Iptables made easy”. 13 November 2003.
URL: http://www.shorewall.net (9 June 2003)

Andreasson, Oskar. Iptables Tutorial 1.1.19. 28 May 2003
URL: http://iptables-tutorial.frozentux.net/iptables-tutorial.html (3 June 2003)

Russell, Rusty. Linux IPCHAINS-HOWTO. 4 July 2000
URL: http://www.netfilter.org/ipchains/HOWTO.html (22 May 2003)

NetFilter Team. Netfilter/Iptables Home page 2 November 2003
URL: http://www.netfilter.org

Kurland, Vadim. Firewall Builder Home page. 23 November 2003
URL: http://www.fwbuilder.org (9 June 2003)

Link, Zachary. Jwall.
URL: http://jwall.sourceforge.net (9 June 2003)

Tsaousis, Costa. Firehol Home page. 13 October 2003
URL: http://firehol.sourceforge.net/ (9 June 2003)

Junnonen, Tomas. Firestarter Home page. 17 August 2003
URL: http://firestarter.sourceforge.net/ (9 June 2003)

Hubinger, Christian. Kmyfirewall Home page. 8 December 2003
URL: http://kmyfirewall.sourceforge.net/ (9 June 2003)

Lasser, Jon and Beale, Jay. Bastille Linux Home page. 5 May 2003
URL: http://www.bastille-linux.org/ (9 June 2003)

Gilmore, John (Founder and Executive) Freeswan Home page. 13 November 2003
URL: http://www.freeswan.org (24 June 2003)

Farmer, Dan, Powell, Brad and Archibald, Matt. Titan Home page
URL: http://www.fish.com/titan/ (26 June 2003)

Center for Internet Security. Linux Benchmarks. October 2003
URL: http://www.cisecurity.org/bench_linux.html (14July 2003)

Deraison, Renaud. Nessus. 5 November 2003
URL: http://www.nessus.org/ (16 July 2003)

The Linux Documentation Project. 11 November 2003
URL: http://www.tldp.org
RedHat Home page
URL: http://www.redhat.com

RedHat Network
URL: http://rhn.redhat.com

Sourceforge: http://sourceforge.net

Google Search Engine: http://www.google.com