Global Information Assurance Certification Paper

GIAC Enterprises

GIAC Practical Assignment
Version 1.9

George D.(Danny) Walker II, CISSP, CISA, SSCP, GCIA, GCIH
June 22, 2003

Table of Contents

Security Architecture

Company Overview

GIAC Enterprises has been selling fortune cookie sayings for 3 generations. Jack Chan has been given control with the death of his father, Hu Chan, and with the help of his sons, John and Adam, he wishes to grow it into the 21st century. Previously Hu Chan focused on the continental United States to sell his witty sayings. Jack wanted to expand the company's influence across the world and planned to use the Internet to help him.

John, the younger of the two brothers, is a graduate of University of Virginia with a degree in Management Information Systems and is half way through towards an MBA, and has worked for a major financial institution as a junior manager 3 years. Adam was a product of the Internet growth phenomena. Right out of high school he started working for a local Internet Service Provider as a junior system administrator and has been working for in the industry for 12 years and currently acting as a Senior Network Security Engineer for a national networking company. John was brought on as Chief Operating Officer (COO) and Adam became the Chief Information Officer (CIO) and both became partners.

The trio came up with the idea of fostering a virtual company that enabled Adam to remain in Boulder, Colorado with his family while John moved back home to Richmond, Virginia with his father. The idea was to leverage the Internet and technology to enable the company to work from home offices instead of incurring the cost of a physical location. Since the company was going to have to start with a small budget Adam decided to use open source software as the initial solution and then upgrades could be added in the future. John worked with Jack on how GIAC Enterprises would structure the business and came up with the following operations and business plan.

Business Operations Overview

As stated above Jack Chan wished to bring GIAC Enterprises into the 21st century and to do this he and John came up with the following business operations plan.

Customers

Customers are actual fortune cookie companies that buy the fortune in both a bulk format (sayings that can be directly put on paper slips) as well as in actual useable form (on required paper slips).

Partners

Partnerships have been formed with companies in South America, Europe, Middle East, Africa and Asia. These companies take the fortune cookie sayings and translate them into the appropriate language for their countries. Then they resell the fortunes to those foreign markets. These companies have exclusive rights in these areas of influence, but GIAC Enterprises earns a commission on all sales they procure (i.e. through web contacts or North American affiliates).

Suppliers
Suppliers include the two companies that provide GIAC Enterprises with their paper slips and their printing services.

Employees
The Chief Executive Officer (CEO), Chief Operating Officer (COO) and Chief Information Officer (CIO) would each be issued a SOHO VPN/Firewall Router and a high-speed Internet connection to enable connectivity to the company network.

To support expanding operations GIAC Enterprises would have to add staff. This would include the following:

Vice President of Sales – This position would be responsible for managing partner relationships as well as supervising the three-salesperson positions. A laptop with VPN client would be issued to allow connection to GIAC Enterprise's network. Although a company paid high-speed Internet connection would be provided the laptop would also connect via Internet dialup when in the field. This position would report to the COO.

Eastern US Salesperson – This position would be responsible for the eastern United States territory. A laptop with VPN client would be issued to allow connection to GIAC Enterprise's network via Internet dialup. This position would be located within the eastern United States and report to the VP of Sales.

Western US Salesperson – This position would be responsible for the western United States territory. A laptop with VPN client would be issued to allow connection to GIAC Enterprise's network via Internet dialup. This position would be located within the western United States and report to the VP of Sales.

Canada Salesperson – This position would be responsible for the Canadian territory. A laptop with VPN client would be issued to allow connection to GIAC Enterprise's network via Internet dialup. . This position would be located within Canada and report to the VP of Sales.

Security/ Systems Administrator – This position would be responsible for system administration and security operations of network. This position would be located in Boulder, Colorado and would report to the CIO.

Access Requirements and Restrictions
Based on the above business operations plan, Adam and John have designed the network to support the needs of the new virtual company.

Customers
- Access to customer webserver via SSL using LDAPS for authentication back to LDAP server.

Partners

- Access to partner webserver via SSL using LDAPS for authentication back to LDAP server.

Suppliers

- Jack and John would use the supplier's extranet to order paper slip supplies and their printing services. They are negotiating with the various suppliers to develop a plan to enable automated ordering of their supplies.

Employees

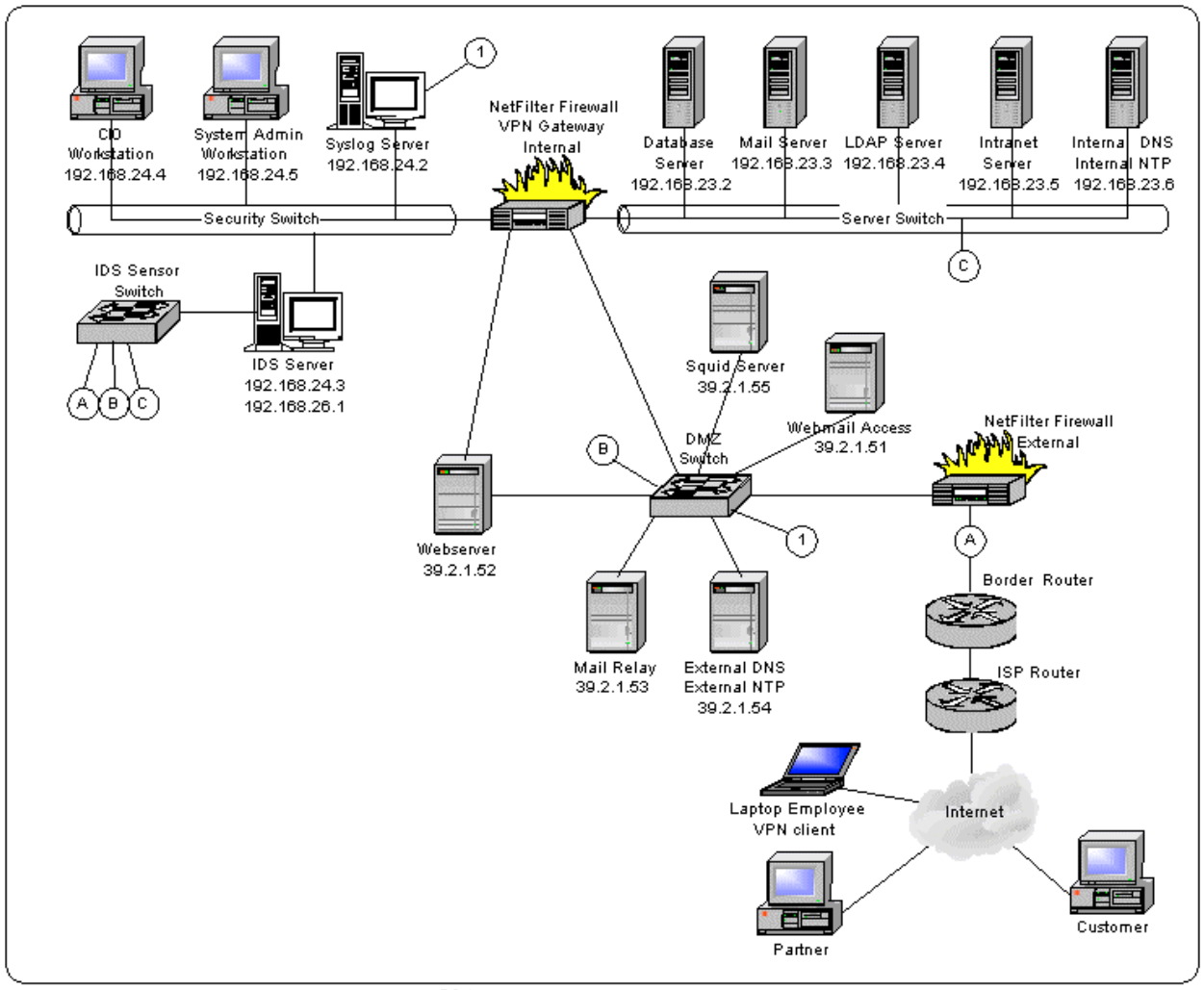
- All remote employees require the ability to connect to internal servers including LDAP server, mail server, database server, file server and intranet. They will access the network using one of following three ways:
  - SOHO IPsec VPN/Firewall router using high speed Internet (cable or DSL).
  - IPsec client using high speed Internet (cable or DSL).
  - IPsec client using a dialup Internet connection.
- System Administrators also need access to SSH from their local workstations.

Summary of User Access Requirements

Group	Requirement	Service	Port
Customers	Access to public webserver	HTTP HTTPS	80 tcp 443 tcp
Customers	Webserver to Database server	MySQL	3306 tcp
Customers	Webserver to LDAP server	LDAPS	636 tcp
Partners	To public webserver	HTTP HTTPS	80 tcp 443 tcp
Partners	Webserver to Database Server	MySQL	3306 tcp
Partners	Webserver to LDAP Server	LDAPS	636 tcp
Employees	Access to VPN server (access to LDAP, Database, Mail and Intranet)	IKE ESP	500 udp protocol 50
Employees	Access to WebMail server	HTTPS	443 tcp
Employees	Webmail server to Mail server	IMAP	143 tcp
Employees	Webmail server to LDAP server	LDAPS	636 tcp
System Administrators	Access to all servers from internal workstation or from across VPN	SSH	22 tcp
System Administrators	Access to Internet web servers	HTTP HTTPS	80 tcp 443 tcp

Technical Architecture

Adam has designed this network using the principals of "Defense in Depth" by layering multiple security controls. This will enable the design to be resistant to intrusion attempts. We will go over the various layers of security components that make up this network.



GIAC-enterprises network diagram

Internet Connection

Adam was able to secure a 10x10 office in the building of Boulders largest ISP. He has made arrangements for a T1 connection and will connect directly to the ISP router via Ethernet. This connection can be upgraded as the company bandwidth needs increase. The ISP is a tier one Internet provider offering a service level agreement with guaranteed 99.5% uptime. In addition, the office space has its own physical security system, card access control system, fire detection and control systems, and security guards.

Router

For our border router we will use a Cisco 2600 router (http://www.cisco.com) running version 12.2(17) of the Cisco IOS software. The border router will filter inbound and outbound traffic at the edge of the network using standard and extended access control lists (ACLs). The Router is only accessible via a serial connection to the CIO Workstation.

NetFilter Firewalls

We will implement 2 stand-alone firewalls both using the NetFilter firewall v1.2.8 software (http://www.netfilter.org) on a hardened Red Hat 9 OS machine. The external firewall will sit

between our border router and DMZ switch and will provide an additional layer of traffic filtering. The internal firewall will sit between the DMZ switch, Security switch, Server switch and will connect to the second NIC of the public webserver. In addition the internal firewall acts as a VPN gateway using the FreeS/wan IPsec 2.01 software (http://www.freeswan.org). A strict "Deny all unless specifically allowed" policy is implemented across both firewalls.

In addition we have implemented host-based NetFilter firewalls on the Database server, Mail server, LDAP server, Intranet/Samba Fileserver, Internal DNS/NTP server, Syslog Server, IDS server, External DNS/NTP server, Mail Relay server and Workstations.

IDS server/sensors

We will deploy 3 IDS sensors running Snort IDS v.2.0 software (http://www.snort.org) on a hardened Red Hat 9 OS machine. Sensor placement is addressed below. Each sensor's ruleset will be consistently tuned to limit the number of false-positives.

1. Logically located between the Border router and External Firewall using NetOptics 10/100BaseT Network tap (http://www.netoptics.com). The snort sensor will have dual NICs with one sniffing the TX channel and one sniffing the RX channel of a full duplex connection.
2. Directly connected to the spanning port of the DMZ switch.
3. Directly connected to the spanning port of the Server switch.

These sensors will all report back to the IDS server across an out-of-band network via the sensor switch. The IDS server will be run the Analysis Console for Intrusion Databases (ACID) v.0.9.6b23 (http://www.andrew.cmu.edu/~rdanyliw/snort/snortacid.html) using a Apache webserver, MySQL database and PHP on a hardened Red Hat 9 OS machine. The IDS server will have a host-based NetFilter firewall on networked interface allowing only incoming/outgoing HTTPS (443 tcp), MySQL (3306 tcp), DNS queries (53 udp), NTP (123 udp) and SSH (22 tcp) traffic.

Syslog server

The Syslog server contains a custom web front-end to a MySQL database application. Administrators connect to a secure Apache webserver to check the Syslog entries. Using msyslog v1.09d (http://sourceforge.net/projects/msyslog), a replacement for the standard Syslog daemon, incoming entries are sent to a MySQL database. The server is dual homed with one NIC connected to the Security switch to receive direct communication from the Database server, Mail server, LDAP server, Intranet/Samba Fileserver, Internal DNS/NTP server, Webserver and Internal Firewall/VPN gateway. The other NIC will be connected to the DMZ switch on a non-IP-addressed interface. For each server (Webmail server, External DNS/NTP and Mail Relay server) and External Firewall we have to do the following to ensure they properly send Syslog data to the Syslog server:

1. In /etc/syslog.conf a selector field of "*.info" and action field of "@39.2.1.62" should be added. Note that 39.2.1.62 doesn't actually exist and will simply be a bogus host.
2. Since we are using a switch we have to tell each server and the firewall where to find this bogus host. For that we will add a static ARP entry of "arp –s 39.2.1.62 00:00:00:00:00:00" where 39.2.1.62 is the bogus host and 00:00:00:00:00:00 is the actual

MAC address of our non-IP-addressed interface of the Syslog server. This will ensure that the data gets sent to the proper switch port.

3. On the Syslog server we will have a stealth-logger using Snort. This logger will implement the frag2 preprocessor in case the packets come across fragmented. Then the logger will simply pass the logged data

The Syslog server will have a host-based NetFilter firewall on both interfaces. On the external interface it allows only incoming/outgoing HTTPS (443 tcp), DNS queries (53 udp), NTP (123 udp) and SSH (22 tcp) traffic and incoming Syslog (514 udp) traffic. On the internal interface it allows only incoming Syslog (514 udp) traffic.

Web server

We will be using Apache 2.0 (http://www.apache.com) with mod_php, mod_auth_mysql, mod_ssl and mod_auth_ldap on a hardened RedHat 9 OS machine. The server will be dual homed with an external interface connected to the DMZ switch and the internal interface connected directly to the eth3 interface of the Internal Firewall/VPN gateway. The webserver uses the internal interface to access the LDAP server, Database server, send Syslog data to the Syslog server and allows incoming SSH for management purposes. The Webserver will have a host-based NetFilter firewall on both interfaces. On the external interface it allows only incoming/outgoing HTTP (80 tcp) and HTTPS (443 tcp) traffic. On the internal interface it allows only incoming/outgoing LDAPS (686 tcp), MySQL (3306 tcp), DNS queries (53 udp), NTP (123 udp) and SSH (22 tcp) traffic and outgoing Syslog (514 udp) traffic.

Database server

For our database server we will use MySQL 4.0.13 (http://www.mysql.com) on a hardened RedHat 9 OS machine. This server will serve as the backend for our webserver. The Database server will have a host-based NetFilter firewall on its single interfaces where it will allow only incoming/outgoing MySQL (3306 tcp), DNS queries (53 udp), NTP (123 udp) and SSH (22 tcp) traffic and outgoing Syslog (514 udp) traffic.

LDAP server

For our LDAP authentication server we are using OpenLDAP v2.1.22 (http://www.openldap.org) on a hardened RedHat 9 OS machine. This server will allow users to authenticate to a variety of internal servers as well as allow customers and partners to access the webserver. The LDAP server will have a host-based NetFilter firewall on its single interfaces where it will allow only incoming/outgoing SLDAP (636 tcp), DNS queries (53 udp), NTP (123 udp) and SSH (22 tcp) traffic and outgoing Syslog (514 udp) traffic.

Mail server

Our mail server uses PostFix v1.1.1 (http://www.postfix.org) for its MTA and Courier IMAP v1.7.3 (http://inter7.com/courierimap.html) for its IMAP services on a hardened RedHat 9 OS machine. Authentication for mail is performed at the LDAP server. In addition users can use the webmail server to access their mail. For users that pass across the VPN to access mail they will use Opera v7.11 (http://www.opera.com) mail client to access the mail server. The mail server will only accept incoming and send outgoing SMTP traffic from/to the Mail Relay found in the DMZ. The Mail server will have a host-based NetFilter firewall on its single interfaces where it

will allow only incoming/outgoing SMTP (25 tcp), DNS queries (53 udp), IMAP (143 tcp), NTP (123 udp) and SSH (22 tcp) traffic and outgoing Syslog (514 udp) traffic.

Mail Relay

Our mail relay uses PostFix v.1.1.1 and RAV AntiVirus v8.4.2 (http://www.ravantivirus.com) on a hardened RedHat 9 OS machine. The relay accepts mail from the world runs it through the anti-virus scanner and removes certain types of attachments and then forwards it on to the Mail Server. When a user wants to send mail out the mail relay accepts the mail from the Mail server, runs it through the anti-virus scanner and then delivers it. The Mail Relay server will have a host-based NetFilter firewall on its single interfaces where it will allow only incoming/outgoing SMTP (25 tcp), DNS queries (53 udp), NTP (123 udp) and SSH (22 tcp) traffic and outgoing Syslog (514 udp) traffic.

Squid server

Our squid server uses Squid Web Proxy Cache v2.5 (http://www.squid-cache.org) with the Jeanne reverse proxy plugin (http://www.ists.dartmouth.edu/IRIA/projects/d_jeanne.htm) on a hardened Red Hat 9.0 OS. This reverse proxy server provides an additional barrier between the webserver and the untrusted public network. All traffic coming from the world would be directed to this server and provides additional filtering and then it would forward on valid requests to the webserver. A second instance of squid will also be installed to proxy HTTP and HTTPS traffic for the CIO and System Administrators machines on port 3128. This proxy will provide no caching for this purpose and will only be used by the two internal users. This will be one of the first services moved to another machine as soon as funding permits, but this solution was better than nothing. This machine will have a host-based NetFilter firewall on its single interfaces where it will allow only incoming/outgoing HTTP (80/3128/8080 tcp), DNS queries (53 udp), HTTPS (443 tcp), NTP (123 udp) and SSH (22 tcp) traffic and outgoing Syslog (514 udp) traffic.

Webmail server

Our webmail server uses SquirrelMail v1.40 (http://www.squirrelmail.com), Apache v2.0 and PHP4 on a hardened RedHat 9 OS machine. This server allows employees to access their mail remotely via a web browser. The Webmail server will have a host-based NetFilter firewall on its single interfaces where it will allow only incoming/outgoing IMAP (143 tcp), DNS queries (53 udp), HTTPS (443 tcp), NTP (123 udp) and SSH (22 tcp) traffic and outgoing Syslog (514 udp) traffic.

Intranet server

Our Intranet server uses Samba v2.2.8a (http://us2.samba.org/samba/samba.html), Apache v2.0 and on a hardened RedHat 9 OS machine. The server authenticates using LDAPS and employees can access it after coming through the VPN gateway. This server will have a host-based NetFilter firewall on its single interfaces where it will allow only incoming/outgoing HTTP (80 tcp),
DNS queries (53 udp), NTP (123 udp) and SSH (22 tcp) traffic and outgoing Syslog (514 udp) traffic

DNS/NTP server

We are running two sets of servers running our Domain Naming System (DNS) and Network Time Protocol (NTP) services on a hardened RedHat 9 OS machine. We have an external server sitting on the DMZ switch and an internal server sitting on the Server switch. We are running BIND 8.4.1 (http://www.isc.org) and NTP 4.1.1 (http://www.ntp.org) on a hardened Red Hat 9.0 OS.
We will run a split DNS which separates the internal name space from the pubic name space. The external DNS server will only contain information on servers that the Internet requires to know about. The external DNS server will only be recursive for our service network servers. The internal DNS server will contain information on our internal and external servers and will be recursive using the external DNS server to forward any requests to the Internet. The internal DNS server will not support any zone transfers and the external DNS server will support only zone transfers to the ISPs DNS servers. This version of BIND has also been configured to use a source port of udp 53 for all communication between the external and internal DNS servers. In addition both servers have logging enabled that reports activity to the Syslog server.

Our two NTP servers serve different areas of the network. Due to the small budget for the project Adam made the decision to use one server on the DMZ switch and one on the Server switch instead of the recommended two redundant servers connecting to two different hosting sites. The primary purpose of the NTP servers is to synchronize time across all of the company servers and devices to enable for accurate security log correlation. Since the National Institute of Standards and Technology Automated Computer Time Service (ACTS) is located in Boulder, Colorado there is a local number. Both the internal and external NTP servers were equipped with a modem that allows only a dial out to the ACTS number. This method was chosen to keep from allowing NTP connections across the internal and external firewalls. Adam plans to upgrade to a set of TymServe 2100 (http://www.tymserve.com) servers to add redundancy to such an important service.

On the external DNS/NTP server there will be a host-based NetFilter firewall and it will allow only incoming/outgoing DNS queries (53 udp), DNS zone transfer (53 tcp) from/to ISPs DNS servers, NTP (123 udp) and SSH (22 tcp) traffic and outgoing Syslog (514 udp) traffic.

Switches

Both the DMZ and Server switches are Cisco Catalyst 2950XL 24-port switches running Cisco IOS 12.1(6). Unused ports are shut off and port security has been implemented. The spanning port is being used for the IDS sensor. The Security and IDS sensor switches are Cisco Catalyst 2950 12-port switches running Cisco IOS 12.1(6). Again unused ports are shut off and port security has been implemented. All switches are accessible via a serial connection to the System Administrators workstation.

SSH Service

The Secure Shell (SSH) service deployed across our environment is a custom compiled version of OpenSSH v3.61p2 (http://www.openssh.org). In addition only the SSH2 protocol is enabled due to security issues with SSH1.

System/Network Administration

Servers will utilize a defined backup policy for data and use Tripwire (http://www.tripwire.com) to protect the integrity of their system files. Each server will be built with a hardened version of Red Hat and the administrators will maintain the patches for both the OS and applications.

Remote Users

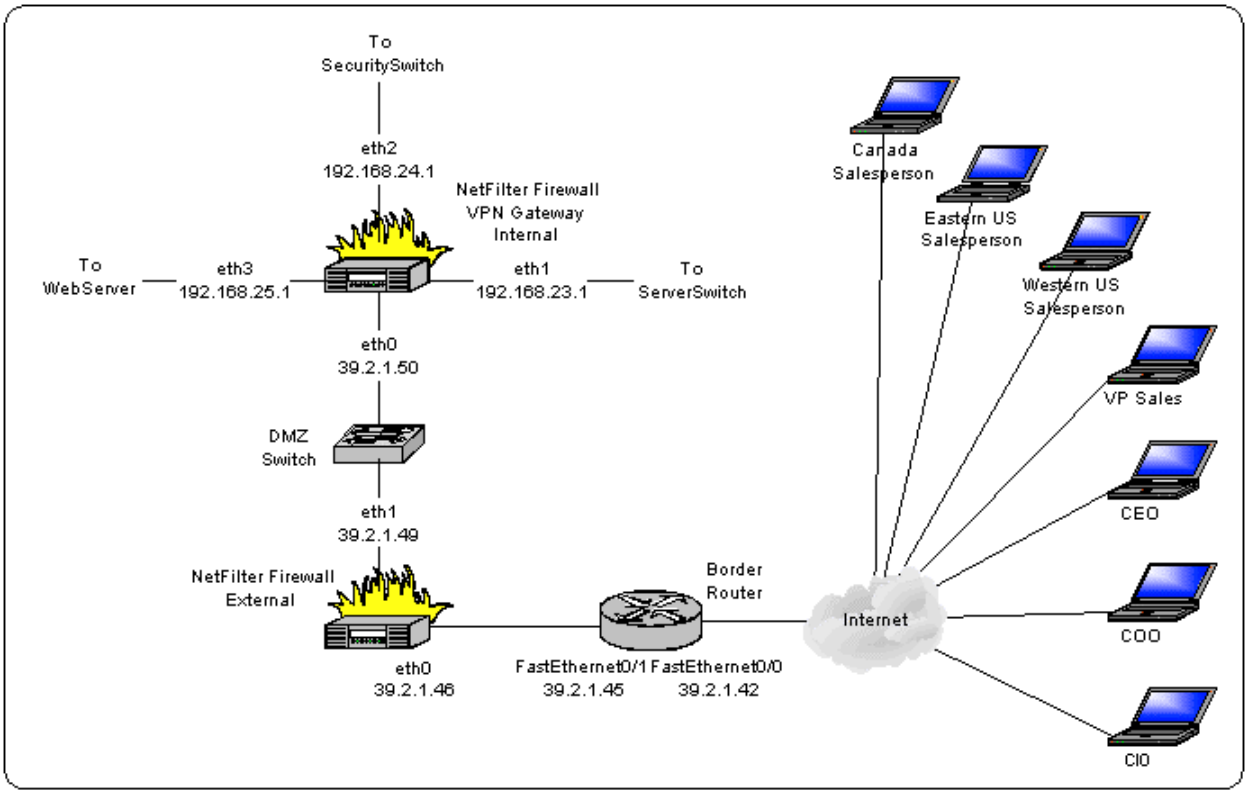
Home users will use a custom VPN client installed on a laptop. This client will allow the user to connect only to either the Internet or to the corporate VPN at one time. This will limit the ability of a Trojan getting installed and tunneling an attacker directly into the network. Each workstation will be maintained remotely with the latest patches for the OS and applications.

IP addressing

GIAC Enterprises was assigned three subnets by its ISP and its routable address space is broken down as follows:

- 39.2.1.40/30 – For use between ISP router and border router
  - 2 hosts consisting of 39.2.1.41 and 39.2.1.42
  - Network address of 39.2.1.40
  - Netmask address of 255.255.255.252
  - Broadcast address of 39.2.1.43
- 39.2.1.44/30 – For use between Border router and External Firewall
  - 2 hosts consisting of 39.2.1.45 and 39.2.1.46
  - Network address of 39.2.1.44
  - Netmask address of 255.255.255.252
  - Broadcast address of 39.2.1.47
- 39.2.1.48/28 – For routable service network
  - 14 hosts between 39.2.1.49 and 39.2.1.62
  - Network address of 39.2.1.48
  - Netmask address of 255.255.255.240
  - Broadcast address of 39.2.1.63

NOTE: The entire 39.0.0.0/8 net range is actually reserved by the Internet Assigned Numbers Authority (IANA) but for the purposes of this paper I have assumed that it has actually been assigned to GIAC Enterprises. This information can be verified at http://ws.arin.net/cgi-bin/whois.pl?queryinput=N%20!%20NET-39-0-0-0-1.



GIAC Enterprises IP addresses

Device/Server	Connection	Interface Name	IP address
Router	External	FastEthernet0/0	39.2.1.42/30
	Internal	FastEthernet0/1	39.2.1.45/30
External Firewall	External	Eth0	39.2.1.46/30
	Internal	Eth1	39.2.1.49/28
Internal Firewall/ VPN GW	DMZ Switch	Eth0	39.2.1.50/28
	Server Switch	Eth1	192.168.23.1/24
	Security Switch	Eth 2	192.168.24.1/24
	Web server	Eth 3	192.168.25.1/24
External DNS / NTP server	DMZ Switch	Eth0	39.2.1.54/28
Mail Relay	DMZ Switch	Eth0	39.2.1.53/28
Squid server	DMZ Switch	Eth0	39.2.1.55/28
Webserver	DMZ Switch	Eth0	39.2.1.52/28
	Internal Firewall	Eth1	192.168.25.2/24
Webmail Access	DMZ Switch	Eth0	39.2.1.51/28
Database server	Server Switch	Eth0	192.168.23.2/24
Mail server	Server Switch	Eth0	192.168.23.3/24
LDAP server	Server Switch	Eth0	192.168.23.4/24
Intranet server	Server Switch	Eth0	192.168.23.5/24
Internal DNS / NTP server	Server Switch	Eth0	192.168.23.6/24

Syslog Server	Security Switch	Eth0	192.168.24.2/24
	DMZ Switch	Eth1	No IP
IDS Database Server	Security Switch	Eth0	192.168.24.3/24
	Sensor Switch	Eth1	192.168.26.1/24
IDS Sensor A	Sensor Switch	Eth0	192.168.26.2/24
	RX channel of Tap	Eth1	No IP
	TX channel of Tap	Eth2	No IP
IDS Sensor B	Sensor Switch	Eth0	192.168.26.3/24
	DMZ Switch	Eth1	No IP
IDS Sensor C	Sensor Switch	Eth0	192.168.26.4/24
	Server Switch	Eth1	No IP
CIO workstation	Security Switch	Eth0	192.168.24.4/24
System Admin workstation	Security Switch	Eth0	192.168.24.5/24

Network Devices/Servers

Device/ Servers	Hardware	Total Cost
Router	Cisco 2600 Router	$879
External Firewall	Dell PowerEdge 1750 (Xeon 3.0 GHz proc [dual capable], 2 GB DDR RAM, RAID-1 card, 2-10K RPM 36 GB SCSI HD, 100bT NIC, 3 yrs Onsite-4hr parts service)	$3,280
Internal Firewall/ IPsec Gateway	Dell PowerEdge 1750 (Dual Xeon 3.0 GHz proc, 2 GB DDR RAM, RAID-1 card, 2-10K RPM 36 GB SCSI HD, 100bT NIC, 3 yrs Onsite-4hr parts service)	$3,590
External DNS and NTP server	Dell PowerEdge 650 (P4 2.6 GHz proc, 1GB DDR RAM, RAID-1 card, 2-7.2K RPM 20GB IDE HD, 100bT NIC, 3yrs. Onsite-4hr parts service)	$1,685
Mail Relay	Dell PowerEdge 650 (P4 2.6 GHz proc, 1GB DDR RAM, RAID-1 card, 2-7.2K RPM 20GB IDE HD, 100bT NIC, 3yrs. Onsite-4hr parts service)	$1,685
Squid server	Dell PowerEdge 1750 (Xeon 3.0 GHz proc [dual capable], 2 GB DDR RAM, RAID-1 card, 2-10K RPM 36 GB SCSI HD, 100bT NIC, 3 yrs Onsite-4hr parts service)	$3,280
Webserver	Dell PowerEdge 1750 (Xeon 3.0 GHz proc [dual capable], 2 GB DDR RAM, RAID-1 card, 2-10K RPM 36 GB SCSI HD, 100bT NIC, 3 yrs Onsite-4hr parts service)	$3,280
Webmail Access	Dell PowerEdge 650 (P4 2.6 GHz proc, 1GB DDR RAM, RAID-1 card, 2-7.2K RPM 20GB IDE HD, 100bT NIC, 3yrs. Onsite-4hr parts service)	$1,685
DMZ Switch	Cisco Catalyst 2950XL 24-port switch	$695
Database Server	Dell PowerEdge 1750 (Xeon 3.0 GHz proc [dual capable], 2 GB DDR RAM, RAID-1 card, 2-10K RPM 146 GB SCSI HD, 100bT NIC, 3 yrs Onsite-4hr parts service)	$3,585
Mail server	Dell PowerEdge 650 (P4 2.6 GHz proc, 1GB DDR RAM,	$1,685

	RAID-1 card, 2-7.2K RPM 20GB IDE HD, 100bT NIC, 3yrs. Onsite-4hr parts service)	
LDAP server	Dell PowerEdge 650 (P4 2.6 GHz proc, 1GB DDR RAM, RAID-1 card, 2-7.2K RPM 20GB IDE HD, 100bT NIC, 3yrs. Onsite-4hr parts service)	$1,685
Intranet Fileserver	Dell PowerEdge 650 (P4 2.6 GHz proc, 1GB DDR RAM, RAID-1 card, 2-7.2K RPM 20GB IDE HD, 100bT NIC, 3yrs. Onsite-4hr parts service)	$1,685
Internal DNS and NTP	Dell PowerEdge 650 (P4 2.6 GHz proc, 1GB DDR RAM, RAID-1 card, 2-7.2K RPM 20GB IDE HD, 100bT NIC, 3yrs. Onsite-4hr parts service)	$1,685
Server Switch	Cisco Catalyst 2950XL 24-port switch	$595
Syslog Server	Dell PowerEdge 1750 (Xeon 3.0 GHz proc [dual capable], 2 GB DDR RAM, RAID-1 card, 2-10K RPM 146 GB SCSI HD, 100bT NIC, 3 yrs Onsite-4hr parts service)	$3,585
IDS Database Server	Dell PowerEdge 1750 (Xeon 3.0 GHz proc [dual capable], 2 GB DDR RAM, RAID-1 card, 2-10K RPM 146 GB SCSI HD, 100bT NIC, 3 yrs Onsite-4hr parts service)	$3,585
IDS sensors (3)	Dell PowerEdge 650 (P4 2.6 GHz proc, 1GB DDR RAM, RAID-1 card, 2-7.2K RPM 20GB IDE HD, 100bT NIC, 3yrs. Onsite-4hr parts service)	$4,500
Sensor Switch	Cisco Catalyst 2950 12-port switch	$559
Security Switch	Cisco Catalyst 2950 12-port switch	$559
Laptops (9)	Latitude D800 (1.7 GHz proc, 512 MB DDR RAM, 30 GB HD, FD, CD, 3 yrs Onsite-4hr parts service	$13,500
Miscellaneous	Surge suppressors, cat5 cable, UPS, 2 Racks, OS, software, etc	$7,000
	TOTAL	$64,267

GIAC Enterprises, Inc. is using Red Hat 9.0 OS on its workstations and servers. To this end we have purchased 2 licenses for Red Hat Enterprise server, 3 licenses for Red Hat Workstation and has a subscription to the Enterprise level of their Red Hat Network. Although our servers run a customized version of Red Hat 9.0 and don't actually run any of the purchased copies we feel this way we are supporting Red Hat in addition to getting continued support.

Information on equipment and pricing from following sites:

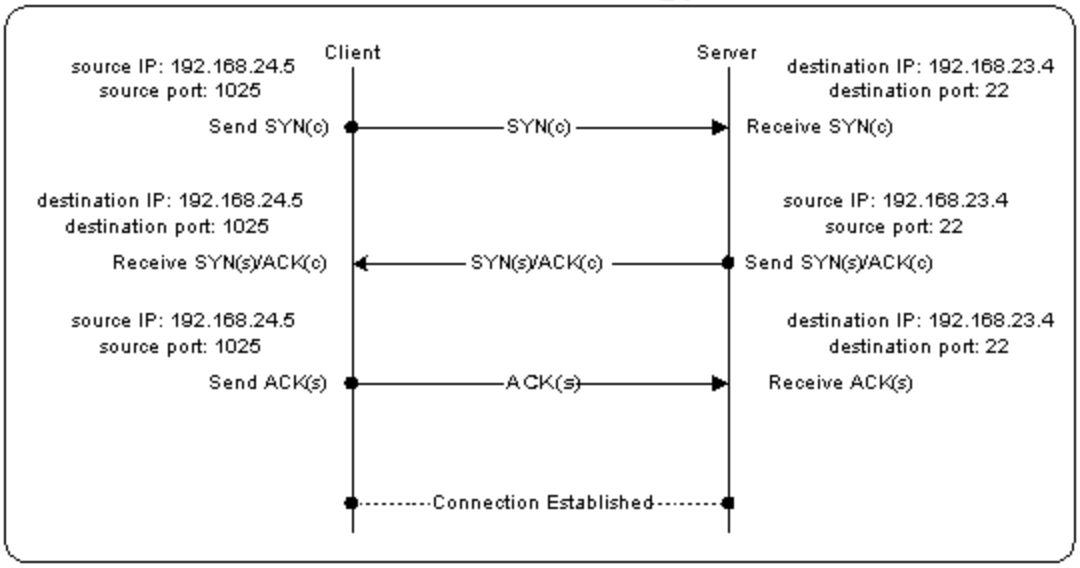
- PC Connection (http://www.pcconnection.com)
- Dell (http://www.dell.com)
- Cisco (http://www.cisco.com)
- RedHat (http://www.redhat.com)
- Cyberguys (http://www.cyberguys.com)

Security Policy and Tutorial

Since the above access requirements simply define what services that the Customers, Partners and Employees need to it really define the complete set of traffic that a firewall would have to deal with. This we would requires a traffic analysis to determine what the expected traffic looks like.

Traffic Analysis

Traffic analysis looks at the traffic that would be seen on a similar configured network. By looking at the services and servers running on the network we can put together a complete table of expected traffic. This would include the above defined access requirements, server-to-server communication, and any other expected traffic that would exist on the network. An added benefit of this process the determination of what servers require what traffic to function which would allow the firewalls to be locked down to only allow certain traffic to connect to certain servers by way of their IP addresses. Another benefit of this process would be the knowledge gained on the network traffic expected to see on the network. For instance, the SSH protocol that is required between the System Administrators Workstation (192.168.24.5) and the LDAP server (192.168.23.4) requires a standard TCP three-way handshake.



TCP three-way handshake diagram [1]

As seen in the diagram above the System Administrators Workstation becomes the client and the SSH server on the LDAP server becomes the server. When the client makes its initial connection it sends a packet with a destination port of 22 and a source port of 1025. This source port is considered an ephemeral port, or a number that is automatically allocated by the TCP/IP stack from a predefined range. This transient port changes for each connection so we will not be able to lock traffic down by it. A look at each protocol's RFC (http://www.ietf.org/rfc/) would be the easiest way to determine its standard communication methods.

Network Traffic Tuning

Adam follows the premise that an efficient network is a secure network and vice versa. As found with the traffic analysis above knowing what traffic should and shouldn't be traveling across his network is the first step to securing the network properly. Unfortunately a typical corporate network is not static. As new services and applications are added this initial rulebase will require modification. This will be especially obvious when a large amount of logging occurs, recording denied traffic is in fact is legitimate. When this occurs the master traffic analysis list should be adjusted to show the necessary changes required across the network.

RuleBase

From the traffic analysis we can build a basic rulebase, the actual traffic our firewall and routers will allow. Since our traffic analysis covers all traffic on our network, not all traffic will traverse each of our firewalls and/or router (i.e. SSH traffic from our admin workstation does not need to traverse our external firewall, internal firewall or router to access our IDS server since they are both on the same network, but it will need rules added to both host-based firewalls). In addition, there will be rules added to deny certain types of traffic (i.e. IANA reserved subnets or manually blacklisted hosts/networks) and allow other types of traffic needed for network performance (i.e. ICMP fragmentation-needed packets). Note that as network traffic tuning happens each firewall and router rulebase might be affected and may require modification.

Border Router Policy

The border router is the first line of defense for our network. It uses access control lists (ACL) to filter traffic that crosses it, which can provide a basic method for eliminating unwanted traffic.

As with our firewalls, the routers follow a "Deny all unless specifically allowed" policy. The router should apply the rules in the following order:

- Anti-spoofing filters and Blocked private addresses
- User permit rules
- Management permit rules
- Noise drops (Router traffic)
- Deny and Log suspicious traffic
- Deny any traffic left

Also remember that ACLs are order dependent. Each incoming packet is checked by each access-list in order from top to bottom and when a packet matches the criteria in any one of the access-lists then that action is performed.

Initial configuration

The router will initially be configured offline using its console port and a TFTP server. Adam will manage the router and switches using the minicom program on his workstation with each tied to a unique COM port. This will allow the remote management of these devices if required. Once the initial configuration is complete then the router may need to have this procedure

repeated since editing ACLs requires the entire list to be removed and reapplied. To facilitate this, Adam will maintain the router configuration in text format and when changes are required he will do the following:

1. Disconnect the external router connection from Internet.
2. Connect the internal router interface to the security switch.
3. Quickly change his workstation IP to one that can communicate to the router (i.e. ifconfig eth0 39.2.1.60 255.255.255.240).
4. Log onto router via console and activate enable mode.
5. Bring down host firewall.
6. Start TFTP server on workstation.
7. On router type "copy running-config tftp" and complete necessary information regarding TFTP server and file to be copied over.
8. Review running-config by typing in "show running-config"
9. If correct, then copy running-config to startup-config by typing "copy running-config startup-config"
10. Disconnect router from security switch and connect to DMZ switch.
11. Reboot router and using console watch boot process and check new configuration.
12. If it looks okay then reconnect external interface to Internet.
13. Then reset workstation by restarting networking to reapply old networking settings (i.e. service network restart), turn off the TFTP server and reapply the host firewall (i.e. service netfilter restart).

Router Configuration Breakdown

Global Configuration

Set up router name

```
hostname rogue
```

Store routers enable secret password in MD5 encrypted format

```
service password-encryption
```

Set secret and enable passwords which will be stored in encrypted format

```
enable secret 5 "enable secret"
enable password 7 "enable password"
```

Assign password to console

```
line console 0
login password "Rwnlttwc2t" (Router will not let traffic thru without
checking 2 times)
```

Disable access to AUX port by removing any existing access list number 10, setting access list number 10 to deny all and log and then apply access list number 10 to line aux 0.

```
no access-list 10
access-list 10 deny any log

line aux 0
access-class 10 in
```

```
transport input all
ip access-group 10 in
```

Disable remote administration via Virtual Terminal Lines (Telnet) by removing any existing access list number 11, setting access list number 11 to deny all and log and then apply access list number 11 to all virtual terminal lines (i.e. line vty 0 4)

```
no access-list 11
access-list 11 deny any log

line vty 0 4
access-class 11 in
transport input none
login local
exec-timeout 0 1
no exec
```

Prevent abuse of small services (i.e. echo, discard, chargen, etc)

```
no service tcp-small-servers
no service udp-small-servers
```

Prevent abuse of finger service

```
no service finger
```

Disable Cisco Discovery Protocol service globally

```
no cdp run
```

Prevent abuse of SNMP service by first ensuring that all previous configuration statements have been removed, then erasing the old community strings, disabling SNMP trap and system-shutdown features and then disabling the SNMP service.

```
no snmp-server community public RO
no snmp-server community admin RW
no snmp-server enable traps
no snmp-server system-shutdown
no snmp-server trap-auth
no snmp-server
```

Prevent abuse of DNS name resolution

```
no ip domain-lookup
```

Prevent IP source routing options from being used

```
no ip source-route
```

Prevent abuse of http server

```
no ip http server
```

Disable loading startup configuration from network

```
no boot network
no service config
```

Establish standard banner

```
banner login #
Use of this computer system, authorized or unauthorized, constitutes consent
to the policies and procedures set forth by GIAC-Enterprises, Inc. All
information placed on or sent to this system may be subject to monitoring
procedures. Evidence of unauthorized use collected during monitoring
```

```
may be used for criminal prosecution by staff, legal counsel and law
enforcement agencies.
#
```

Setup timestamping service for logging purposes

```
service timestamps log datetime localtime show-timezone msec
```

Set timezone to GMT

```
clock timezone GMT 0
```

Setup logging component to Syslog server. (NOTE: we are sending traffic to the Bogus IP for stealth logger)

```
logging on
logging trap debugging
logging facility local1
logging 39.2.1.62
```

Set up NTP servers

```
ntp server 39.2.1.54
```

Restrict the use of subnet zero addresses

```
ip subnet-zero
```

Limit use of tftp to internal interface

```
ip tftp source-interface FastEthernet0/0
```

Configure a gateway of last resort (to ISPs Router)

```
ip default-gateway 39.2.1.41
```

Enables the use of subnet addresses

```
ip classless
```

Internal Network Interface Configuration

Configure FastEthernet 0/1 interface

```
interface FastEthernet0/0
```

Setup IP address for interface

```
 ip address 39.2.1.45 255.255.255.252
```

Prevent use of router as 'smurf' amplifier

```
 no ip directed-broadcast
```

Disable fast switching algorithms

```
 no ip route-cache
```

Disable multicast route cache

```
 no ip mroute-cache
```

Auto detect wire speed

```
 speed auto
```

Defaults to half-duplex transmission

```
 half-duplex
```

External Network Interface Configuration

Configure FastEthernet 0/1 interface

```
interface FastEthernet0/1
```

Setup IP address for interface

```
ip address 39.2.1.42 255.255.255.252
```

Apply access-group 101 to all inbound traffic

```
ip access-group 101 in
```

Apply access-group 102 to all outbound traffic

```
ip access-group 102 out
```

Prevent use of router as 'smurf' amplifier

```
no ip directed-broadcast
```

Disable 'Host unreachable' ICMP messages

```
no ip unreachable
```

Disable 'Redirect' ICMP messages

```
no ip redirect
```

Disable 'Mask Reply' ICMP messages

```
no mask-reply
```

Disable Ad-hoc routing

```
no ip proxy-arp
```

Disable fast switching algorithms

```
no ip route-cache
```

Disable mutlicast route cache

```
no ip mroute-cache
```

Auto detect wire speed

```
speed auto
```

Defaults to half-duplex transmission

```
half-duplex
```

Disable NTP service

```
no ntp enable
```

Access List for incoming traffic on external interface

Reset access-list 101

```
no access-list 101
```

Setup Anti-spoofing and Non-routable access list

Deny any traffic coming from IPs of router interface (LAND attack) and log

```
access-list 101 deny ip host 39.2.1.42 host 39.2.1.42 log
access-list 101 deny ip host 39.2.1.45 host 39.2.1.45 log
```

Deny any traffic coming from internal IP routable address range and log

```
access-list 101 deny ip 39.2.1.48 0.0.0.240 log
access-list 101 deny ip 39.2.1.44 0.0.0.252 log
```

Deny icmp redirect

```
access-list 101 deny icmp any any redirect
```

Deny localhost traffic

```
access-list 101 deny ip 127.0.0.0 0.255.255.255 any
```

Deny multicast traffic

```
access-list 101 deny 224.0.0.0 7.255.255.255 any
```

Deny non-routable addresses

```
access-list 101 deny ip any 10.0.0.0 0.255.255.255
access-list 101 deny ip any 172.16.0.0 0.15.255.255
access-list 101 deny ip any 192.168.0.0 0.0.255.255
```

Setup Access list for incoming traffic on external interface

Permit required ICMP traffic

```
access-list 101 permit icmp any any administratively-prohibited
access-list 101 permit icmp any any packet-too-big
access-list 101 permit icmp any any time-exceeded
access-list 101 permit icmp any any unreachable
```

Permit incoming http/https traffic to squid proxy

```
access-list 101 permit tcp any host 39.2.1.55 eq 80
access-list 101 permit tcp any host 39.2.1.55 eq 443
```

Permit incoming https traffic to WebMail server

```
access-list 101 permit tcp any host 39.2.1.51 eq 443
```

Permit incoming smtp traffic to Mail Relay

```
access-list 101 permit tcp any host 39.2.1.53 eq 25
```

Permit IPSec IKE negotiations

```
access-list 101 permit udp any host 39.2.1.50 eq 500
```

Permit IPSec ESP protocol

```
access-list 101 permit esp any host 39.2.1.50
```

Permit incoming DNS zone transfers to External DNS/NTP server

```
access-list 101 permit tcp host <ISP DNS server> host 39.2.1.54 eq 53
```

Permit established tcp connections

```
access-list 101 permit tcp any 0.0.0.0 255.255.255.0 established
```

Deny all remainder tcp traffic

```
access-list 101 deny tcp any any
```

Deny all remainder udp traffic

```
access-list 101 deny udp any any
```

Deny any other protocol types and log

```
access-list 101 deny ip any any log
```

Access List for outgoing traffic on external interface

Reset access-list 102

```
no access-list 102
```

Deny outgoing time-exceeded and unreachable ICMP messages to limit the ability to do network enumeration

```
access-list 102 deny icmp any any time-exceeded
access-list 102 deny icmp any any unreachable
```

Minimize malicious insider/worm communication threat by ensuring that external http/https/smtp/dns traffic can only come from Squid proxy, Web Mail server, Mail Relay or External DNS/NTP server.

Permit outgoing http/https traffic from Squid proxy and https traffic from Web Mail server and deny/log all others

```
access-list 102 permit tcp host 39.2.1.55 eq 80 any
access-list 102 permit tcp host 39.2.1.55 eq 443 any
access-list 102 permit tcp host 39.2.1.51 eq 443 any
access-list 102 deny tcp any any eq 80 log
access-list 102 deny tcp any any eq 443 log
```

Permit outgoing smtp traffic from Mail Relay and deny/log all others

```
access-list 102 permit tcp host 39.2.1.53 eq 80 any
access-list 102 deny tcp any any eq 80 log
```

Permit outgoing traffic from External DNS/NTP server and deny/log all others.

```
access-list 102 permit udp host 39.2.1.53 any eq 53
access-list 102 permit tcp host 39.2.1.53 host <ISP DNS server> eq 53
access-list 102 deny udp any any eq 53 log
access-list 102 deny tcp any any eq 53 log
```

Permit outgoing traffic from VPN gateway server

Permit IPSec IKE negotiations

```
access-list 102 permit udp host 39.2.1.50 any eq 500
```

Permit IPSec ESP protocol

```
access-list 102 permit esp host 39.2.1.50 any
```

Deny and log all other outgoing traffic

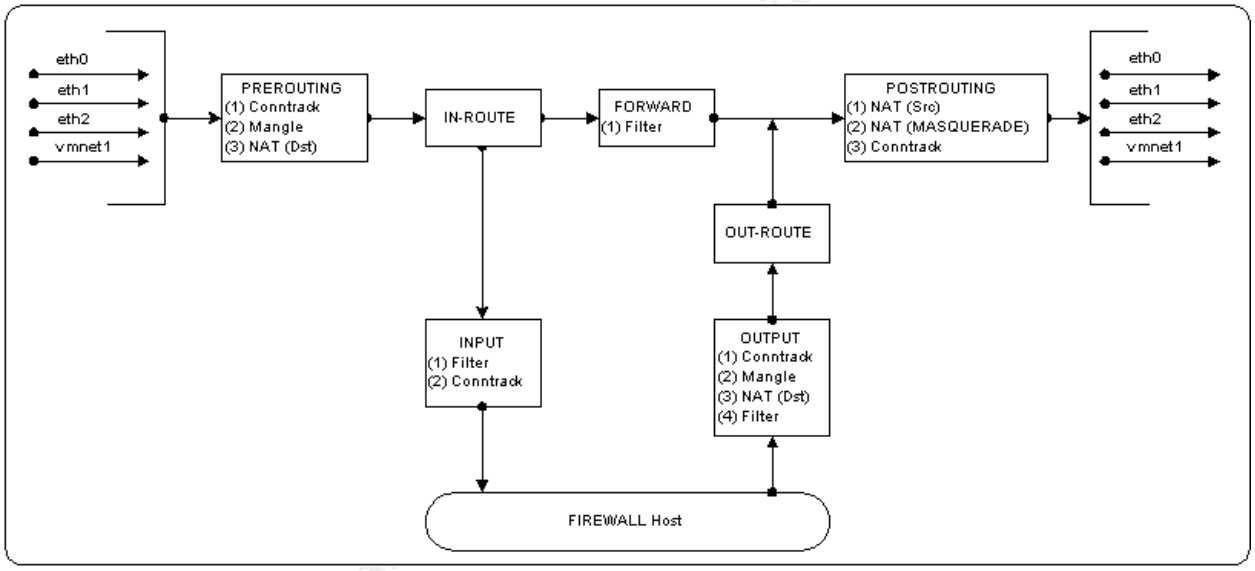
```
access-list 102 deny ip any any log
```

++
For each rule in all policies, you must include the general purpose of the rule and why it is important.

You must also include a discussion of the order of the rules, and why order is (or is not) important.
++

NetFilter Firewall Overview

The firewalls use Netfilter v1.28 (http://www.netfilter.org/). As with the router above it operates on a first-match-wins process so the first rule that the packet matches would be what action it performs.



Flow a Packet Travels

External Netfilter Firewall

The firewall configuration file has been written as a traditional Red Hat init file found in /etc/init.d. This style would allow for the administrator to run one of the following commands to modify the firewall

```
[root@EXTERNAL /]#service netfilter {start|stop|restart|debug|status|panic)
```

where

- Start – starts firewall
- Stop – stops firewall and opens up networking
- Restart – restarts the firewall (Uses Stop and Start above)

- ? Debug – sets the Debug flag and then starts firewall. This adds additional logging.
- ? Status – gives the ruleset for the firewall
- ? Panic – locks down the firewall by denying all traffic

This firewall should apply the rules in the following order:

- ? Anti-spoofing filters and Blocked private addresses
- ? User permit rules
- ? Management permit rules
- ? Deny standard trojan/worm traffic (See http://isc.incidents.org/top10.html)
- ? Deny and Log suspicious traffic (including Blacklist from below)
- ? Deny remainder of traffic

Setup global variables. Identify location of iptables command.

```
IPT=/usr/local/sbin/iptables
```

Define LOGLEVEL variable and set DEBUG variable

```
LOGLEVEL=notice
DEBUG=0
```

Define Interfaces – changes between test and production environments

```
EXTIF=vmnet1
INTIF=eth0
```

Ensure iptables command is available or Error and exit

```
if [ ! -x $IPT ]; then
      echo "ERROR: Cannot locate iptables command."
      exit 0
fi
```

Ensure ipchains command is not running or Error and exit

```
if  /sbin/lsmod 2>/dev/null |grep -q ipchains ; then
      echo "ERROR: Cannot run both ipchains and iptables"
      exit 0
fi
```

Set up local variables. Define location of the blacklist.conf file which will setup the BLACKLIST chain. Define the location of the vpnclient.conf file which will add rules to allow authorized users to access the VPN.

```
BLACKLIST=/etc/blacklist.conf
VPNCLIENT=/etc/vpnclient.conf
```

Log to screen and syslog the start message

```
echo "Starting External NetFilter Firewall"
logger -p local3.info NetFilter Firewall Started
```

 # Enable syn cookie support

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

 # Enable logging of Martians (oddball addresses).

```
echo 1 > /proc/sys/net/ipv4/conf/all/log_martians
```

 # Enable IP Forward support.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

 # Disable response to ping

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
```

 # Disable response to Broadcasts (to keep from being SMURF amplifier)

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

 # Enable Dynamic IP Address support.

```
echo 1 > /proc/sys/net/ipv4/ip_dynaddr
```

 # Enable anti spoofing support.

```
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter
```

 # Disable source routing support.

```
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route
```

 # Disable icmp redirect support.

```
echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects
```

 # Flush,delete and zero existing chains

```
$IPT -t filter -F
$IPT -t nat -F
$IPT -t mangle -F
$IPT -X
$IPT -Z
```

Set up default policies for nat, mangle and filter tables.

```
$IPT -t nat -P PREROUTING ACCEPT
$IPT -t nat -P POSTROUTING ACCEPT
$IPT -t nat -P OUTPUT ACCEPT
$IPT -t mangle -P PREROUTING ACCEPT
$IPT -t mangle -P INPUT ACCEPT
$IPT -t mangle -P FORWARD ACCEPT
$IPT -t mangle -P OUTPUT ACCEPT
$IPT -t mangle -P POSTROUTING ACCEPT
$IPT -t filter -P FORWARD DROP
$IPT -t filter -P INPUT DROP
$IPT -t filter -P OUTPUT DROP
```

NAT table module rules - This table is consulted when a packet that creates a new connection is encountered. Use PREROUTING to drop unwanted new traffic before passing on to FORWARD or INPUT Filters.

Block Reserved/Private Addresses coming in the external interface. Data pulled from http://www.iana.org/assignments/ipv4-address-space. Dropped 39.0.0.0/8 from Reserved list since I have given part of this address space to GIAC Enterprises.

This creates a rule for each of these defined IANA_RESERVED addresses and IANA_Multicast addresses by looping through each of them.

```
IANA_RESERVED="0 1 2 5 7 23 27 31 36 37 41 42 58 59 70 71 72 73 74 75 76 77
78 79 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104
105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123
124 125 126 127 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187
189 190 197 223 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254
255"

for NET in $IANA_RESERVED; do
      $IPT -t nat -A PREROUTING -i $EXTIF -s $NET.0.0.0/8 -j DROP
done

IANA_MULTICAST="224 225 226 227 228 229 230 231 232 233 234 235 236 237 238
239"

for NET in $IANA_MULTICAST; do
      $IPT -t nat -A PREROUTING -i $EXTIF -s $NET.0.0.0/8 -j DROP
done
```

IANA PRIVATE IPv4 address space via RFC 1918.

```
$IPT -t nat -A PREROUTING -i $EXTIF -s 10.0.0.0/8 -j DROP
$IPT -t nat -A PREROUTING -i $EXTIF -s 172.16/12 -j DROP
$IPT -t nat -A PREROUTING -i $EXTIF -s 192.168.0.0/16 -j DROP
$IPT -t nat -A PREROUTING -i $EXTIF -s 192.88.99.0/24 -j DROP
```

Mangle Module Rules

Rules to mangle TOS values of packets routed through the firewall that is based on RFC 1060/1349. This includes the following settings:

- **Minimize-Delay 16 (0x10)**
- **Maximize-Throughput 8 (0x08)**
- **Maximize-Reliability 4 (0x04)**
- **Minimize-Cost 2 (0x02)**
- **Normal-Service 0 (0x00)**

Set minimize-delay for SSH, SMTP and DNS traffic and set maximize-throughput for HTTP traffic.

```
$IPT -t mangle -A PREROUTING -p tcp --dport 22 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 25 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p udp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 80 -j TOS --set-tos 8
```

Filter Module Rules

Setup debugging for tracking traffic flows. Activated if DEBUG variable is 1.

```
if [ $DEBUG = 1 ]; then
      echo "Extra Debug logging enabled"
```

Log all localhost traffic entering and leaving the LO interface

```
$IPT -A INPUT -i lo -j LOG --log-level $LOGLEVEL \
      --log-prefix "Local-in: "
$IPT -A OUTPUT -o lo -j LOG --log-level $LOGLEVEL \
```

```
      --log-prefix "Local-out: "
```

Log all Encapsulated Security Payload (ESP) traffic entering and leaving the external interface.

```
$IPT -A FORWARD -p 50 -i $EXTIF -j LOG --log-level $LOGLEVEL \
      --log-prefix "ESP-in: "
$IPT -A FORWARD -p 50 -i $INTIF -j LOG --log-level $LOGLEVEL \
      --log-prefix "ESP-out: "
```

Log all Internet Key Exchange (IKE) traffic entering and leaving the external interface.

```
$IPT -A FORWARD -p udp -i $EXTIF --sport 500 --dport 500 -j LOG \
      --log-level $LOGLEVEL --log-prefix "IKE-in: "
$IPT -A FORWARD -p udp -i $INTIF --sport 500 --dport 500 -j LOG \
      --log-level $LOGLEVEL --log-prefix "IKE-out: "
```

Log all DNS, NTP, SYSLOG, HTTP, HTTPS, SMTP and SSH traffic traversing the FORWARD chain of the firewall.

```
$IPT -A FORWARD -p udp --sport 53 --dport 53 -j LOG \
      --log-level $LOGLEVEL --log-prefix "DNS: "
$IPT -A FORWARD -p udp --sport 123 --dport 123 -j LOG \
      --log-level $LOGLEVEL --log-prefix "NTP: "
$IPT -A FORWARD -p udp --sport 514 --dport 514 -j LOG \
      --log-level $LOGLEVEL --log-prefix "SYSLOG: "
$IPT -A FORWARD -p tcp --dport 80 -j LOG \
      --log-level $LOGLEVEL --log-prefix "HTTP: "
$IPT -A FORWARD -p tcp --dport 443 -j LOG \
      --log-level $LOGLEVEL --log-prefix "HTTPS: "
$IPT -A FORWARD -p tcp --dport 25 -j LOG \
      --log-level $LOGLEVEL --log-prefix "SMTP: "
$IPT -A FORWARD -p tcp --dport 22 -j LOG \
      --log-level $LOGLEVEL --log-prefix "SSH: "
fi
```

Allow all localhost traffic entering and leaving the Loopback interface.

```
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT
```

Allow IPsec into $EXTIF. Filter access to IKE Negotiations to only allow authorized clients defined in /etc/vpnclient.conf. This vpnclient.conf file can use # to comment the file for easy maintenance and will be stripped out when being parsed

```
if [ "$VPNCLIENT" != "" ]; then
      echo "Loading authorized VPN clients from $VPNCLIENT"
      for host in `grep -v ^# $VPNCLIENT | awk '{print $1}'` ; do
```

Setup rules to allow IKE negotiations with defined hosts

```
            $IPT -A FORWARD -p udp -i $EXTIF -s $host --sport 500 \
                  --dport 500 -j ACCEPT
            $IPT -A FORWARD -p udp -i $INTIF --sport 500 -d $host \
                  --dport 500 -j ACCEPT
      echo "VPN Client: $host "
      done
fi
```

Accept ESP encryption and authentication traffic

```
$IPT -A FORWARD -p 50 -i $EXTIF -d 39.2.1.50 -j ACCEPT
$IPT -A FORWARD -p 50 -i $INTIF -s 39.2.1.50 -j ACCEPT
```

Setup new ICMP traffic chain

```
$IPT -N ICMP_TRAFFIC
```

Accept needed ICMP traffic from net for EXTERNAL router. Accept all destination-unreachable traffic

```
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 3 -j ACCEPT
```

Accept all time-exceeded icmp traffic

```
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 11 -j ACCEPT
```

Log ICMP traffic by type and drop.

```
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 0 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Echo Reply: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 8 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Echo Request: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 4 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Source Quench: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 5 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Redirect: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 9 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Router Advertisement: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 10 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Router Selection: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 30 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Traceroute: "
$IPT -A ICMP_TRAFFIC -j DROP
```

Push all ICMP traffic to ICMP chain

```
$IPT -A INPUT -p icmp -j ICMP_TRAFFIC
$IPT -A FORWARD -p icmp -j ICMP_TRAFFIC
$IPT -A OUTPUT -p icmp -j ICMP_TRAFFIC
```

Set up a hot list that uses a file located at /etc/blacklist.conf to parse through when building the ruleset. This would allow the administrator the ability to add IP addresses that have had malicious activity. This blacklist.conf file can use # to comment the file for easy maintenance and will be stripped out when being parsed.

- ? Known bad guys – From Incidents.org Top Ten list (http://isc.incidents.org/top10.html) and Dshield.org recommended block list (http://feeds.dshield.org/block.txt).
- ? Historical problems – Check logs for problem IPs
- ? Firewall subverting services – (i.e. http://www.gotomypc.com)
- ? Countries from which possible state run attacks could come from (and no business relationships within) – China and North Korea

Setup Blacklist chain

```
$IPT -N BLACKLIST
$IPT -A BLACKLIST -j LOG --log-level $LOGLEVEL --log-prefix "BLACKLIST: "
$IPT -A BLACKLIST -j DROP
```

Populate Blacklist Ruleset using external file.

```
Use INPUT/FORWARD/OUTPUT chains to ensure ALL traffic is looked at
if [ "$BLACKLIST" != "" ]; then
      echo "Loading list of blocked hosts from $BLACKLIST"
      for host in `grep -v ^# $BLACKLIST | awk '{print $1}'`; do
```

```
            $IPT -A INPUT -s $host -j BLACKLIST
            $IPT -A INPUT -d $host -j BLACKLIST
            $IPT -A FORWARD -s $host -j BLACKLIST
            $IPT -A FORWARD -d $host -j BLACKLIST
            $IPT -A OUTPUT -s $host -j BLACKLIST
            $IPT -A OUTPUT -d $host -j BLACKLIST
            echo "Blacklisted: $host "
      done
fi
```

Allow IDENT Packets and log any packets with header "FW-IDENT:"

```
$IPT -A INPUT -p tcp -i $EXTIF --dport 113 -j LOG --log-level $LOGLEVEL --
log-prefix "FW-IDENT: "
```

Reject any IDENT packets with tcp-reset

```
$IPT -A INPUT -p tcp -i $EXTIF --dport 113 -j REJECT \
      --reject-with tcp-reset
```

Accept DNS traffic between External FW and External DNS/NTP server

```
$IPT -A INPUT -p udp -i $INTIF -s 39.2.1.49 -d 39.2.1.54 \
      --dport 53 -j ACCEPT
$IPT -A OUTPUT -p udp -o $INTIF -s 39.2.1.54 --sport 53 \
      -d 39.2.1.49 -j ACCEPT
```

Accept DNS traffic between External DNS/NTP server and ALL

```
$IPT -A FORWARD -p udp -i $INTIF -s 39.2.1.54 --sport 53 \
      --dport 53 -j ACCEPT
$IPT -A FORWARD -p udp -i $EXTIF --sport 53 -d 39.2.1.54 \
      --dport 53 -j ACCEPT
```

Accept HTTP/HTTPS traffic between ALL and Squid server

```
$IPT -A FORWARD -p tcp -i $EXTIF -d 39.2.1.55 --dport 80 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.55 --sport 80 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $EXTIF -d 39.2.1.55 --dport 443 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.55 --sport 443 \
      -j ACCEPT
```

Accept HTTPS traffic between ALL and WebMail server

```
$IPT -A FORWARD -p tcp -i $EXTIF -d 39.2.1.51 --dport 443 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.51 --sport 443 \
      -j ACCEPT
```

Accept SYSLOG traffic between Router and External Syslog server

```
$IPT -A FORWARD -p udp -i $EXTIF --sport 514 -d 39.2.1.62 \
      --dport 514 -j ACCEPT
```

Accept SMTP traffic between ALL and Mail Relay

```
$IPT -A FORWARD -p tcp -i $EXTIF -d 39.2.1.53 --dport 25 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.53 --sport 25 \
      -j ACCEPT
```

Accept NTP traffic between Router and Ext DNS/NTP server

```
$IPT -A FORWARD -p udp -i $EXTIF -s 39.2.1.45 -d 39.2.1.54 \
      --dport 123 -j ACCEPT
$IPT -A FORWARD -p udp -i $INTIF -s 39.2.1.54 --sport 123 \
      -d 39.2.1.45 -j ACCEPT
```

Accept NTP traffic between Ext FW and Ext DNS/NTP server

```
$IPT -A OUTPUT -p udp -o $INTIF -s 39.2.1.49 -d 39.2.1.54 \
      --dport 123 -j ACCEPT
$IPT -A INPUT -p udp -i $INTIF -s 39.2.1.54 --sport 123 \
      -d 39.2.1.49 -j ACCEPT
```

Accept SSH traffic between CIO/Admin Workstations and External Firewall

```
$IPT -A INPUT -p tcp -i $INTIF -s 192.168.24.5 -d 39.2.1.49 \
      --dport 22 -j ACCEPT
$IPT -A INPUT -p tcp -i $INTIF -s 192.168.24.4 -d 39.2.1.49 \
      --dport 22 -j ACCEPT
$IPT -A OUTPUT -p tcp -o $INTIF -s 39.2.1.49 --sport 22 \
      -d 192.168.24.5 -j ACCEPT
$IPT -A OUTPUT -p tcp -o $INTIF -s 39.2.1.49 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
```

Accept Zone Transfer between External DNS/NTP server and ISP DNS server

```
$IPT -A FORWARD -p tcp -i $EXTIF -s 39.0.1.20 --sport 53 \
      -d 39.2.1.54 --dport 53 -j ACCEPT
$IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.54 --sport 53 \
      -d 39.0.1.20 --dport 53 -j ACCEPT
```

LOG all packets that are dropped due to default FORWARD POLICY

```
$IPT -A FORWARD -j LOG --log-level $LOGLEVEL \
      --log-prefix "FORWARD DROP:"
```

LOG all packets that are dropped due to default INPUT POLICY

```
$IPT -A INPUT -j LOG --log-level $LOGLEVEL \
      --log-prefix "INPUT DROP:"
```

LOG all packets that are dropped due to default OUTPUT POLICY

```
$IPT -A OUTPUT -j LOG --log-level $LOGLEVEL \
      --log-prefix "OUTPUT DROP:"
```

Server Host Netfilter Firewalls

The firewall configuration file has been written as a traditional Red Hat init file found in /etc/init.d. This style would allow for the administrator to run one of the following commands to modify the firewall

```
[root@SERVER /]#service netfilter {start|stop|restart|debug|status|panic)
```

where

- Start – starts firewall
- Stop – stops firewall and opens up networking
- Restart – restarts the firewall (Uses Stop and Start above)
- Debug – sets the Debug flag and then starts firewall. This adds additional logging.

- ? Status – gives the ruleset for the firewall
- ? Panic – locks down the firewall by denying all traffic

This firewall should apply the rules in the following order:

- ? User permit rules in order of traffic volume
- ? Management permit rules
- ? Deny remainder of traffic

Setup global variables. Identify location of iptables command.

```
IPT=/usr/local/sbin/iptables
```

Define LOGLEVEL variable and set DEBUG variable

```
LOGLEVEL=notice
DEBUG=0
```

Define Interfaces – changes between test and production environments

```
EXTIF=eth0
```

Ensure iptables command is available or Error and exit

```
if [ ! -x $IPT ]; then
      echo "ERROR: Cannot locate iptables command."
      exit 0
fi
```

Ensure ipchains command is not running or Error and exit

```
if  /sbin/lsmod 2>/dev/null |grep -q ipchains ; then
      echo "ERROR: Cannot run both ipchains and iptables"
      exit 0
fi
```

Log to screen and syslog the start message

```
echo "Starting NetFilter Firewall"
logger -p local3.info NetFilter Firewall Started
```

Enable syn cookie support

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

Enable logging of Martians (oddball addresses).

```
echo 1 > /proc/sys/net/ipv4/conf/all/log_martians
```

Enable IP Forward support.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Disable response to ping

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
```

Disable response to Broadcasts (to keep from being SMURF amplifier)

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

Enable Dynamic IP Address support.

```
echo 1 > /proc/sys/net/ipv4/ip_dynaddr
```

Enable anti spoofing support.

```
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter
```

Disable source routing support.

```
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route
```

Disable icmp redirect support.

```
echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects
```

Flush,delete and zero existing chains

```
$IPT -t filter -F
$IPT -t nat -F
$IPT -t mangle -F
$IPT -X
$IPT -Z
```

Set up default policies for nat, mangle and filter tables.

```
$IPT -t nat -P PREROUTING ACCEPT
$IPT -t nat -P POSTROUTING ACCEPT
$IPT -t nat -P OUTPUT ACCEPT
$IPT -t mangle -P PREROUTING ACCEPT
$IPT -t mangle -P INPUT ACCEPT
$IPT -t mangle -P FORWARD ACCEPT
$IPT -t mangle -P OUTPUT ACCEPT
$IPT -t mangle -P POSTROUTING ACCEPT
$IPT -t filter -P FORWARD DROP
$IPT -t filter -P INPUT DROP
$IPT -t filter -P OUTPUT ACCEPT
```

NAT table module rules - This table is consulted when a packet that creates a new connection is encountered.

Mangle Module Rules

Rules to mangle TOS values of packets routed through the firewall that is based on RFC 1060/1349. This includes the following settings:

- **Minimize-Delay 16 (0x10)**
- **Maximize-Throughput 8 (0x08)**
- **Maximize-Reliability 4 (0x04)**
- **Minimize-Cost 2 (0x02)**
- **Normal-Service 0 (0x00)**

Set minimize-delay for SSH and DNS traffic

```
$IPT -t mangle -A PREROUTING -p tcp --dport 22 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p udp --dport 53 -j TOS --set-tos 16
```

Filter Module Rules

Setup debugging for tracking traffic flows. Activated if DEBUG variable is 1.

```
if [ $DEBUG = 1 ]; then
      echo "Extra Debug logging enabled"
```

Log all localhost traffic entering and leaving the LO interface

```
$IPT -A INPUT -i lo -j LOG --log-level $LOGLEVEL \
      --log-prefix "local-in: "
$IPT -A OUTPUT -o lo -j LOG --log-level $LOGLEVEL \
      --log-prefix "local-out: "
```

Log all DNS, NTP, SYSLOG, LDAPS and SSH traffic traversing the firewall.

```
$IPT -A INPUT -p udp --sport 53 --dport 53 -j LOG \
      --log-level $LOGLEVEL --log-prefix "DNS: "
$IPT -A INPUT -p udp --sport 123 --dport 123 -j LOG \
      --log-level $LOGLEVEL --log-prefix "NTP: "
$IPT -A INPUT -p udp --sport 514 --dport 514 -j LOG \
      --log-level $LOGLEVEL --log-prefix "SYSLOG: "
$IPT -A INPUT -p tcp --dport 636 -j LOG \
      --log-level $LOGLEVEL --log-prefix "LDAPS: "
$IPT -A INPUT -p tcp --dport 22 -j LOG \
      --log-level $LOGLEVEL --log-prefix "SSH: "
fi
```

Allow all localhost traffic entering and leaving the Loopback interface.

```
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT
```

Allow incoming traffic if is part of an previous connection

```
$IPT -A INPUT -p TCP -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Allow DNS traffic between LDAP server and Internal DNS/NTP server

```
$IPT -A OUTPUT -o $EXTIF -p udp -s 192.168.23.4 -d 192.168.23.6 \
      --dport 53 -j ACCEPT
$IPT -A INPUT -i $EXTIF -p udp -s 192.168.23.6 --sport 53 \
      -d 192.168.23.4 -j ACCEPT
```

Allow NTP traffic between LDAP server and Internal DNS/NTP server

```
$IPT -A OUTPUT -o $EXTIF -p udp -s 192.168.23.4 -d 192.168.23.6 \
      --dport 123 -j ACCEPT
$IPT -A INPUT -i $EXTIF -p udp -s 192.168.23.6 --sport 123 \
      -d 192.168.23.4 -j ACCEPT
```

Allow SYSLOG traffic between LDAP server and Syslog server

```
$IPT -A OUTPUT -o $EXTIF -p udp -s 192.168.23.4 -d 192.168.24.2 \
      --dport 514 -j ACCEPT
```

Allow LDAPS traffic into LDAP server

```
$IPT -A INPUT -i $EXTIF -p tcp -d 192.168.23.4 \
      --dport 636 -j ACCEPT
$IPT -A OUTPUT -o $EXTIF -p tcp -s 192.168.23.4 --sport 636 \
      -j ACCEPT
```

Allow SSH traffic between LDAP server and CIO/Admin workstation

```
$IPT -A INPUT -i $EXTIF -p tcp -s 192.168.24.4 -d 192.168.23.4 \
```

```
      --dport 22 -j ACCEPT
$IPT -A OUTPUT -o $EXTIF -p tcp -s 192.168.23.4 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A INPUT -i $EXTIF -p tcp -s 192.168.24.5 -d 192.168.23.4 \
      --dport 22 -j ACCEPT
$IPT -A OUTPUT -o $EXTIF -p tcp -s 192.168.23.4 --sport 22 \
      -d 192.168.24.5 –j ACCEPT
```

LOG all packets that are dropped due to default FORWARD POLICY

```
$IPT -A FORWARD -j LOG --log-level $LOGLEVEL \
      --log-prefix "FORWARD DROP:"
```

LOG all packets that are dropped due to default INPUT POLICY

```
$IPT -A INPUT -j LOG --log-level $LOGLEVEL \
      --log-prefix "INPUT DROP:"
```

Log to screen NetFilter stopping message

```
echo "NetFilter Firewall started successfully"
```

Tutorial – How to set up a policy for a NetFilter firewall with FreeS/Wan Gateway

Overview

This tutorial will show how to set up and deploy a NetFilter firewall with a FreeS/Wan VPN Gateway. This will be deployed on a quad-homed machine similar to the design for GIAC-Enterprises, but could be used for any multi-homed design as well.

Requirements

RedHat 9.0 Linux with 2.4.20-8 kernel from http://www.redhat.com
Freeswan-module-2.01_2.4.20_8-0.i386.rpm from http://www.freeswan.org
Freeswan-userland-2.01_2.4.20_8-0.i386.rpm from http://www.freeswan.org
iptables-1.2.8.tar.bz2 from http://www.netfilter.org

HostOS

Install a minimal Red Hat OS and Lock down it down. Several resources are available for this purpose including the following:

- Bastille-Linux (http://www.bastille-linux.org) – a set of hardening scripts available in RPM format.
- Hardening Linux Systems (http://www.linux-mag.com/2002-09/guru_01.html) - guidance in hardening linux systems from Linux Magazine.
- Red Hat Linux Security Guide (http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/security-guide/) - overview of security best practices from Red Hat.
- Linux Security HOWTO (http://tldp.org/HOWTO/Security-HOWTO/index.html) - a general overview of security issues that face the administrator of Linux systems.

Firewall Installation

This process uses the GNU C compiler to compile the code. Typically a firewall boxwith a minimum installation would not have a compiler on it. For this portion, a compiler was installed via RPM and removed once the software has been compiled and tested. This process could also be done on another machine and the appropriate binaries could be moved to the firewall.

Download the latest version of the firewall from http://www.netfilter.org. Be sure to check the gpg signature to verify the file hasn't been tampered with. You will need to download the file .sig file as well as coreteam-gpg-key.txt (gpg public key).

Next import the gpg key to your keyring

```
[root@INTERNAL opt]#gpg –import coreteam-gpg-key.txt
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: key CA9A8D5B: public key "Netfilter Core Team <coreteam@netfilter.org>"
imported
gpg: Total number processed: 1
gpg:               imported: 1
```

Then verify the signature by typing

```
[root@INTERNAL opt]#gpg –-verify iptables-1.2.8.tar.bz2.sig iptables-
1.2.8.tar.bz2
gpg: Signature made Sun 13 Apr 2003 11:33:42 AM EDT using DSA key ID CA9A8D5B
gpg: Good signature from "Netfilter Core Team <coreteam@netfilter.org>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 02AC E2A4 74DD 09D7 FD45  2E2E 35FA 89CC CA9A 8D58
```

This tells us that based on the gpg key we pulled from netfilter.org that the signature file for iptables-1.2.8.tar.bz2 is good. The WARNING that also prints out is just a reminder that the key used to verify the signature was not trusted (i.e. when imported the key didn't identified as a trusted key). This is fine since a user shouldn't trust a gpg key just because it was downloaded from a website that seemed correct. We are simply limiting our risk regarding this code by checking file against the signature and the gpg key since it would take a much larger effort to modify the iptables source, forge the gpg key, create the file signature and either upload all three to the netfilter website or spoof it.

Next we need to uncompress and untar the file

```
[root@INTERNAL opt]#bzip2 –d iptables-1.2.8.tar.bz2
[root@INTERNAL opt]#tar -xvf iptables-1.2.8.tar.bz2
```

Change into the iptables directory

```
[root@INTERNAL opt]#cd iptables-1.2.8
```

Run make to compile the binary

```
[root@EXTERNAL iptables-1.2.8]#make
```

Run make install to install the files into the appropriate directories

```
[root@EXTERNAL iptables-1.2.8]#make install
```

These files were placed in the following directories. This will help should you need to move them from another machine after compilation.

- /usr/local/sbin/
  - Iptables
  - iptables-save
  - iptables-restore
  - ip6tables
- /usr/local/man/man8/
  - iptables.8
  - iptables-restore.8
  - iptables-save.8
  - ip6tables.8
- /usr/local/lib/iptables/
  - libipt_ah.so
  - libipt_conntrack.so
  - libipt_dscp.so
  - libipt_ecn.so
  - libipt_esp.so
  - libipt_helper.so
  - libipt_icmp.so
  - libipt_iplimit.so
  - libipt_length.so
  - libipt_limit.so
  - libipt_mac.so
  - libipt_mark.so
  - libipt_multiport.so
  - libipt_owner.so
  - libipt_physdev.so
  - libipt_pkttype.so
  - libipt_rpc.so
  - libipt_standard.so
  - libipt_state.so
  - libipt_tcp.so
  - libipt_tcpmss.so
  - libipt_tos.so
  - libipt_ttl.so
  - libipt_udp.so
  - libipt_unclean.so
  - libipt_DNAT.so
  - libipt_DSCP.so
  - libipt_ECN.so
  - libipt_LOG.so
  - libipt_MARK.so
  - libipt_MASQUERADE.so
  - libipt_MIRROR.so
  - libipt_REDIRECT.so
  - libipt_REJECT.so
  - libipt_SAME.so
  - libipt_SNAT.so
  - libipt_TARPIT.so
  - libipt_TCPMSS.so

- libipt_TOS.so
- libipt_TTL.so
- libipt_ULOG.so
- libip6t_eui64.so
- libip6t_hl.so
- libip6t_icmpv6.so
- libip6t_length.so
- libip6t_limit.so
- libip6t_mac.so
- libip6t_mark.so
- libip6t_multiport.so
- libip6t_owner.so
- libip6t_standard.so
- libip6t_tcp.so
- libip6t_udp.so
- libip6t_HL.so
- libip6t_LOG.so
- libip6t_MARK.so

FreeSWAN Installation

Download proper version of freeswan modules from
ftp://ftp.xs4all.nl/pub/crypto/freeswan/binaries/RedHat-RPMs

Check signatures
While you are there, grab the RPM signing key

freeswan-rpmsign.asc

Login as root
Import this key into the RPM database:

```
[root@INTERNAL opt]#rpm --import freeswan-rpmsign.asc
```

Check the signatures on both RPMs using:

```
[root@INTERNAL opt]#rpm --checksig freeswan*.rpm
```

You should see output like:

```
freeswan-module-2.00_2.4.20_8-0.i386.rpm: pgp md5 OK
freeswan-userland-2.00_2.4.20_8-0.i386.rpm: pgp md5 OK
```

Install your RPMs with:

```
[root@INTERNAL opt]#rpm -ivh freeswan*.rpm
```

Then, start FreeS/WAN:

```
[root@INTERNAL opt]#service ipsec start
```

To check that you have a successful install, run:

```
[root@INTERNAL opt]#ipsec verify
```

You should see at least:

```
Checking your system to see if IPsec got installed and started correctly
Version check and ipsec on-path                                 [OK]
Checking for KLIPS support in kernel                            [OK]
Checking for RSA private key (/etc/ipsec.secrets)               [OK]
Checking that pluto is running                                  [OK]
```

If any of these first four checks fails, see their troubleshooting guide at http://www.freeswan.org/freeswan_trees/freeswan-2.00/doc/trouble.html#install.check

VPN Access Policy

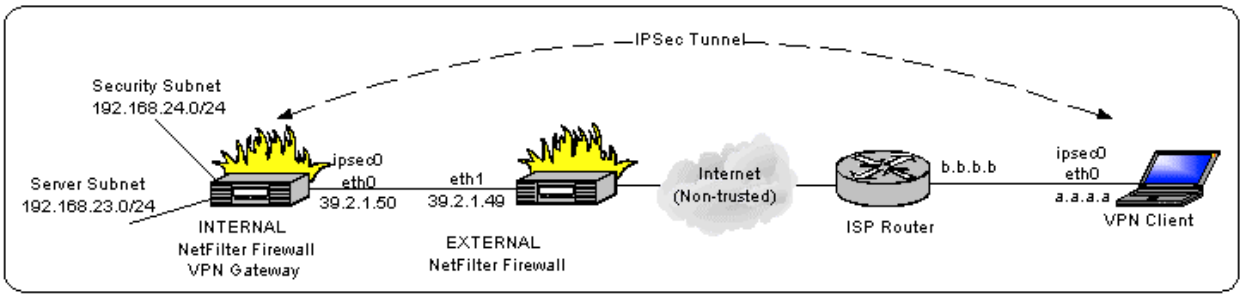
It was decided that only remote users coming from a known static IP would be able to access the corporate network. This policy was decided upon due to security concerns of unknown remote connections (i.e. via dialup or hotel high-speed access). All remote users will utilize high-speed Internet from their home office and the company will pay for each user to have a static IP. All users will use a custom firewall client that will only allow one connection to the Internet or to the Corporate VPN at one time. This will assist with keeping the remote users from becoming backdoor gateways into the internal network. In addition different tunnels, authenticated with different RSA keys, will be set up for users requiring access to the Server subnet and Security subnet which will control the users accessing the Security subnet.

Configuration

Next we need to set up the configuration files before we worry about the firewall. This is to ensure we configure it correctly to allow connections. By doing this first we take away the possibility that the firewall could cause problems with the gateway. I will be giving an overview of the VPN gateway configuration focusing on how its implementation will affect the firewall policy. This is not intended to be a tutorial on the FreeSWAN VPN configuration. Such tutorials and information can be found at:

- http://www.freeswan.org - FreeSWAN site which includes information on installation, configuration, interoperability issues, troubleshooting and much more.
- http://www.linuxjournal.com/article.php?sid=4772 - Using a bootable Linux distribution to install and run an IPSec-based VPN gateway with a firewall.
- http://www.ddj.com/documents/s=8213/ddj0306c/ - Using FreeSWAN to secure wireless networks.
- http://www.opentech.at/howtos/freeswan.html - Setting up an IPsec VPN using FreeS/WAN.

We will use the following architecture to connect our remote users to the corporate network.



The configurations that are being used for the VPN gateway and VPN clients can be found below.

Gateway IPsec.conf

On the Netfilter Firewall VPN Gateway the following file can be found in /etc/ipsec.conf.

Identifies that this file conforms to second version of ipsec.conf specification.

```
version     2.0
```

Sets the basic configuration section

```
config setup
```

Defines the interface on which the host will listen for IPsec connections

```
      interfaces="ipsec0=eth0"
```

Debug-logging controls – none for (almost) none and all for everything. Klipsdebug covers the kernel-interface daemon (KLIPS) and plutodebug covers IKE keying daemon (Pluto).

```
      klipsdebug=none
      plutodebug=none
```

Causes the gateway to drop an existing connection for a particular ID if another connection tries to connect with the same ID. This allows clients to reconnect should their connection drop and keeps the server from maintaining dead connections.

```
      uniqueids=yes
```

Sets the default connection section that has settings to use with all individual connections

```
conn %default
```

Defines how many times a connection can try to reestablish itself if lost. Zero indicates infinite retries.

```
      keyingtries=0
```

Defines how the machines must authenticate with each other. This states that RSA signatures will be used.

```
      authby=rsasig
```

Enables disablearrivalcheck which causes KLIPS to make sure that each packet entering the host from an IPsec tunnel has a correct source and destination IP address in its header.

```
      disablearrivalcheck=no
```

Sets the connection up for achan to the server subnet

```
conn achan_svr
```

Defines the IP address of our gateway. NOTE: the distinction between left and right is strictly up to the individual as long as the definition is the same on both the clients and gateway (i.e. left = gateway information and right = client information stays the same in each ipsec.conf used)

```
left=39.2.1.50
```

Defines the subnet the tunnel will allow access to. In this case it is our Server subnet.

```
leftsubnet=192.168.23.0/24
```

Defines the next hop from the gateway. In this case it is the IP of our external firewall.

```
leftnexthop=39.2.1.49
```

Sets the ID that is used for the gateways machine for this connection.

```
leftid=@secgw_svr.giac-enterprises.com
```

Identifies the RSA public key for the Gateway and was specifically generated for user use to authenticate to the Server subnet.

```
# RSA 2192 bits   INTERNAL_SVR   Sun May 13 19:14:57 2003
leftrsasigkey=……
```

Defines the IP address of our client. NOTE: Again note discussion of left and right above.

```
right=a.a.a.a
```

Defines the next hop from the client. In this case we will identify the ISPs router upstream from our client host.

```
rightnexthop=b.b.b.b
```

Sets the ID that is used for the client machine for this connection.

```
rightid=@achan.giac-enterprises.com
```

Identifies the RSA public key for Adam Chan's client which ties back to a secret key on his client.

```
# RSA 2192 bits   ACHAN   Sun May 13 17:10:44 2003
rightrsasigkey=……
```

Causes the connection definition to be added upon the start of the FreeS/WAN server.

```
auto=add
```

Sets the connection up for achan to the security subnet

```
conn achan_sec
```

Defines the IP address of our gateway

```
left=39.2.1.50
```

Defines the subnet the tunnel will allow access to. In this case it is our Security subnet.

```
leftsubnet=192.168.24.0/24
```

Defines the next hop from the gateway. In this case it is the IP of our external firewall.

```
leftnexthop=39.2.1.49
```

Sets the ID that is used for the gateways machine for this connection.

```
leftid=@secgw_sec.giac-enterprises.com
```

Identifies the RSA public key for the Gateway and was specifically generated for user use to authenticate to the Security subnet.

```
# RSA 2192 bits   INTERNAL_SEC   Sun May 13 19:14:57 2003
leftrsasigkey=……
```

Defines the IP address of our client.

```
right=a.a.a.a
```

Defines the next hop from the client. In this case we will identify the ISPs router upstream from our client host.

```
rightnexthop=b.b.b.b
```

Sets the ID that is used for the client machine for this connection.

```
rightid=@achan.giac-enterprises.com
```

Identifies the RSA public key for Adam Chan's client which ties back to a secret key on his client.

```
# RSA 2192 bits   ACHAN   Sun May 13 17:10:44 2003
rightrsasigkey=……
```

Causes the connection definition to be added upon the start of the FreeS/WAN server.

```
auto=add
```

Disables Opportunistic Encryption by denying the default connection descriptions from being added. May be a unique requirement for Red Hat 9.0.

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

VPN client IPsec.conf

Identifies that this file conforms to second version of ipsec.conf specification.

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

Sets the basic configuration section

```
config setup
```

Defines the interface on which the host will listen for IPsec connections

```
      interfaces="ipsec0=eth0"
```

Debug-logging controls – none for (almost) none and all for everything. Klipsdebug covers the kernel-interface daemon (KLIPS) and plutodebug covers IKE keying daemon (Pluto).

```
klipsdebug=none
plutodebug=none
```

Causes the gateway to drop an existing connection for a particular ID if another connection tries to connect with the same ID. This allows clients to reconnect should their connection drop and keeps the server from maintaining dead connections.

```
uniqueids=yes
```

Sets the default connection section that has settings to use with all individual connections

```
conn %default
```

Defines how many times a connection can try to reestablish itself if lost. Zero indicates infinite retries.

```
keyingtries=0
```

Defines how the machines must authenticate with each other. This states that RSA signatures will be used.

```
authby=rsasig
```

Enables disablearrivalcheck which causes KLIPS to make sure that each packet entering the host from an IPsec tunnel has a correct source and destination IP address in its header.

```
disablearrivalcheck=no
```

Sets the connection up for achan to the server subnet

```
conn achan_svr
```

Defines the IP address of our client.

```
right=a.a.a.a
```

Defines the next hop from the client. In this case we will identify the ISPs router upstream from our client host.

```
rightnexthop=b.b.b.b
```

Sets the ID that is used for the client machine for this connection.

```
rightid=@achan.giac-enterprises.com
```

Identifies the RSA public key for Adam Chan's client which ties back to a secret key on his client.

```
# RSA 2192 bits   ACHAN   Sun May 13 17:10:44 2003
rightrsasigkey=……
```

Defines the IP address of our gateway

```
left=39.2.1.50
```

Defines the next hop from the gateway. In this case it is the IP of our external firewall.

```
leftnexthop=39.2.1.49
```

Defines the subnet the tunnel will allow access to. In this case it is our Server subnet.

```
leftsubnet=192.168.23.0/24
```

Sets the ID that is used for the gateways machine for this connection.

```
leftid=@secgw_svr.giac-enterprises.com
```

Identifies the RSA public key for the Gateway and was specifically generated for user use to authenticate to the Server subnet.

```
      # RSA 2192 bits   INTERNAL_SVR   Sun May 13 19:14:57 2003
      leftrsasigkey=……
```

Causes the connection definition to be added upon the start of the FreeS/WAN server.

```
      auto=add
```

Sets the connection up for achan to the security subnet

```
conn achan_sec
```

Defines the IP address of our client.

```
      right=a.a.a.a
```

Defines the next hop from the client. In this case we will identify the ISPs router upstream from our client host.

```
      rightnexthop=b.b.b.b
```

Sets the ID that is used for the client machine for this connection.

```
      rightid=@achan.giac-enterprises.com
```

Identifies the RSA public key for Adam Chan's client which ties back to a secret key on his client.

```
      # RSA 2192 bits   ACHAN   Sun May 13 17:10:44 2003
      rightrsasigkey=……
```

Defines the IP address of our gateway

```
      left=39.2.1.50
```

Defines the next hop from the gateway. In this case it is the IP of our external firewall.

```
      leftnexthop=39.2.1.49
```

Defines the subnet the tunnel will allow access to. In this case it is our Security subnet.

```
      leftsubnet=192.168.24.0/24
```

Sets the ID that is used for the gateways machine for this connection.

```
      leftid=@secgw_sec.giac-enterprises.com
```

Identifies the RSA public key for the Gateway and was specifically generated for user use to authenticate to the Security subnet.

```
      # RSA 2192 bits   INTERNAL_SEC   Sun May 13 19:14:57 2003
      leftrsasigkey=……
```

Causes the connection definition to be added upon the start of the FreeS/WAN server.

```
      auto=add
```

Disables Opportunistic Encryption by denying the default connection descriptions from being added. May be a unique requirement for Red Hat 9.0.

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

```
      auto=ignore
conn packetdefault
      auto=ignore
```

Once the /etc/ipsec.conf files have been modified then the ipsec daemon has to be restarted.

```
[root@INTERNAL /]#service ipsec restart

Verify that the FreeS/WAN has been set up correctly by doing the following:

[root@INTERNAL /]#ipsec auto --up achan_sec
104 "achan_sec" #1: STATE_MAIN_I1: initiate
106 "achan_sec" #1: STATE_MAIN_I2: sent MI2, expecting MR2
108 "achan_sec" #1: STATE_MAIN_I3: sent MI3, expecting MR3
004 "achan_sec" #1: STATE_MAIN_I4: ISAKMP SA established
112 "achan_sec" #2: STATE_QUICK_I1: initiate
004 "achan_sec" #2: STATE_QUICK_I2: sent QI2, IPsec SA established

The users will have a custom firewall client that will control the activation
of all ipsec communication.
```

Netfilter/IPTables Syntax

The netfilter script on this gateway uses the following syntax in determining a rule:

```
[root@INTERNAL /]#iptables [-t table] [-A chain] [-i interface][-p protocol]
[-s source] [--sport source-port] [-d destination] [--dport destination-port]
[-j target]
```

[-t table] – Defines the packet matching table that is affected

- filter – Default table if no –t setting defined. Contains 3 built-in chains.
  - INPUT – for packets coming to machine itself
  - FORWARD – for packets being routed through the machine
  - OUTPUT – for packets leaving the machine itself
- nat – Consulted when a packet that new connection is received. Contains 3 built-in chains.
  - PREROUTING – for altering packets when received
  - OUTPUT – for altering locally-generated packets leaving machine
  - POSTROUTING – for altering packets before sent out
- mangle – Used for packet alteration in special ways. Contains 5 built-in chains
  - INPUT – for altering packets coming to machine itself
  - FORWARD – for altering packets being routed through the machine
  - OUTPUT – for altering packets leaving the machine itself
  - PREROUTING – for altering packets when received
  - POSTROUTING – for altering packets before sent out

[-A chain] - Defines the chain that the rule affects. Could be a custom chain or one of the above pre-defined ones.

[-i interface] – Defines the interface the rule should look for matching packets on.

[-p protocol] – Defines the protocol of the packet to check. Can uses the IP protocol number or specified name from /etc/protocols.

[-s source] – Defines the source IP address the rule should match on.

[--sport source-port] – Defines the source port address the rule should match on. Will only be used when –p is tcp or udp.

[-d destination] – Defines the destination IP address the rule should match on.

[--dport destination-port] – Defines the destination port address the rule should match on. Will only be used when –p is tcp or udp.

[-j target] – Selects the target to send the packet to when it matches the rule.

- ? ACCEPT – Allow the packet through
- ? DROP – Drop the packet to the bin bucket
- ? REJECT – Deny the packet and send an ICMP error message back to the source
- ? LOG – Log matching packets to kernel log
  - o --log-level – use to define level of logging
  - o --log-prefix – defines a prefix attached to each log record

Netfilter script

The firewall configuration file has been written as a traditional Red Hat init file found in /etc/init.d. This style would allow for the administrator to run one of the following commands to modify the firewall

```
[root@INTERNAL /]#service netfilter {start|stop|restart|debug|status|panic)
```

where

- ? Start – starts firewall
- ? Stop – stops firewall and opens up networking
- ? Restart – restarts the firewall (Uses Stop and Start above)
- ? Debug – sets the Debug flag and then starts firewall. This adds additional logging.
- ? Status – gives the ruleset for the firewall
- ? Panic – locks down the firewall by denying all traffic

This firewall should apply the rules in the following order:

- ? Anti-spoofing filters and Blocked private addresses
- ? User permit rules
- ? Management permit rules
- ? Deny and Log suspicious traffic (including Blacklist from below)
- ? Deny remainder of traffic

Setup global variables. Identify location of iptables command.

```
IPT=/usr/local/sbin/iptables
```

Define LOGLEVEL variable and set DEBUG variable

```
LOGLEVEL=notice
DEBUG=0
```

Define Interfaces – changes between test and production environments

```
EXTIF=eth0
SRVIF=vmnet1
SECIF=vmnet2
WEBIF=vmnet3
```

Ensure iptables command is available or Error and exit

```
if [ ! -x $IPT ]; then
      echo "ERROR: Cannot locate iptables command."
      exit 0
fi
```

Ensure ipchains command is not running or Error and exit

```
if  /sbin/lsmod 2>/dev/null |grep -q ipchains ; then
      echo "ERROR: Cannot run both ipchains and iptables"
      exit 0
fi
```

Log to screen and syslog the start message.

```
echo "Starting NetFilter Firewall"
logger -p local3.info NetFilter Firewall Started
```

Enable syn cookie support.

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

Enable logging of Martians (oddball addresses).

```
echo 1 > /proc/sys/net/ipv4/conf/all/log_martians
```

Enable IP Forward support.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Disable response to ping.

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
```

Disable response to Broadcasts (to keep from being SMURF amplifier)

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

Enable Dynamic IP Address support.

```
echo 1 > /proc/sys/net/ipv4/ip_dynaddr
```

Enable anti spoofing support.

```
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter
```

Disable source routing support.

```
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route
```

Disable icmp redirect support.

```
echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects
```

Flush,delete and zero existing chains

```
$IPT -t filter -F
$IPT -t nat -F
$IPT -t mangle -F
$IPT -X
$IPT -Z
```

Set up default policies for nat, mangle and filter tables.

```
$IPT -t nat -P PREROUTING ACCEPT
$IPT -t nat -P POSTROUTING ACCEPT
$IPT -t nat -P OUTPUT ACCEPT

$IPT -t mangle -P PREROUTING ACCEPT
$IPT -t mangle -P INPUT ACCEPT
$IPT -t mangle -P FORWARD ACCEPT
$IPT -t mangle -P OUTPUT ACCEPT
$IPT -t mangle -P POSTROUTING ACCEPT

$IPT -t filter -P FORWARD DROP
$IPT -t filter -P INPUT DROP
$IPT -t filter -P OUTPUT DROP
```

NAT table module rules - This table is consulted when a packet that creates a new connection is encountered. Use PREROUTING to drop unwanted new traffic before passing on to FORWARD or INPUT Filters.

Mangle Module Rules

Rules to mangle TOS values of packets routed through the firewall that is based on RFC 1060/1349. This includes the following settings:

- **Minimize-Delay 16 (0x10)**
- **Maximize-Throughput 8 (0x08)**
- **Maximize-Reliability 4 (0x04)**
- **Minimize-Cost 2 (0x02)**
- **Normal-Service 0 (0x00)**

Set minimize-delay for SSH, SMTP and DNS traffic and set maximize-throughput for HTTP traffic.

```
$IPT -t mangle -A PREROUTING -p tcp --dport 22 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 25 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p udp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 80 -j TOS --set-tos 8
```

Filter Module Rules

Setup debugging for tracking traffic flows. Activated if DEBUG variable is 1.

```
if [ $DEBUG = 1 ]; then
      echo "Extra Debug logging enabled"
```

Log all localhost traffic entering and leaving the LO interface

```
      $IPT -A INPUT -i lo -j LOG --log-level $LOGLEVEL \
            --log-prefix "local-in: "
      $IPT -A OUTPUT -o lo -j LOG --log-level $LOGLEVEL \
            --log-prefix "local-out: "
```

Log all Encapsulated Security Payload (ESP) traffic entering and leaving the external interface.

```
      $IPT -A INPUT -p 50 -i $EXTIF -j LOG --log-level $LOGLEVEL \
            --log-prefix "ESP-in: "
      $IPT -A OUTPUT -p 50 -o $EXTIF -j LOG --log-level $LOGLEVEL \
            --log-prefix "ESP-out: "
```

Log all Internet Key Exchange (IKE) traffic entering and leaving the external interface.

```
      $IPT -A INPUT -p udp -i $EXTIF --sport 500 --dport 500 -j LOG \
            --log-level $LOGLEVEL --log-prefix "IKE-in: "
      $IPT -A OUTPUT -p udp -o $EXTIF --sport 500 --dport 500 -j LOG \
            --log-level $LOGLEVEL --log-prefix "IKE-out: "
```

Log all traffic going to and from any IPsec interface (tunnel)

```
      $IPT –A FORWARD –I ipsec+ -j LOG –log-level $LOGLEVEL \
            --log-prefix "IPSec: "
```

Log all DNS, NTP, SYSLOG, HTTP, HTTPS, SMTP, IMAP, MySQL, LDAPS and SSH traffic traversing the FORWARD chain of the firewall.

```
      $IPT -A FORWARD -p udp --sport 53 --dport 53 -j LOG \
            --log-level $LOGLEVEL --log-prefix "DNS: "
      $IPT -A FORWARD -p udp --sport 123 --dport 123 -j LOG \
            --log-level $LOGLEVEL --log-prefix "NTP: "
      $IPT -A FORWARD -p udp --sport 514 --dport 514 -j LOG \
            --log-level $LOGLEVEL --log-prefix "SYSLOG: "
      $IPT -A FORWARD -p tcp --dport 80 -j LOG \
            --log-level $LOGLEVEL --log-prefix "HTTP: "
      $IPT -A FORWARD -p tcp --dport 443 -j LOG \
            --log-level $LOGLEVEL --log-prefix "HTTPS: "
      $IPT -A FORWARD -p tcp --dport 25 -j LOG \
            --log-level $LOGLEVEL --log-prefix "SMTP: "
      $IPT -A FORWARD -p tcp --dport 143 -j LOG \
            --log-level $LOGLEVEL --log-prefix "IMAP: "
      $IPT -A FORWARD -p tcp --dport 3306 -j LOG \
            --log-level $LOGLEVEL --log-prefix "MySQL: "
      $IPT -A FORWARD -p tcp --dport 636 -j LOG \
            --log-level $LOGLEVEL --log-prefix "LDAPS: "
      $IPT -A FORWARD -p tcp --dport 22 -j LOG \
            --log-level $LOGLEVEL --log-prefix "SSH: "
fi
```

Allow all localhost traffic entering and leaving the Loopback interface.

```
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT
```

Allow IPsec into external interface. Accept IKE Negotiations with source port and destination port of 500.

```
$IPT -A INPUT -p udp -i $EXTIF --sport 500 --dport 500 -j ACCEPT
$IPT -A OUTPUT -p udp -o $EXTIF --sport 500 --dport 500 -j ACCEPT
```

Accept ESP encryption and authentication traffic

```
$IPT -A INPUT -p 50 -i $EXTIF -j ACCEPT
$IPT -A OUTPUT -p 50 -o $EXTIF -j ACCEPT
```

Accept any forwarded traffic to or from an IPSec interface (tunnel)

```
$IPT –A FORWARD –i ipsec+ -j ACCEPT
$IPT –A FORWARD –o ipsec+ -j ACCEPT
```

Setup new ICMP traffic chain

```
$IPT -N ICMP_TRAFFIC
```

Log ICMP traffic by type and drop.

```
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 0 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Echo Reply: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 8 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Echo Request: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 3 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Dest Unreachable: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 4 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Source Quench: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 5 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Redirect: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 9 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Router Advertisement: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 10 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Router Selection: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 11 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Time Exceeded: "
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 30 -j LOG --log-level $LOGLEVEL \
      --log-prefix "ICMP-Traceroute: "
$IPT -A ICMP_TRAFFIC -j DROP
```

Push all ICMP traffic to ICMP chain

```
$IPT -A INPUT -p icmp -j ICMP_TRAFFIC
$IPT -A FORWARD -p icmp -j ICMP_TRAFFIC
$IPT -A OUTPUT -p icmp -j ICMP_TRAFFIC
```

Accept DNS traffic between Security subnet and Internal DNS/NTP server

```
$IPT -A FORWARD -i $SECIF -p udp -s 192.168.24.0/24 -d 192.168.23.6 \
      --dport 53 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.6 -d 192.168.24.0/24 \
      --dport 53 -j ACCEPT
```

Accept DNS traffic between Internal DNS/NTP server & External DNS/NTP server

```
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.6 --sport 53 \
      -d 39.2.1.54 --dport 53 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p udp -s 39.2.1.54 --sport 53 \
      -d 192.168.23.6 --dport 53 -j ACCEPT
```

Accept NTP traffic between Security subnet and Internal DNS/NTP server

```
$IPT -A FORWARD -i $SECIF -p udp -s 192.168.24.0/24 -d 192.168.23.6 \
      --dport 123 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.6 –sport 123 \
      -d 192.168.24.0/24 -j ACCEPT
```

Accept NTP traffic between Webserver and Internal DNS/NTP server

```
$IPT -A FORWARD -i $WEBIF -p udp -s 192.168.25.2 -d 192.168.23.6
      --dport 123 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.6 –sport123 \
      -d 192.168.25.2 -j ACCEPT
```

Accept NTP traffic between Internal FW/VPN GW and Internal DNS/NTP server

```
$IPT -A INPUT -i $SRVIF -p udp -s 192.168.23.6 –sport 123 \
      –d 192.168.23.1 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p udp -s 192.168.23.1 –d 192.168.23.6 \
      --dport 123 -j ACCEPT
```

Accept SYSLOG traffic between syslog server and Server subnet

```
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.0/24 --sport 514 \
      -d 192.168.24.2 --dport 514 -j ACCEPT
```

Accept SYSLOG traffic between syslog server and Webserver

```
$IPT -A FORWARD -i $WEBIF -p udp -s 192.168.25.2 --sport 514 \
      -d 192.168.24.2 --dport 514 -j ACCEPT
```

Accept SYSLOG traffic between Internal FW/VPN GW and syslog server

```
$IPT -A OUTPUT -o $SECIF -p udp --sport 514 -d 192.168.24.2 \
      --dport 514 -j ACCEPT
```

Accept LDAPS traffic between VPN users and LDAP server

```
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.4 --sport 636 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.4 --dport 636 -j ACCEPT
```

Accept HTTP/HTTPS traffic between VPN users and Intranet server

```
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.5 --sport 80 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.5 --dport 80 -j ACCEPT
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.5 --sport 443 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.5 --dport 443 -j ACCEPT
```

Accept SMTP/IMAP traffic between VPN users and Mail server

```
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.3 --sport 143 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.3 --dport 143 -j ACCEPT
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.3 --sport 25 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.3 --dport 25 -j ACCEPT
```

Accept MySQL traffic between VPN users and Database server

```
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.2 --sport 3306 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.2 --dport 3306 -j ACCEPT
```

Accept NTP/DNS traffic between VPN users and Internal NTP/DNS server

```
$IPT -A INPUT -i $SRVIF -p udp -s 192.168.23.6 --sport 53 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p udp -d 192.168.23.6 --dport 53 -j ACCEPT
$IPT -A INPUT -i $SRVIF -p udp -s 192.168.23.6 --sport 123 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p udp -d 192.168.23.6 --dport 123 -j ACCEPT
```

Accept SSH traffic between VPN users and CIO/Admin workstation

```
$IPT -A INPUT -i $SECIF -p tcp -s 192.168.24.4 –-sport 22 -j ACCEPT
$IPT -A OUTPUT -o $SECIF -p tcp -d 192.168.24.4 --dport 22 -j ACCEPT
$IPT -A INPUT -i $SECIF -p tcp -s 192.168.24.5 --sport 22 -j ACCEPT
$IPT -A OUTPUT -o $SECIF -p tcp -d 192.168.24.5 --dport 22 -j ACCEPT
LDAPS traffic between webserver and LDAP server
```

```
$IPT -A FORWARD -i $WEBIF -p tcp -s 192.168.25.2 \
      -d 192.168.23.4 --dport 636 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.4 --sport 636 \
      -d 192.168.25.2 -j ACCEPT
```

Accept LDAPS traffic between webserver and LDAP server

```
$IPT -A FORWARD -i $EXTIF -p tcp -s 192.168.25.2 \
      -d 192.168.23.4 --dport 636 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.4 --sport 636 \
      -d 192.168.25.2 -j ACCEPT
```

Accept MySQL traffic between webserver and database server

```
$IPT -A FORWARD -i $WEBIF -p tcp -s 192.168.25.2 \
      -d 192.168.23.2 --dport 3306 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.2 --sport 3306 \
      -d 192.168.25.2 -j ACCEPT
```

Accept SMTP traffic between Mail server and Mail Relay

```
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.53 --sport 25 \
      -d 192.168.24.3 --dport 25 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.24.3 --sport 25 \
      -d 39.2.1.53 --dport 25 -j ACCEPT
```

Accept LDAPS traffic between WebMail server and LDAP server

```
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 \
      -d 192.168.23.4 --dport 636 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.4 --sport 636 \
      -d 39.2.1.51 -j ACCEPT
```

Accept SMTP/IMAP traffic between WebMail server and Mailserver

```
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 --sport 25 \
      -d 192.168.24.3 --dport 25 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.24.3 --sport 25 \
      -d 39.2.1.51 --dport 25 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 \
      -d 192.168.24.3 --dport 143 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.24.3 --sport 143 \
      -d 39.2.1.51 -j ACCEPT
```

Accept SSH traffic between CIO/Admin workstations and Server subnet

```
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 192.168.23.0/24 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.0/24 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 192.168.23.0/24 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.0/24 --sport 22 \
      -d 192.168.24.5 -j ACCEPT
```

Accept SSH traffic between CIO/Admin workstations and webserver

```
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 192.168.25.2 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $WEBIF -p tcp -s 192.168.25.2 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 192.168.25.2 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $WEBIF -p tcp -s 192.168.25.2 --sport 22 \
```

```
      -d 192.168.24.5 -j ACCEPT
```

Accept SSH traffic between CIO/Admin workstations and Internal FW/VPN GW

```
$IPT -A INPUT -i $SECIF -p tcp -s 192.168.24.4 \
      -d 192.168.24.1 --dport 22 -j ACCEPT
$IPT -A OUTPUT -o $SECIF -p tcp -s 192.168.24.1 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A INPUT -i $SECIF -p tcp -s 192.168.24.5 \
      -d 192.168.24.1 --dport 22 -j ACCEPT
$IPT -A OUTPUT -o $SECIF -p tcp -s 192.168.24.1 --sport 22 \
      -d 192.168.24.5 -j ACCEPT
```

Accept SSH traffic between CIO/Admin workstations and WebMail server

```
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.51 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.51 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 --sport 22 \
      -d 192.168.24.5 -j ACCEPT
```

Accept SSH traffic between CIO/Admin workstations and Squid server

```
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.55 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.55 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 22 \
      -d 192.168.24.5 -j ACCEPT
```

Accept SSH traffic between CIO/Admin workstations and Mail Relay server

```
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.53 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.53 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.53 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.53 --sport 22 \
      -d 192.168.24.5 -j ACCEPT
```

Accept SSH traffic between CIO/Admin workstations & External DNS/NTP server

```
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.54 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.54 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.54 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.54 --sport 22 \
      -d 192.168.24.5 -j ACCEPT
```

Accept SSH traffic between CIO/Admin workstations and External FW

```
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.49 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.49 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
```

```
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.49 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.49 --sport 22 \
      -d 192.168.24.5 -j ACCEPT
```

Accept HTTP/HTTPS traffic between CIO/Admin workstations and proxy server

```
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.55 --dport 8080 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.55 –dport 8080 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 8080 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 8080 \
      -d 192.168.24.5 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.55 --dport 8080 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.55 --dport 8080 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 8080 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 8080 \
      -d 192.168.24.5 -j ACCEPT
```

LOG all packets that are dropped due to default FORWARD POLICY

```
$IPT -A FORWARD -j LOG --log-level $LOGLEVEL \
      --log-prefix "FORWARD DROP:"
```

LOG all packets that are dropped due to default INPUT POLICY

```
$IPT -A INPUT -j LOG --log-level $LOGLEVEL \
      --log-prefix "INPUT DROP:"
```

LOG all packets that are dropped due to default OUTPUT POLICY

```
$IPT -A OUTPUT -j LOG --log-level $LOGLEVEL \
      --log-prefix "OUTPUT DROP:"
```

Log to screen NetFilter stopping message

```
echo "NetFilter Firewall started successfully"
```

Adding this netfilter script finishes up the firewall policy for our Gateway.

Firewall Policy Verification

The following is a technical audit of GIAC-enterprises primary firewall and will be performed to verify that the policies defined above have actually been implemented. GIAC Enterprises Internal Netfilter/VPN Gateway firewall will be audited and both the External and all Host-based firewalls will be audited in a similar fashion.

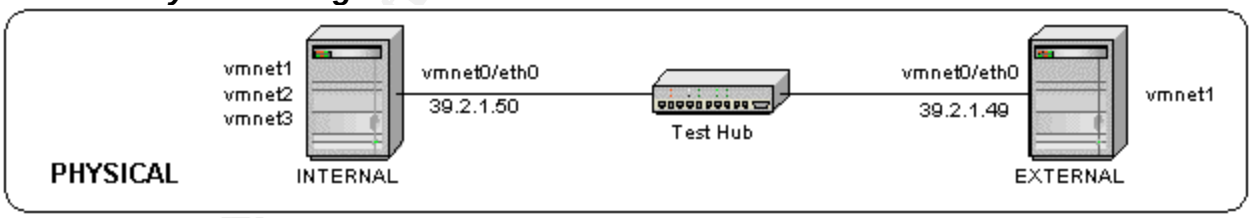
Test Network

Using two computers running Red Hat 9.0, VMware 4.0, and a hub Adam will set up a test network that will simulate our production network. This will be useful when evaluating a change in server configuration, policy change or software upgrade. Although this configuration does not mirror the production network, it is of considerable benefit given its expense.

VMware

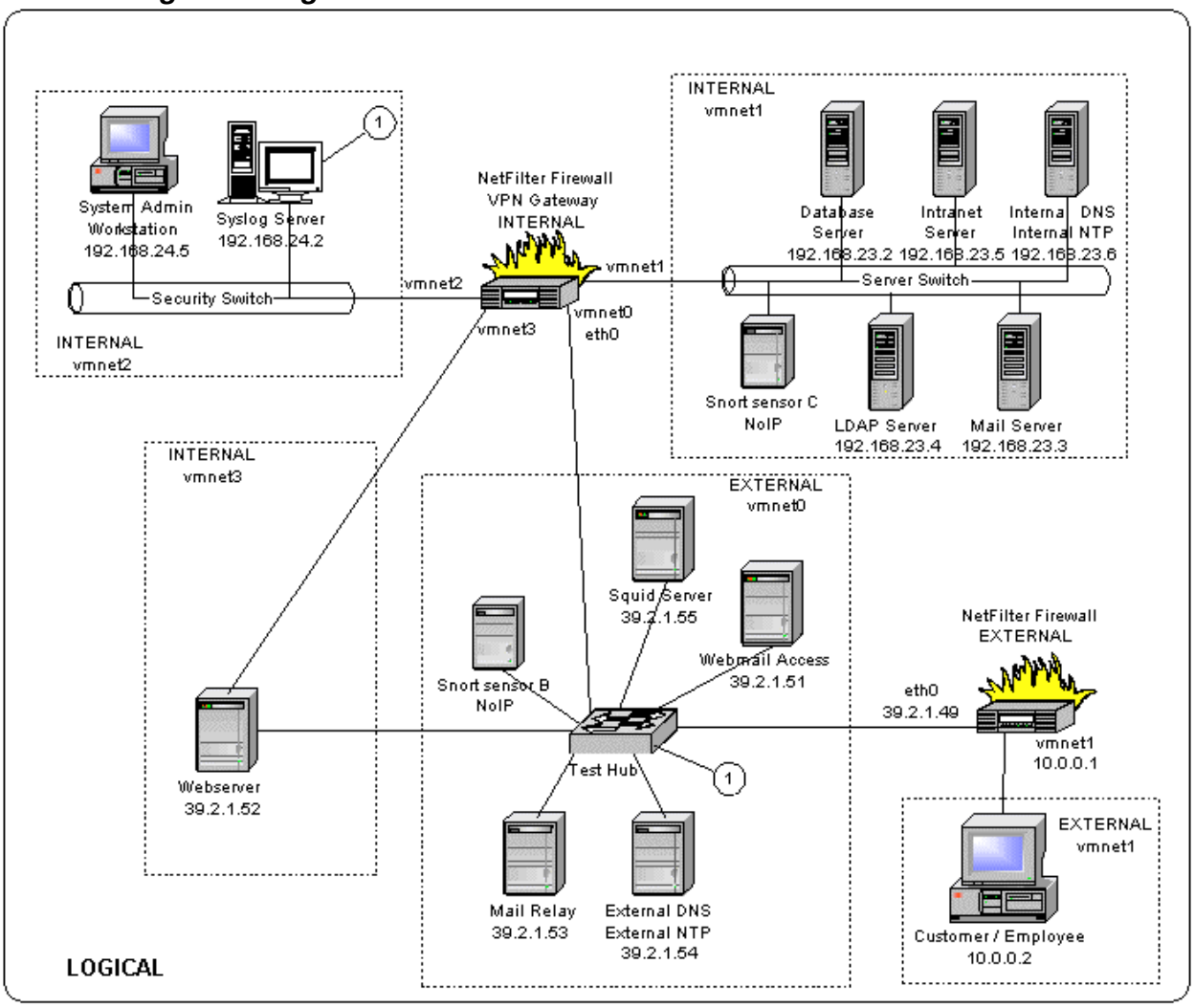
VMware Workstation (http://www.vmware.com) is virtual machine software that runs multiple operating systems simultaneously on a single computer. These operating systems are isolated in secure virtual machines and can co-exist on a single piece of hardware. It uses a virtualization layer that maps the virtual machine's resources to the physical hardware resources, which means each virtual machine has its own CPU, memory, disks, I/O devices, etc. We will use a combination of virtual hubs, called vmnets, and bridges to connect the servers together. Like a physical hub, a virtual hub lets you connect other networking components together a separate network. A bridge lets you connect your virtual machine to the LAN used by your host computer by mapping the virtual network adapter in your virtual machine to the physical Ethernet adapter on your host computer.

VMware Physical Design



Our firewalls will sit on the INTERNAL and EXTERNAL hosts with the other servers divided up among the many virtual hubs.

VMware Logical Design



Network List

Network Simulated	Name	Hosts
Internet	EXTERNAL/vmnet1	1
39.2.1.48/28	Test hub and EXTERNAL/vmnet0	6
192.168.23.0/24	INTERNAL/vmnet1	6
192.168.24.0/24	INTERNAL/vmnet2	2
192.168.25.0/24	INTERNAL/vmnet3	1

Device/Server List

Device/Server	Simulated Connection	Actual Connection	Interface Name	IP address
External Firewall	External	Internal Hub (vmnet1)	vmnet1	192.168.1.1/24
	Internal	Test Hub	vmnet0/eth0	39.2.1.49/28
External DNS /	DMZ switch	Internal Hub (vmnet0)	eth0	39.2.1.54/28

NTP server				
Mail Relay	DMZ switch	Internal Hub (vmnet0)	eth0	39.2.1.53/28
Squid server	DMZ switch	Internal Hub (vmnet0)	eth0	39.2.1.55/28
Internal Firewall/ VPN GW	DMZ Hub	Test Hub	vmnet0/eth0	39.2.1.50/28
	Server switch	Internal Hub (vmnet1)	vmnet1	192.168.23.1/24
	Security switch	Internal Hub (vmnet2)	vmnet2	192.168.24.1/24
	Web server	Internal Hub (vmnet3)	vmnet3	192.168.25.1/24
Webserver	DMZ switch	Test Hub	eth0	39.2.1.52/28
	Internal Firewall	Internal Hub (vmnet3)	eth1	192.168.25.2/24
Webmail server	DMZ switch	Internal Hub (vmnet0)	eth0	39.2.1.51/28
Database server	Server switch	Internal Hub (vmnet1)	eth0	192.168.23.2/24
Mail server	Server switch	Internal Hub (vmnet1)	eth0	192.168.23.3/24
LDAP server	Server switch	Internal Hub (vmnet1)	eth0	192.168.23.4/24
Intranet server	Server switch	Internal Hub (vmnet1)	eth0	192.168.23.5/24
Internal DNS / NTP server	Server switch	Internal Hub (vmnet1)	eth0	192.168.23.6/24
Syslog Server	Security switch	Internal Hub (vmnet2)	eth0	192.168.24.2/24
	DMZ switch	Test Hub	eth1	No IP
IDS Sensor B	DMZ switch	Internal Hub (vmnet0)	eth1	No IP
IDS Sensor C	Server switch	Internal Hub (vmnet1)	eth1	No IP
CIO workstation	Security switch	Internal Hub (vmnet2)	eth0	192.168.24.4/24

Adam can test any set of servers that is required without having to bring up the entire network. For example, if he was testing the interaction across the IPsec VPN then the external test host and an internal set of hosts would be brought up to test the total configuration.

Plan the audit

Since we will have a complete test network to work on, the production network will require only minimal downtime. In addition any changes to the production network (i.e. upgrades, patches or configuration changes) can now be done on the test network and reviewed before applying to the production network. This will further reduce downtime and other troubleshooting issues. Once testing has been completed the production network can be adjusted during scheduled maintenance periods during non-peak hours. For the initial audit we will first perform the tests on the test network and then perform the same tests on the production network to ensure the implementations are as identical as possible. In addition a monthly review is scheduled to review the state of the network including the following:

- ? Any software changes, upgrades, or patches
- ? Any firewall ruleset changes
- ? Any new vulnerabilities or exploits for software/hardware used in our network

Cost and Effort

Since we will be using our test environment to audit the firewall policy and this test environment is required for troubleshooting and upgrading purposes then no additional equipment would be

necessary. Adams time used during the audit should be considered as well as time spent mitigating problems.

Description	Estimated Hours	Cost (USD)
Time spent performing audit	10 @ $200 per hour	$2,000
Time spent mitigating problems	3 @ $200 per hour	$600
Total		$2,600

Technical approach

For our firewall audit we came up with a set of requirements and then obtained the tools necessary to test these requirements. This is a very important step in the process as an audit's success is based on whether these requirements are met or not. A full firewall audit would cover OS security hardening and Authentication/Encryption issues (i.e. VPNs, remote management, etc.), but this section will cover only the verification of the firewall policy.

For this audit our requirements are as follows:

- **Firewall Host**
  - Physical Access – Review operational plans regarding physical access.
  - OS security – Verify OS security using Nessus vulnerability scanner.
  - Remote management – Review operational plans regarding remote management.
- **Firewall Rulebase**
  - TCP Filtering – Verify which TCP ports of your firewall are open and match our initial rulebase from above. This includes ports open to the firewall as well as through the firewall.
  - UDP Filtering - Verify which UDP ports of your firewall are open and match our initial rulebase from above. This includes ports open to the firewall as well as through the firewall.

Conduct the audit

The audit will utilize the test network. In order to test just the internal firewall, I will configure the external firewall so it will route all packets. This will allow me to test the internal firewall from all four interfaces.

Tools required

NMAP 3.30 (nmap-3.30-1.i386.rpm) from http://www.insecure.org/nmap/index.html
NESSUS 2.0.7 from http://www.nessus.org
TCPDump 3.7.2 from http://www.tcpdump.org

Firewall Host

Although this audit was intended to verify the firewall policy, no firewall audit could be complete without also looking at the following components:

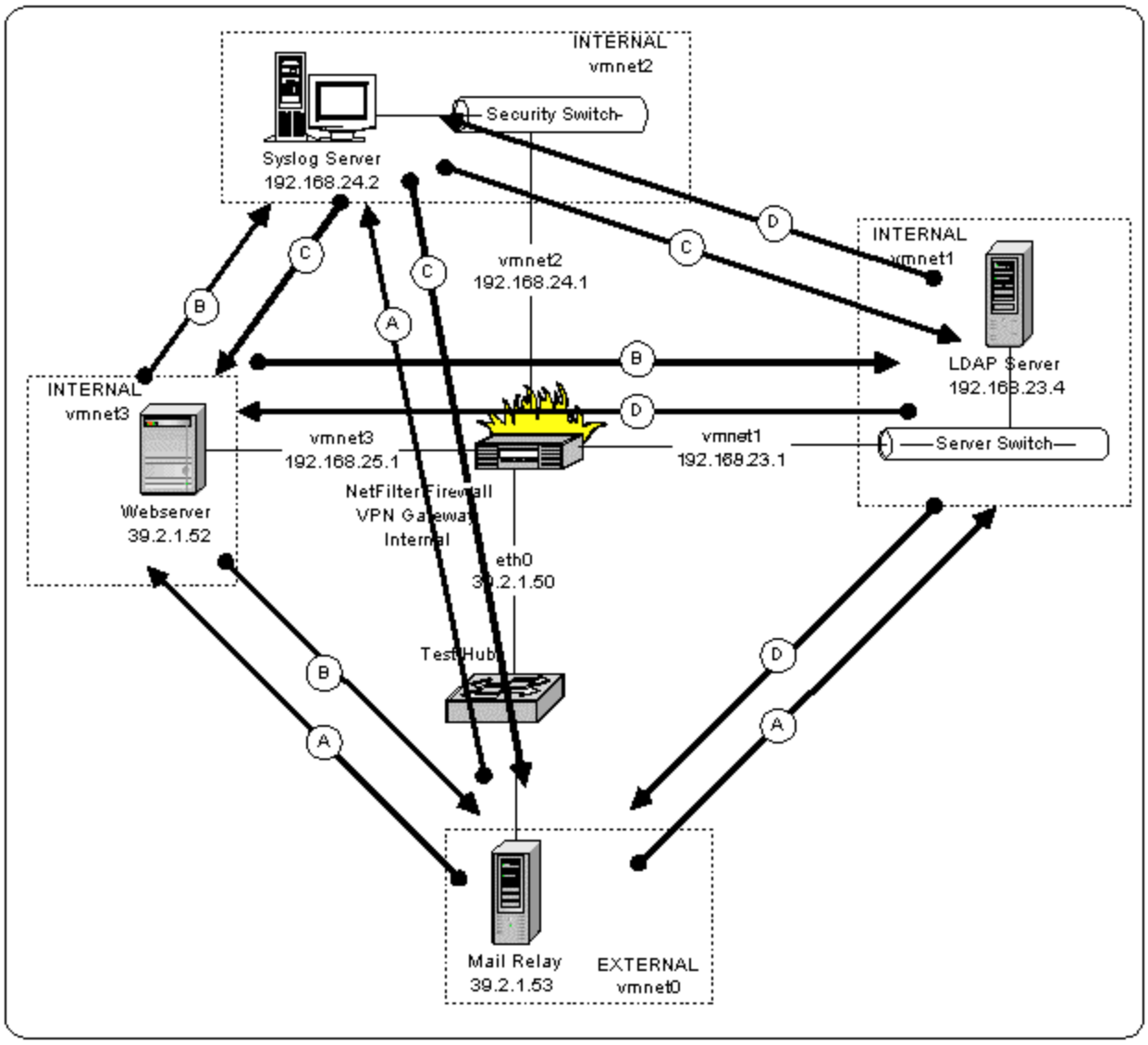
Physical Access
A simple review of the operational procedures in relation to the physical location of the firewalls would be a start. Ensure that access to the physical box itself and any consoles is limited to authorized personnel. As stated above the firewall will be located in an office space that has a physical security system, card access control system and security guards.

OS security
Nessus, an open-sourced security scanner, will test the operating system security of the firewall. The firewall will be dropped and a full scan will be performed against the firewall host. This section will not get into the actual process since Nessus is very simple to operate. Additionally, for the purposes of this paper, we are focusing on verifying the firewall policy.

Remote management
A review of the operational procedures relating to remote management will be conducted. Only limited remote access to the firewalls should be available to authorized personnel. As stated above only two workstations residing on the security network will have SSH access to the firewalls. A layered approach has been taken since either the CIO or Administrator could access their workstations through the VPN gateway and then could access the firewalls. This allows both individuals to access the firewalls in case of emergency and is considered a necessary risk.

Firewall Rulebase

To properly scan the firewall rulebase we will follow a structured approach attempting to scan from each subnet as follows.



Since we have the entire production network simulated in the test environment we could actually perform the following scans from each host on each subnet, which would validate the entire ruleset. For the purposes of the process described below we will be using the LDAP server on the server subnet, the webserver, the Syslog server on the security subnet and the mail relay in the DMZ. For the purposes of this test we will disable all server firewalls. In addition, I will walk-through the audit process from one host giving a step-by-step. This same process will then be used with all four hosts to obtain the results used in the audit evaluation step below. Also we will not be evaluating any VPN components during this audit.

TCP Filtering

Using nmap we will attempt the following scans from outside the firewall. In addition we will run tcpdump on the host we are attempting to scan in order to verify that no traffic can makes it through the firewalls. Using tcpdump on the host to be scanned also will warn us of any partial firewall rules that might be in place (i.e. port 22 is allowed to go from the internal to the external

interface, but not the other way. Typical scan would fail due to no response but tcpdump would show that a packet got through even though a full connection was not set up)

The first scan will use the "-sF" option, which is a Stealth FIN scan that sends a FIN packet as the probe. Closed ports are required to reply to the probe with a RST and open ports should ignore the probe.

---

(1) Steath FIN scan from MailRelay (39.2.1.53) to LDAP server (192.168.23.4)

```
[root@MailRelay root]#nmap –P0 –sF 192.168.23.4
```

Where –P0 tells nmap not to attempt to ping the host before scanning. Since we have ping turned off on the host this keeps us from getting false results stating that the host seems down.

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 11:46 EDT
All 1644 scanned ports on 192.168.23.4 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1904.404 seconds
```

Firewall Log

```
---snip---
Aug 1 11:48:58 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet1 SRC=39.2.1.53
DST=192.168.23.4 LEN=40 TOS=0x00 PREC=0x00 TTL=42 ID=13628 PROTO=TCP
SPT=35027 DPT=3292 WINDOW=4096 RES=0x00 FIN URGP=0
Aug 1 11:48:58 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet1 SRC=39.2.1.53
DST=192.168.23.4 LEN=40 TOS=0x00 PREC=0x00 TTL=42 ID=20272 PROTO=TCP
SPT=35027 DPT=175 WINDOW=4096 RES=0x00 FIN URGP=0
---snip---
```

TCPdump Results

```
Nothing showed up
```

---

(2) Steath FIN scan from MailRelay (39.2.1.53) to SYSLOG server (192.168.24.2)

```
[root@MailRelay root]#nmap –P0 –sF 192.168.24.2
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 12:34 EDT
All 1644 scanned ports on 192.168.24.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1869.235 seconds
```

Firewall Log

```
---snip---
Aug 1 12:18:38 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet2 SRC=39.2.1.53
DST=192.168.24.2 LEN=40 TOS=0x00 PREC=0x00 TTL=41 ID=1225 PROTO=TCP SPT=44678
DPT=345 WINDOW=4096 RES=0x00 FIN URGP=0
Aug 1 12:18:38 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet2 SRC=39.2.1.53
DST=192.168.24.2 LEN=40 TOS=0x00 PREC=0x00 TTL=41 ID=47859 PROTO=TCP
SPT=44678 DPT=577 WINDOW=4096 RES=0x00 FIN URGP=0
```

```
---snip---
```

TCPdump Results
```
Nothing showed up
```

---

(3) Steath FIN scan from MailRelay (39.2.1.53) to Webserver (192.168.25.2)

```
[root@MailRelay root]#nmap –P0 –sF 192.168.25.2
```

Nmap Results
```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 12:41 EDT
All 1644 scanned ports on 192.168.25.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1712.221 seconds
```

Firewall Log
```
---snip---
Aug 1 12:45:22 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet3 SRC=39.2.1.53
DST=192.168.25.2 LEN=40 TOS=0x00 PREC=0x00 TTL=50 ID=25520 PROTO=TCP
SPT=37486 DPT=1024 WINDOW=4096 RES=0x00 FIN URGP=0
Aug 1 12:45:22 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet3 SRC=39.2.1.53
DST=192.168.25.2 LEN=40 TOS=0x00 PREC=0x00 TTL=51 ID=23796 PROTO=TCP
SPT=37486 DPT=1438 WINDOW=4096 RES=0x00 FIN URGP=0
---snip---
```

TCPdump Results
```
Nothing showed up
```

The second scan will use the "-sA" option, which is designed specifically to test firewall rulebases. This method works by sending only ACK packets to probe ports. An ACK packet will always receive a RST packet in response, which does NOT tell you if the port is opened or closed. However, it will tell you that the packet got through and thus is NOT filtered by the firewall.

---

(1) ACK scan from MailRelay (39.2.1.53) to LDAP server (192.168.23.4)

```
[root@MailRelay root]#nmap –P0 –sA 192.168.23.4
```

Nmap Results
```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 12:59 EDT
All 1644 scanned ports on 192.168.23.4 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1380.290 seconds
```

Firewall Log
```
---snip---
Aug 1 11:48:58 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet1 SRC=39.2.1.53
DST=192.168.23.4 LEN=40 TOS=0x00 PREC=0x00 TTL=42 ID=43988 PROTO=TCP
SPT=34903 DPT=11 WINDOW=1024 RES=0x00 ACK URGP=0
```

```
Aug 1 11:48:58 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet1 SRC=39.2.1.53
DST=192.168.23.4 LEN=40 TOS=0x00 PREC=0x00 TTL=41 ID=18777 PROTO=TCP
SPT=34903 DPT=666 WINDOW=4096 RES=0x00 ACK URGP=0
---snip---
```

TCPdump Results
Nothing showed up

---

(2) ACK scan from MailRelay (39.2.1.53) to Syslog server (192.168.24.2)

```
[root@MailRelay root]#nmap –P0 –sA 192.168.24.2
```

Nmap Results
Starting nmap 3.30 (http://www.insecure.org/nmap/) at 2003-08-01 13:21 EDT
All 1644 scanned ports on 192.168.24.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1361.564 seconds

Firewall Log

```
---snip---
Aug 1 13:24:18 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet2 SRC=39.2.1.53
DST=192.168.24.2 LEN=40 TOS=0x00 PREC=0x00 TTL=55 ID=64905 PROTO=TCP
SPT=39641 DPT=314 WINDOW=3072 RES=0x00 ACK URGP=0
Aug 1 13:24:18 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet2 SRC=39.2.1.53
DST=192.168.24.2 LEN=40 TOS=0x00 PREC=0x00 TTL=54 ID=14545 PROTO=TCP
SPT=39641 DPT=1525 WINDOW=3072 RES=0x00 ACK URGP=0
---snip---
```

TCPdump Results
Nothing showed up

---

(3) ACK scan from MailRelay (39.2.1.53) to Webserver (192.168.25.2)

```
[root@MailRelay root]#nmap –P0 –sA 192.168.25.2
```

Nmap Results
Starting nmap 3.30 (http://www.insecure.org/nmap/) at 2003-08-01 13:43 EDT
All 1644 scanned ports on 192.168.25.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1423.029 seconds

Firewall Log

```
---snip---
Aug 1 13:47:12 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet3 SRC=39.2.1.53
DST=192.168.25.2 LEN=40 TOS=0x00 PREC=0x00 TTL=50 ID=39961 PROTO=TCP
SPT=52007 DPT=78 WINDOW=2048 RES=0x00 ACK URGP=0
Aug 1 13:47:12 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet3 SRC=39.2.1.53
DST=192.168.25.2 LEN=40 TOS=0x00 PREC=0x00 TTL=47 ID=20347 PROTO=TCP
SPT=52007 DPT=676 WINDOW=4096 RES=0x00 ACK URGP=0
---snip---
```

TCPdump Results
Nothing showed up

The last scan will use a newer function offered in nmap that is the –ttl option. This allows nmap another way to determine a firewalls ruleset. This method will depend on your firewall generating an ICMP TTL expired error message. When a router or firewall routes an IP packet, the TTL (Time to Live) is decremented by one. This is done to ensure that packets do not end up in endless routing loops. If the router or firewall has a rule in place to allow the packet through then the device will attempt to forward the packet. The devices then decrements its TTL to zero, the packet gets dropped and an ICMP error message gets sent to the remote host (ICMP Type 11, Code 0). This would lets the remote hosts know if the packet never reached its intended designation because the TTL expired. If the device doesn't have a rule allowing the packet then it is simply dropped.

---

(1) SYN scan with TTL=1 from MailRelay (39.2.1.53) to LDAP server (192.168.23.4)

```
[root@MailRelay root]#nmap –P0 –ttl 1 192.168.23.4
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 14:02 EDT
All 1644 scanned ports on 192.168.23.4 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1344.202 seconds
```

Firewall Log

```
---snip---
Aug 1 14:04:15 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=23267
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.23.4 LEN=40 TOS=0x00
PREC=0x00 TTL=1 ID=32789 PROTO=TCP SPT=56715 DPT=318 WINDOW=2048 RES=0x00 SYN
URGP=0 ]
Aug 1 14:04:15 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=23268
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.23.4 LEN=40 TOS=0x00
PREC=0x00 TTL=1 ID=51300 PROTO=TCP SPT=56715 DPT=68 WINDOW=2048 RES=0x00 SYN
URGP=0 ]
---snip---
```

TCPdump Results

```
Nothing showed up
```

---

(2) SYN scan with TTL=1 from MailRelay (39.2.1.53) to SYSLOG server (192.168.24.2)

```
[root@MailRelay root]#nmap –P0 –ttl 1 192.168.24.2
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 14:42 EDT
All 1644 scanned ports on 192.168.24.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1414.322 seconds
```

Firewall Log

```
---snip---
Aug 1 14:44:55 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=24707
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.24.2 LEN=40 TOS=0x00
PREC=0x00 TTL=1 ID=24789 PROTO=TCP SPT=51313 DPT=2000 WINDOW=2048 RES=0x00
SYN URGP=0 ]
Aug 1 14:44:55 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=24708
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.24.2 LEN=40 TOS=0x00
PREC=0x00 TTL=1 ID=22313 PROTO=TCP SPT=51313 DPT=238 WINDOW=2048 RES=0x00 SYN
URGP=0 ]
---snip---
```

TCPdump Results

```
Nothing showed up
```

---

(3) SYN scan with TTL=1 from MailRelay (39.2.1.53) to Webserver (192.168.25.2)

```
[root@MailRelay root]#nmap –P0 –ttl 1 192.168.25.2
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 15:12 EDT
All 1644 scanned ports on 192.168.25.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1314.520 seconds
```

Firewall Log

```
---snip---
Aug 1 15:15:43 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=26017
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.25.2 LEN=40 TOS=0x00
PREC=0x00 TTL=1 ID=32789 PROTO=TCP SPT=48150 DPT=364 WINDOW=2048 RES=0x00 SYN
URGP=0 ]
Aug 1 15:15:43 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=26018
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.25.2 LEN=40 TOS=0x00
PREC=0x00 TTL=1 ID=51300 PROTO=TCP SPT=48150 DPT=33 WINDOW=2048 RES=0x00 SYN
URGP=0 ]
---snip---
```

TCPdump Results

```
Nothing showed up
```

UDP Filtering

This above first method of scanning may work well for TCP, but not as well for UDP. UDP scanning works by sending a UDP packet to a UDP port. If the UDP port is not open and is not listening; an ICMP Port Unreachable error message is generated and sent back to the remote system. This would tell us that the port is not open. If the port is open then the packet would be accepted and no return packet would follow. Since our goal is to determine whether or not a UDP port is open on the firewall the above method gives too many possibilities. If the packet doesn't respond, is it due to the firewall dropping it or letting it pass and the host simply

accepting it. To solve this problem we would utilize a sniffer on the other side of the firewall to determine if the packet made it through. As above, this method can also detect firewall rules only allowing traffic to go in one direction.

---

(1) UDP scan from MailRelay (39.2.1.53) to LDAP server (192.168.23.4)

```
[root@MailRelay root]#nmap –P0 -sU 192.168.23.4
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 15:30 EDT
All 1471 scanned ports on 192.168.23.4 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1779.020 seconds
```

Firewall Log

```
---snip---
Aug 1 15:32:25 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet1 SRC=39.2.1.53
DST=192.168.23.4 LEN=28 TOS=0x00 PREC=0x00 TTL=50 ID=10051 PROTO=UDP
SPT=42578 DPT=769 LEN=8
Aug 1 15:32:25 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet1 SRC=39.2.1.53
DST=192.168.23.4 LEN=28 TOS=0x00 PREC=0x00 TTL=44 ID=62801 PROTO=UDP
SPT=42578 DPT=1669 LEN=8
---snip---
```

TCPdump Results

```
Nothing showed up
```

---

(2) UDP scan from MailRelay (39.2.1.53) to SYSLOG server (192.168.24.2)

```
[root@MailRelay root]#nmap –P0 -sU 192.168.24.2
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 15:51 EDT
All 1471 scanned ports on 192.168.24.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1777.210 seconds
```

Firewall Log

```
---snip---
Aug 1 15:54:55 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet2 SRC=39.2.1.53
DST=192.168.24.2 LEN=28 TOS=0x00 PREC=0x00 TTL=56 ID=23557 PROTO=UDP
SPT=51153 DPT=2032 LEN=8
Aug 1 15:54:55 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet2 SRC=39.2.1.53
DST=192.168.24.2 LEN=28 TOS=0x00 PREC=0x00 TTL=45 ID=17982 PROTO=UDP
SPT=51153 DPT=150 LEN=8
---snip---
```

TCPdump Results

```
Nothing showed up
```

---

(3) UDP scan from MailRelay (39.2.1.53) to Webserver (192.168.25.2)

```
[root@MailRelay root]#nmap –P0 -sU 192.168.25.2
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 16:19 EDT
All 1471 scanned ports on 192.168.25.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1743.811 seconds
```

Firewall Log

```
---snip---
Aug 1 16:24:33 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet3 SRC=39.2.1.53
DST=192.168.25.2 LEN=28 TOS=0x00 PREC=0x00 TTL=34 ID=43051 PROTO=UDP
SPT=61626 DPT=108 LEN=8
Aug 1 16:24:33 INTERNAL kernel: FORWARD DROP:IN=eth0 OUT=vmnet3 SRC=39.2.1.53
DST=192.168.25.2 LEN=28 TOS=0x00 PREC=0x00 TTL=41 ID=23527 PROTO=UDP
SPT=61626 DPT=943 LEN=8
---snip---
```

TCPdump Results

```
Nothing showed up
```

The last scanning method will be similar to the third method used above using the –ttl option with nmap. This method will depend on your firewall generating an ICMP TTL expired error message. When a router or firewall routes an IP packet, the TTL (Time to Live) is decremented by one which is done to ensure that packets do not end up in endless routing loops. If the router or firewall has a rule in place to allow the packet through then the device will attempt to forward the packet. The devices then decrements its TTL to zero, the packet gets dropped and an ICMP error message gets sent to the remote host (ICMP Type 11, Code 0). This would lets the remote hosts know if the packet never reached its intended designation because the TTL expired. If the device doesn't have a rule allowing the packet then it is simply dropped.

---

(1) UDP scan with –ttl=1 from MailRelay (39.2.1.53) to LDAP server (192.168.23.4)

```
[root@MailRelay root]#nmap –P0 –sU –ttl 1 192.168.23.4
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 16:41 EDT
All 1471 scanned ports on 192.168.23.4 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1772.112 seconds
```

Firewall Log

```
---snip---
Aug 1 16:45:13 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=28547
```

```
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.23.4 LEN=28 TOS=0x00
PREC=0x00 TTL=1 ID=15117 PROTO=UDP SPT=58943 DPT=306 LEN=8 ]
Aug 1 16:45:13 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=28548
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.23.4 LEN=28 TOS=0x00
PREC=0x00 TTL=1 ID=10452 PROTO=UDP SPT=58943 DPT=2043 LEN=8 ]
---snip---
```

TCPdump Results

```
Nothing showed up
```

---

(2) UDP scan with –ttl=1 from MailRelay (39.2.1.53) to Syslog server (192.168.24.2)

```
[root@MailRelay root]#nmap –P0 –sU –ttl 1 192.168.24.2
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 17:01 EDT
All 1471 scanned ports on 192.168.24.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1771.913 seconds
```

Firewall Log

```
---snip---
Aug 1 17:03:31 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=30243
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.24.2 LEN=28 TOS=0x00
PREC=0x00 TTL=1 ID=12187 PROTO=UDP SPT=59543 DPT=782 LEN=8 ]
Aug 1 17:03:31 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=30244
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.24.2 LEN=28 TOS=0x00
PREC=0x00 TTL=1 ID=9354 PROTO=UDP SPT=59543 DPT=678 LEN=8 ]
---snip---
```

TCPdump Results

```
Nothing showed up
```

---

(3) UDP scan with –ttl=1 from MailRelay (39.2.1.53) to Webserver (192.168.25.2)

```
[root@MailRelay root]#nmap –P0 –sU –ttl 1 192.168.25.2
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 17:31 EDT
All 1471 scanned ports on 192.168.25.2 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 1781.911 seconds
```

Firewall Log

```
---snip---
Aug 1 17:36:41 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=33479
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.25.2 LEN=28 TOS=0x00
PREC=0x00 TTL=1 ID=2108 PROTO=UDP SPT=59322 DPT=5191 LEN=8 ]
```

```
Aug 1 17:36:41 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=39.2.1.50 DST=39.2.1.53 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=33480
PROTO=ICMP Type=11 CODE=0 [SRC=39.2.1.53 DST=192.168.25.2 LEN=28 TOS=0x00
PREC=0x00 TTL=1 ID=39743 PROTO=UDP SPT=59322 DPT=922 LEN=8 ]
---snip---
```

TCPdump Results

```
Nothing showed up
```

This process would have to be done from each host to truly test the entire ruleset. This is because the firewall rules are tied to source and destination IPs as well as source and destination ports. For example we expect that the internal firewall to allow SLDAP traffic to the LDAP server from the webserver. To verify this, see the following FIN scan below.

Stealth FIN scan from Webserver (192.168.25.2) to LDAP server (192.168.23.4)

```
[root@Webserver root]#nmap –P0 –sF 192.168.23.4
```

Nmap Results

```
Starting nmap 3.30 ( http://www.insecure.org/nmap/ ) at 2003-08-01 18:21 EDT
All 1643 scanned ports on 192.168.23.4 are: filtered
Interesting ports on 192.168.23.4:
Port        State       Service
636/tcp     open        ldapssl

Nmap run completed -- 1 IP address (1 host up) scanned in 193.023 seconds
```

Firewall Log

```
---snip---
Aug 1 18:26:31 INTERNAL kernel: FORWARD DROP:IN=vmnet3 OUT=vmnet1
SRC=192.168.25.2 DST=192.168.23.4 LEN=40 TOS=0x00 PREC=0x00 TTL=57 ID=32723
PROTO=TCP SPT=52409 DPT=48 WINDOWS=2048 RES=0x00 FIN URGP=0
Aug 1 18:26:31 INTERNAL kernel: ICMP-Time Exceeded: IN= OUT=eth0
SRC=192.168.25.2 DST=192.168.23.4 LEN=40 TOS=0x00 PREC=0x00 TTL=64 ID=51787
PROTO=TCP SPT=52409 DPT=3922 WINDOWS=2048 RES=0x00 FIN URGP=0
---snip---
```

TCPdump Results

```
18:25:04.856306 192.168.25.2.35850 > 192.168.23.4.ldaps: F 0:0(0) win 1024
18:25:11.978345 192.168.23.4.35850 > 192.168.23.4.ldaps: F 0:0(0) win 3072
```

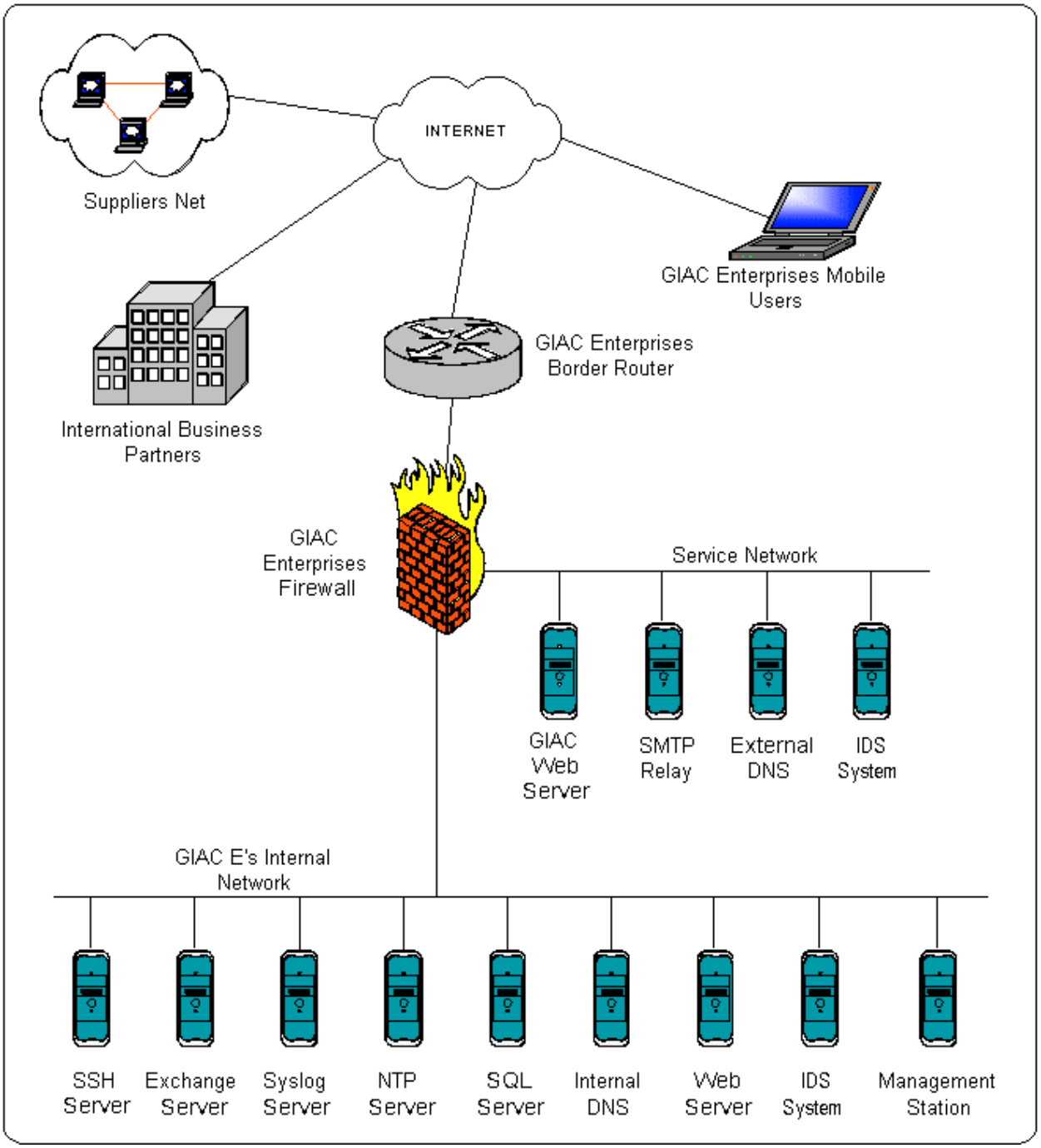
Evaluate the audit

Based on the initial scanning results the ruleset appears complete and tight. Only after scanning the firewall from each host would a complete assessment be available. A change was made to the estimated time required to complete an audit. This was increased based on the time it took to scan the different subnets from a single host.

The firewall successfully limits the amount of information given out based on the scans performed. Even the scan attempting to determine our ruleset based on the firewall generating an ICMP TTL expired error message failed since the firewall logs and drops all outbound ICMP traffic.

Design Under Fire

For this section, I choose to attack the design from James Carlson, GCFW #398. The design can be found at http://www.giac.org/practical/GCFW/James_Carlson_GCFW.pdf. This diagram was reproduced in Visio from page 7 of James Carlson's practical due to my inability to pull an actual copy of his network diagram.[2]



Overview

For this section I will be attempting the following three types of attacks:

- ? Firewall attack
- ? Denial of service attack
- ? Internal system compromise through the perimeter system

Reconnaissance

For the purposes of all three attacks we will assume a file folder was found in the dumpster of a network consulting firm who had previously done some work on GIAC Enterprises IDS system. This folder contained a copy of the above network diagram along with information on the firewall, IDS systems, and other key information about the company.

Firewall attack

From the company folder obtained during reconnaissance we know that as of several weeks ago that GIAC Enterprises was using a Secure Computing Gauntlet Firewall version 6.0 running on a hardened version of Solaris 8 OS.

Firewall Vulnerabilities Found

I found several advisories identifying two vulnerabilities for Gauntlet Firewall for Unix.

Advisory	Description	Link
gauntlet-cyberdaemon-bo (4503)	Gauntlet Firewall CyberPatrol integration buffer overflow	http://xforce.iss.net/xforce/xfdb/4503
CVE-2000-0437	Buffer overflow in the CyberPatrol daemon "cyberdaemon" used in Gauntlet and WebShield allows remote attackers to cause a denial of service or execute arbitrary commands.	http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0437
Bugtraq ID 1234	Gauntlet Firewall Remote Buffer Overflow Vulnerability	http://www.securityfocus.com/bid/1234

The above advisory stated that all Unix versions of Gauntlet were vulnerable to the above. It was later found that version 5.5 was, but version 6.0 was not.

Advisory	Description	Link
Vulnerability Note VU#206723	Network Associates CSMAP and smap/smapd vulnerable to buffer overflow thereby allowing arbitrary command execution	http://www.kb.cert.org/vuls/id/206723
CERT® Advisory CA-2001-25	Buffer Overflow in Gauntlet Firewall allows intruders to execute arbitrary code	http://www.cert.org/advisories/CA-2001-25.html
L-140	Gauntlet Firewall CSMAP and	http://www.ciac.org/ciac/bulle

	smap/smapd Buffer Overflow Vulnerability	tins/l-140.shtml
gauntlet-csmap-bo (7088)	Gauntlet Firewall smap/smapd and CSMAP daemons buffer overflow	http://xforce.iss.net/xforce/xfdb/7088

The original date of the vulnerability was September 6, 2001 and a patch was released to resolve the issue. Although Secure Computing has acquired Gauntlet their patch site at http://www.securecomputing.com/index.cfm?sKey=987 shows that a total of 6 patch levels have been applied regarding the csmap issue with the latest being issued on April 21, 2003.

Design the attack

Since no proof-of-concept code was found on the Internet, we could attempt to create one by attempting to reverse engineer the patch to determine the specifications of the vulnerability.

Explain the results

Since the patch to fix the vulnerability is on the mandatory list, designing working exploit code on the off chance that such a patch was left out seems futile. There are much easier ways to penetrate the network defenses, as we will see later. This attack would be classified as a failed attack since effort had evaluated whether it would work.

Countermeasures

Keeping up to date on both the firewall and OS patches are the key to counter attacks to such defined vulnerabilities.

Denial of service attack

For this attack we will take 50 compromised cable modem/DSL systems running the Windows OS and attack GIAC Enterprises. This attack is totally feasible, given the large amount of uneducated users running the Windows operating system with no security controls in place.

For this portion of the attack we will be using Tribe FloodNet 2K (TFN2K) to generate a Distributed Denial of Service (DDOS) attack. The TFN2K software consists of a master server that communicates commands one way to multiple agents via encrypted session. TFN2K can be found at http://mixter.void.ru/tfn2k.tgz and an advisory can be found as follows.

Advisory	Description	Link
CERT® Advisory CA-1999-17	Denial-of-Service Tools	http://www.cert.org/advisories/CA-1999-17.html
CERT® Advisory CA-2000-01	Denial-of-Service Developments	http://www.cert.org/advisories/CA-2000-01.html

Design the attack

For each of our compromised hosts we will be uploading the TFN2K agent. TFN2K has several options including the ability to attack targets with the following:

- TCP/SYN packet flood
- UDP packet flood

- ? ICMP/PING packet flood
- ? BROADCAST PING (SMURF) packet flood
- ? Randomly alternate between all four types
- ? Targa3 flood (IP stack penetration)

Explain the results

Several things could happen including:

- ? Causing a DOS on the router due to the load of packets being evaluated, logged and/or dropped.
- ? Causing any internet routable host to be flooded so legitimate traffic cannot enter (i.e. Webserver, SMTP Relay, etc.)
- ? Causing the firewall to bog down especially when dealing with traffic being proxied through the firewall.
- ? Causing a significant traffic load on the limited bandwith effectively shutting down the entire Internet connection

Countermeasures

Several countermeasures can be performed to assist with protecting against such attacks.

- ? Protect the hosts themselves. For Web servers add a HTTP reverse proxy or web application proxy.
- ? Limiting the logging of ACLs that potentially could have a large amount of traffic.
- ? Use of load balancing technology and upgrading the routers would assist with the traffic load.
- ? Use of inline traffic monitor such as Toplayer's Attack Mitigator Intrusion Prevention System (IPS).
- ? Use of load balancing technology for the firewalls.

Internal system compromise through the perimeter system

This last attack will show why a Defense-in-Depth approach is the only way. We will perform additional reconnaissance on GIAC Enterprises via the web. Through examination of the company website and searching the newsgroup archives we obtain the email addresses of several employees. This tells us the email format they use. While further researching GIAC enterprises, we find an article on a local charity that the company assisted with a fund driver, which listed the names of additional employees. These email addresses will be our initial targets.

Design the attack

We take a cute electronic cartoon found on the net and bundled it with a custom written windows rootkit and email it to each of our previously found email addresses. Our custom kernel rootkit has the ability to perform keystroke monitoring, SSL based http tunneling, process hiding, port scanner, directory hiding, privilege escalation, port redirection and token manipulation. Out of everyone that it was sent to only two people actually looked at the cartoon, which loaded our rootkit. The rootkit then tunnels back to a predefined IP via an SSL-enabled http-tunnel. This predefined IP is a host that was previously compromised by another rootkit and acts as Master Control for our Agent rootkit. Next our Agent rootkit sets up in the registry to enable auto-load

upon boot capability. The Agents will automatically start keystroke monitoring upon restart for the first 30 minutes, sending any compromised passwords back through the http-tunnel. From the Master we start using a port scanner in stealth mode as to stay under any port-scanning threshold. Since there is no separation between the user subnet and server subnet we are able to enumerate all the internal servers. In addition we are waiting for other users to log into our compromised machines to gain their userids/passwords. We could even cause some problems by changing account information, which would require a systems administrator to log into fix. Then we would have an administrator account and password. We would then attempt to log into all of the Windows servers or workstations on the subnet to upload our rootkit and modify the appropriate auto-start registry settings. Then upon reboot we would own those machines also. They would promptly register with the Master Control server and all would be lost.

Explain the results

Effectively the above scenario would yield total control of the internal network.

Detection

The above scenario would bypass traditional virus defense and signature-based IDS systems since both rely on a signature to detect malicious files and/or traffic.

Countermeasures

Proper Defense-in-Depth includes the use of host-based Intrusion Detection Systems, host-based firewalls, internal firewalls dividing up server and user subnets, File-integrity protection on workstations and servers and up-to-date patches on all servers/ workstations. Also the principle of least privilege would support the denial of direct Internet access for servers unless required.

Appendix A – Traffic Analysis

Service	Protocol	Traffic Source	Source IP	SPort	Traffic Destination	Destination IP	DPort	Router	ExtFW	IntFW	Host FW
DNS	udp	ALL	ALL	53	Ext DNS/NTP	39.2.1.54	E	X	X		X
DNS	tcp	Ext DNS/NTP	39.2.1.54	53	ISP DNS server	39.0.1.20	53	X	X		X
DNS	udp	Ext DNS/NTP	39.2.1.54	E	ALL	ALL	53	X	X		X
DNS	tcp	ISP DNS server	39.0.1.20	53	Ext DNS/NTP	39.2.1.54	53	X	X		X
ESP	50	ALL	ALL	-	Int FW/VPN GW	39.2.1.50	-	X	X	X	
ESP	50	Int FW/VPN GW	39.2.1.50	-	ALL	ALL	-	X	X	X	
HTTP	tcp	ALL	ALL	E	Squid server	39.2.1.55	80	X	X		
HTTP	tcp	Squid server	39.2.1.55	80	ALL	ALL	E	X	X		X
HTTPS	tcp	ALL	ALL	E	Squid server	39.2.1.55	443	X	X		X
HTTPS	tcp	ALL	ALL	E	WebMail server	39.2.1.51	443	X	X		X
HTTPS	tcp	Squid server	39.2.1.55	443	ALL	ALL	E	X	X		X
HTTPS	tcp	WebMail server	39.2.1.51	443	ALL	ALL	E	X	X		X
IKE	udp	ALL	ALL	500	Int FW/VPN GW	39.2.1.50	500	X	X	X	
IKE	udp	Int FW/VPN GW	39.2.1.50	500	ALL	ALL	500	X	X	X	
NTP	udp	Border Router	39.2.1.45	E	Ext DNS/NTP	36.2.1.54	123	X	X		X
NTP	udp	Ext DNS/NTP	36.2.1.54	123	Border Router	39.2.1.45	E	X	X		X
SMTP	tcp	ALL	ALL	E	Mail Relay	39.2.1.53	25	X	X		X
SMTP	tcp	Mail Relay	39.2.1.53	25	ALL	E	25	X	X		X
SYSLOG	udp	Border Router	39.2.1.54	514	Syslog server	39.2.1.62	514	X	X		X
DNS	udp	Database server	192.168.23.2	E	Int DNS/NTP	192.168.23.6	53				X
DNS	udp	Ext DNS/NTP	39.2.1.54	53	External FW	39.2.1.49	E		X		X
DNS	udp	Ext DNS/NTP	39.2.1.54	53	Int DNS/NTP	192.168.23.6	53			X	X
DNS	udp	Ext DNS/NTP	39.2.1.54	53	Mail Relay	39.2.1.53	E				X
DNS	udp	Ext DNS/NTP	39.2.1.54	53	Squid server	39.2.1.55	E				X
DNS	udp	Ext DNS/NTP	39.2.1.54	53	WebMail server	39.2.1.51	E				X
DNS	udp	Ext DNS/NTP	39.2.1.54	53	Webserver	192.168.25.2	E				X
DNS	udp	External FW	39.2.1.49	E	Ext DNS/NTP	39.2.1.54	53		X		X
DNS	udp	Int DNS/NTP	192.168.23.6	53	Database server	192.168.23.2	E				X
DNS	udp	Int DNS/NTP	192.168.23.6	53	Ext DNS/NTP	39.2.1.54	53			X	X
DNS	udp	Int DNS/NTP	192.168.23.6	53	Intranet server	192.168.23.5	E				X
DNS	udp	Int DNS/NTP	192.168.23.6	53	LDAP server	192.168.23.4	E				X
DNS	udp	Int DNS/NTP	192.168.23.6	53	Mail server	192.168.23.3	E				X
DNS	udp	Int DNS/NTP	192.168.23.6	53	VPN tunnel	192.168.23.1	E			X	X
DNS	udp	Intranet server	192.168.23.5	E	Int DNS/NTP	192.168.23.6	53				X
DNS	udp	LDAP server	192.168.23.4	E	Int DNS/NTP	192.168.23.6	53				X
DNS	udp	Mail Relay	39.2.1.53	E	Ext DNS/NTP	39.2.1.54	53				X
DNS	udp	Mail server	192.168.23.3	E	Int DNS/NTP	192.168.23.6	53				X
DNS	udp	Squid server	39.2.1.55	E	Ext DNS/NTP	39.2.1.54	53				X
DNS	udp	VPN tunnel	192.168.23.1	E	Int DNS/NTP	192.168.23.6	53			X	X
DNS	udp	WebMail server	39.2.1.51	E	Ext DNS/NTP	39.2.1.54	53				X
DNS	udp	Webserver	192.168.25.2	E	Ext DNS/NTP	39.2.1.54	53				X
HTTP	tcp	Admin wksn	192.168.24.5	E	Squid server	39.2.1.55	3128			X	X
HTTP	tcp	CIO wksn	192.168.24.4	E	Squid server	39.2.1.55	3128			X	X
HTTP	tcp	Intranet server	192.168.23.5	80	VPN tunnel	192.168.23.1	E			X	X
HTTP	tcp	Squid server	39.2.1.55	3128	Admin wksn	192.168.24.5	E			X	X

HTTP	tcp	Squid server	39.2.1.55	3128	CIO wksn	192.168.24.4	E		X	X
HTTP	tcp	Squid server	39.2.1.55	8080	Webserver	39.2.1.52	80			X
HTTP	tcp	VPN tunnel	192.168.23.1	E	Intranet server	192.168.23.5	80		X	X
HTTP	tcp	Webserver	39.2.1.52	80	Squid server	39.2.1.55	8080			X
HTTPS	tcp	Admin wksn	192.168.24.5	E	IDS server	192.168.24.3	443			X
HTTPS	tcp	Admin wksn	192.168.24.5	E	Squid server	39.2.1.55	3128		X	X
HTTPS	tcp	CIO wksn	192.168.24.4	E	IDS server	192.168.24.3	443			X
HTTPS	tcp	CIO wksn	192.168.24.4	E	Squid server	39.2.1.55	3128		X	X
HTTPS	tcp	IDS server	192.168.24.3	443	Admin wksn	192.168.24.5	E			X
HTTPS	tcp	IDS server	192.168.24.3	443	CIO wksn	192.168.24.4	E			X
HTTPS	tcp	Squid server	39.2.1.55	3128	Admin wksn	192.168.24.5	E		X	X
HTTPS	tcp	Squid server	39.2.1.55	3128	CIO wksn	192.168.24.4	E		X	X
HTTPS	tcp	Squid server	39.2.1.55	443	Webserver	39.2.1.52	443			X
HTTPS	tcp	Webserver	39.2.1.52	443	Squid server	39.2.1.55	443			X
IMAP	tcp	Mail server	192.168.23.3	143	VPN tunnel	192.168.23.1	E		X	X
IMAP	tcp	Mail server	192.168.23.3	143	WebMail server	39.2.1.51	143		X	X
IMAP	tcp	VPN tunnel	192.168.23.1	E	Mail server	192.168.23.3	143		X	X
IMAP	tcp	WebMail server	39.2.1.51	143	Mail server	192.168.23.3	143		X	X
MySQL	tcp	Database server	192.168.23.2	3306	VPN tunnel	192.168.23.1	E		X	X
MySQL	tcp	Database server	192.168.23.2	3306	Webserver	192.168.25.2	E		X	X
MySQL	tcp	IDS sensor A	192.168.26.2	E	IDS server	192.168.24.3	3306			X
MySQL	tcp	IDS sensor B	192.168.26.3	E	IDS server	192.168.24.3	3306			X
MySQL	tcp	IDS sensor C	192.168.26.4	E	IDS server	192.168.24.3	3306			X
MySQL	tcp	IDS server	192.168.24.3	3306	IDS sensor A	192.168.26.2	E			X
MySQL	tcp	IDS server	192.168.24.3	3306	IDS sensor B	192.168.26.3	E			X
MySQL	tcp	IDS server	192.168.24.3	3306	IDS sensor C	192.168.26.4	E			X
MySQL	tcp	VPN tunnel	192.168.23.1	E	Database server	192.168.23.2	3306		X	X
MySQL	tcp	Webserver	192.168.25.2	E	Database server	192.168.23.2	3306		X	X
NTP	udp	Database server	192.168.23.2	E	Int DNS/NTP	192.168.23.6	123			X
NTP	udp	Ext DNS/NTP	36.2.1.54	123	External FW	39.2.1.49	E	X		X
NTP	udp	Ext DNS/NTP	36.2.1.54	123	Mail Relay	39.2.1.53	E			X
NTP	udp	Ext DNS/NTP	36.2.1.54	123	Squid server	39.2.1.55	E			X
NTP	udp	Ext DNS/NTP	36.2.1.54	123	WebMail server	39.2.1.51	E			X
NTP	udp	Ext DNS/NTP	39.2.1.54	123	Webserver	192.168.25.2	E			X
NTP	udp	External FW	39.2.1.49	E	Ext DNS/NTP	36.2.1.54	123	X		X
NTP	udp	IDS Server	192.168.24.3	E	Int DNS/NTP	192.168.23.6	123		X	X
NTP	udp	Int DNS/NTP	192.168.23.6	123	Database server	192.168.23.2	E			X
NTP	udp	Int DNS/NTP	192.168.23.6	123	IDS Server	192.168.24.3	E		X	X
NTP	udp	Int DNS/NTP	192.168.23.6	123	Intranet server	192.168.23.5	E			X
NTP	udp	Int DNS/NTP	192.168.23.6	123	LDAP server	192.168.23.4	E			X
NTP	udp	Int DNS/NTP	192.168.23.6	123	Mail server	192.168.23.3	E			X
NTP	udp	Int DNS/NTP	192.168.23.6	123	Syslog server	192.168.24.2	E		X	X
NTP	udp	Int DNS/NTP	192.168.23.6	123	VPN tunnel	192.168.24.1	E		X	X
NTP	udp	Intranet server	192.168.23.5	E	Int DNS/NTP	192.168.23.6	123			X
NTP	udp	LDAP server	192.168.23.4	E	Int DNS/NTP	192.168.23.6	123			X
NTP	udp	Mail Relay	39.2.1.53	E	Ext DNS/NTP	36.2.1.54	123			X
NTP	udp	Mail server	192.168.23.3	E	Int DNS/NTP	192.168.23.6	123			X
NTP	udp	Squid server	39.2.1.55	E	Ext DNS/NTP	36.2.1.54	123			X
NTP	udp	Syslog server	192.168.24.2	E	Int DNS/NTP	192.168.23.6	123		X	X
NTP	udp	VPN tunnel	192.168.23.1	E	Int DNS/NTP	192.168.23.6	123		X	X
NTP	udp	WebMail server	39.2.1.51	E	Ext DNS/NTP	36.2.1.54	123			X
NTP	udp	Webserver	192.168.25.2	E	Int DNS/NTP	192.168.23.6	123		X	X
SLDAP	tcp	Intranet server	192.168.23.4	E	LDAP server	192.168.23.4	689			X
SLDAP	tcp	LDAP server	192.168.23.4	689	Intranet server	192.168.23.4	E			X

SLDAP	tcp	LDAP server	192.168.23.4	689	VPN tunnel	192.168.23.1	E		X	X
SLDAP	tcp	LDAP server	192.168.23.4	689	WebMail server	39.2.1.51	E		X	X
SLDAP	tcp	LDAP server	192.168.23.4	689	Webserver	192.168.25.2	E		X	X
SLDAP	tcp	VPN tunnel	192.168.23.1	E	LDAP server	192.168.23.4	689		X	X
SLDAP	tcp	WebMail server	39.2.1.51	E	LDAP server	192.168.23.4	689		X	X
SLDAP	tcp	Webserver	192.168.25.2	E	LDAP server	192.168.23.4	689		X	X
SMTP	tcp	Mail Relay	39.2.1.53	25	Mail Server	192.168.23.3	25		X	X
SMTP	tcp	Mail Server	192.168.23.3	25	Mail Relay	39.2.1.53	25		X	X
SMTP	tcp	Mail server	192.168.23.3	25	VPN tunnel	192.168.23.1	E		X	X
SMTP	tcp	Mail server	192.168.23.3	25	WebMail server	39.2.1.51	E		X	X
SMTP	tcp	VPN tunnel	192.168.23.1	E	Mail server	192.168.23.3	25		X	X
SMTP	tcp	WebMail server	39.2.1.51	E	Mail server	192.168.23.3	25		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	Database server	192.168.23.2	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	Ext DNS/NTP	39.2.1.54	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	External FW	39.2.1.49	22	X	X	X
SSH	tcp	Admin wksn	192.168.24.5	E	IDS server	192.168.24.3	22			X
SSH	tcp	Admin wksn	192.168.24.5	E	Int DNS/NTP	192.168.23.6	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	Int FW/VPN GW	192.168.24.1	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	Intranet server	192.168.23.5	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	LDAP server	192.168.23.4	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	Mail Relay	39.2.1.53	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	Mail server	192.168.23.3	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	Squid server	39.2.1.55	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	Syslog server	192.168.24.2	22			X
SSH	tcp	Admin wksn	192.168.24.5	E	WebMail server	39.2.1.51	22		X	X
SSH	tcp	Admin wksn	192.168.24.5	E	Webserver	192.168.25.2	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	Database server	192.168.23.2	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	Ext DNS/NTP	39.2.1.54	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	External FW	39.2.1.49	22	X	X	X
SSH	tcp	CIO wksn	192.168.24.4	E	IDS server	192.168.24.3	22			X
SSH	tcp	CIO wksn	192.168.24.4	E	Int DNS/NTP	192.168.23.6	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	Int FW/VPN GW	192.168.24.1	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	Intranet server	192.168.23.5	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	LDAP server	192.168.23.4	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	Mail Relay	39.2.1.53	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	Mail server	192.168.23.3	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	Squid server	39.2.1.55	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	Syslog server	192.168.24.2	22			X
SSH	tcp	CIO wksn	192.168.24.4	E	WebMail server	39.2.1.51	22		X	X
SSH	tcp	CIO wksn	192.168.24.4	E	Webserver	192.168.25.2	22		X	X
SSH	tcp	Database server	192.168.23.2	22	Admin wksn	192.168.24.5	E		X	X
SSH	tcp	Database server	192.168.23.2	22	CIO wksn	192.168.24.4	E		X	X
SSH	tcp	Ext DNS/NTP	39.2.1.54	22	Admin wksn	192.168.24.5	E		X	X
SSH	tcp	Ext DNS/NTP	39.2.1.54	22	CIO wksn	192.168.24.4	E		X	X
SSH	tcp	External FW	39.2.1.49	22	Admin wksn	192.168.24.5	E	X	X	X
SSH	tcp	External FW	39.2.1.49	22	CIO wksn	192.168.24.4	E	X	X	X
SSH	tcp	IDS sensor A	192.168.26.2	22	IDS server	192.168.26.1	22			X
SSH	tcp	IDS sensor B	192.168.26.3	22	IDS server	192.168.26.1	22			X
SSH	tcp	IDS sensor C	192.168.26.4	22	IDS server	192.168.26.1	22			X
SSH	tcp	IDS server	192.168.24.3	22	Admin wksn	192.168.24.5	E			X
SSH	tcp	IDS server	192.168.24.3	22	CIO wksn	192.168.24.4	E			X
SSH	tcp	IDS server	192.168.26.1	22	IDS sensor A	192.168.26.2	22			X
SSH	tcp	IDS server	192.168.26.1	22	IDS sensor B	192.168.26.3	22			X
SSH	tcp	IDS server	192.168.26.1	22	IDS sensor C	192.168.26.4	22			X

SSH	tcp	Int DNS/NTP	192.168.23.6	22	Admin wksn	192.168.24.5	E	X	X
SSH	tcp	Int DNS/NTP	192.168.23.6	22	CIO wksn	192.168.24.4	E	X	X
SSH	tcp	Int FW/VPN GW	192.168.24.1	22	Admin wksn	192.168.24.5	E	X	X
SSH	tcp	Int FW/VPN GW	192.168.24.1	22	CIO wksn	192.168.24.4	E	X	X
SSH	tcp	Intranet server	192.168.23.5	22	Admin wksn	192.168.24.5	E	X	X
SSH	tcp	Intranet server	192.168.23.5	22	CIO wksn	192.168.24.4	E	X	X
SSH	tcp	LDAP server	192.168.23.4	22	Admin wksn	192.168.24.5	E	X	X
SSH	tcp	LDAP server	192.168.23.4	22	CIO wksn	192.168.24.4	E	X	X
SSH	tcp	Mail Relay	39.2.1.53	22	Admin wksn	192.168.24.5	E	X	X
SSH	tcp	Mail Relay	39.2.1.53	22	CIO wksn	192.168.24.4	E	X	X
SSH	tcp	Mail server	192.168.23.3	22	Admin wksn	192.168.24.5	E	x	X
SSH	tcp	Mail server	192.168.23.3	22	CIO wksn	192.168.24.4	E	X	X
SSH	tcp	Squid server	39.2.1.55	22	Admin wksn	192.168.24.5	E	X	X
SSH	tcp	Squid server	39.2.1.55	22	CIO wksn	192.168.24.4	E	X	X
SSH	tcp	Syslog server	192.168.24.2	22	Admin wksn	192.168.24.5	E		X
SSH	tcp	Syslog server	192.168.24.2	22	CIO wksn	192.168.24.4	E		X
SSH	tcp	WebMail server	39.2.1.51	22	Admin wksn	192.168.24.5	E	X	X
SSH	tcp	WebMail server	39.2.1.51	22	CIO wksn	192.168.24.4	E	X	X
SSH	tcp	Webserver	192.168.25.2	22	Admin wksn	192.168.24.5	E	X	X
SSH	tcp	Webserver	192.168.25.2	22	CIO wksn	192.168.24.4	E	X	X
SYSLOG	udp	Database server	192.168.23.2	514	Syslog server	192.168.24.2	514	X	X
SYSLOG	udp	Ext DNS/NTP	39.2.1.54	514	Syslog server	39.2.1.62	514	X	X
SYSLOG	udp	External FW	39.2.1.49	514	Syslog server	39.2.1.62	514	X	X
SYSLOG	udp	Int DNS/NTP	192.168.23.6	514	Syslog server	192.168.24.2	514	X	X
SYSLOG	udp	Int FW/VPN GW	192.168.24.1	514	Syslog server	192.168.24.2	514	X	X
SYSLOG	udp	Intranet server	192.168.23.5	514	Syslog server	192.168.24.2	514	X	X
SYSLOG	udp	LDAP server	192.168.23.4	514	Syslog server	192.168.24.2	514	X	X
SYSLOG	udp	Mail Relay	39.2.1.53	514	Syslog server	39.2.1.62	514	X	X
SYSLOG	udp	Mail server	192.168.23.3	514	Syslog server	192.168.24.2	514	X	X
SYSLOG	udp	Squid server	39.2.1.55	514	Syslog server	39.2.1.62	514	X	X
SYSLOG	udp	WebMail server	39.2.1.51	514	Syslog server	39.2.1.62	514	X	X
SYSLOG	udp	Webserver	192.168.25.2	514	Syslog server	192.168.24.2	514	X	X

E – Ephermal port

Appendix B – External Firewall Configuration

```
------------------ start of external firewall configuration -----------------
#!/bin/sh
#
# Startup script for NetFilter firewall on EXTERNAL
# Replacement for the default iptables script
# Requires iptables-1.2.8 from http://www.netfilter.org
#
# 030622 - written by GDW
#

# Setup global variables
IPT=/usr/local/sbin/iptables
LOGLEVEL=notice
DEBUG=0
# Define Interfaces
EXTIF=vmnet1
INTIF=eth0

# Ensure iptables command is available
if [ ! -x $IPT ]; then
      echo "ERROR: Cannot locate iptables command."
      exit 0
fi

# Ensure ipchains command is not running
if  /sbin/lsmod 2>/dev/null |grep -q ipchains ; then
      echo "ERROR: Cannot run both ipchains and iptables"
      exit 0
fi

start() {

      # Set up local variables
      BLACKLIST=/etc/blacklist.conf
      VPNCLIENT=/etc/vpnclient.conf

      # Log to screen and syslog the start message
      echo "Starting External NetFilter Firewall"
      logger -p local3.info NetFilter Firewall Started

      # Enable syn cookie support
      echo 1 > /proc/sys/net/ipv4/tcp_syncookies

      # Enable logging of Martians (oddball addresses).
      echo 1 > /proc/sys/net/ipv4/conf/all/log_martians

      # Enable IP Forward support.
      echo 1 > /proc/sys/net/ipv4/ip_forward

      # Disable response to ping
      echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all

      # Disable response to Broadcasts (to keep from being SMURF amplifier)
```

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# Enable Dynamic IP Address support.
echo 1 > /proc/sys/net/ipv4/ip_dynaddr

# Enable anti spoofing support.
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter

# Disable source routing support.
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route

# Disable icmp redirect support.
echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects

##################################
# Flush,delete and zero existing chains
##################################

  $IPT -t filter -F
  $IPT -t nat -F
  $IPT -t mangle -F

$IPT -X
$IPT -Z

##################################
# Set up default policies
##################################

$IPT -t nat -P PREROUTING ACCEPT
$IPT -t nat -P POSTROUTING ACCEPT
$IPT -t nat -P OUTPUT ACCEPT

$IPT -t mangle -P PREROUTING ACCEPT
$IPT -t mangle -P INPUT ACCEPT
$IPT -t mangle -P FORWARD ACCEPT
$IPT -t mangle -P OUTPUT ACCEPT
$IPT -t mangle -P POSTROUTING ACCEPT

$IPT -t filter -P FORWARD DROP
$IPT -t filter -P INPUT DROP
$IPT -t filter -P OUTPUT DROP

##################################
# nat Module Rules
##################################

# This table is consulted when a packet which creates a new
# connection is encountered. Use PREROUTING to drop unwanted new
# traffic before passing on to FORWARD or INPUT Filters.

# Block Reserved/Private Addresses coming in the external interface.
# Data pulled from http://www.iana.org/assignments/ipv4-address-space
# Dropped 39.0.0.0/8 from Reserved list since I have given part of
# this address space to GIAC Enterprises

IANA_RESERVED="0 1 2 5 7 23 27 31 36 37 41 42 58 59 70 71 72 73 \
```

```
74 75 76 77 78 79 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 \
99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 \
116 117 118 119 120 121 122 123 124 125 126 127 173 174 175 176 177 \
178 179 180 181 182 183 184 185 186 187 189 190 197 223 240 241 242 \
243 244 245 246 247 248 249 250 251 252 253 254 255"

for NET in $IANA_RESERVED; do
      $IPT -t nat -A PREROUTING -i $EXTIF -s $NET.0.0.0/8 -j DROP
done

IANA_MULTICAST="224 225 226 227 228 229 230 231 232 233 234 235 236 \
237 238 239"
for NET in $IANA_MULTICAST; do
      $IPT -t nat -A PREROUTING -i $EXTIF -s $NET.0.0.0/8 -j DROP
done

# IANA PRIVATE IPv4 address space
# RFC 1918 Private Network
$IPT -t nat -A PREROUTING -i $EXTIF -s 10.0.0.0/8 -j DROP
# RFC 1918 Private Network
$IPT -t nat -A PREROUTING -i $EXTIF -s 172.16/12 -j DROP
# RFC 1918 Private Network
$IPT -t nat -A PREROUTING -i $EXTIF -s 192.168.0.0/16 -j DROP
# RFC 3068 IANA Special Use
$IPT -t nat -A PREROUTING -i $EXTIF -s 192.88.99.0/24 -j DROP

################################
# Mangle Module Rules.
################################

# Rules to mangle TOS values of packets routed through the firewall.
# Based on RFC 1060/1349

# TOS stuff: (type: iptables -m tos -h)
# Minimize-Delay 16 (0x10)
# Maximize-Throughput 8 (0x08)
# Maximize-Reliability 4 (0x04)
# Minimize-Cost 2 (0x02)
# Normal-Service 0 (0x00)

$IPT -t mangle -A PREROUTING -p tcp --dport 22 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 25 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p udp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 80 -j TOS --set-tos 8

###################################
# Filter Module Rules
###################################

# Setup debugging for tracking traffic flows
if [ $DEBUG = 1 ]; then
      echo "Extra Debug logging enabled"
      $IPT -A INPUT -i lo -j LOG --log-level $LOGLEVEL \
            --log-prefix "Local-in: "
      $IPT -A OUTPUT -o lo -j LOG --log-level $LOGLEVEL \
            --log-prefix "Local-out: "
```

```
        $IPT -A FORWARD -p 50 -i $EXTIF -j LOG --log-level $LOGLEVEL \
              --log-prefix "ESP-in: "
        $IPT -A FORWARD -p 50 -i $INTIF -j LOG --log-level $LOGLEVEL \
              --log-prefix "ESP-out: "
        $IPT -A FORWARD -p udp -i $EXTIF --sport 500 --dport 500 -j LOG \
              --log-level $LOGLEVEL --log-prefix "IKE-in: "
        $IPT -A FORWARD -p udp -i $INTIF --sport 500 --dport 500 -j LOG \
              --log-level $LOGLEVEL --log-prefix "IKE-out: "
        $IPT -A FORWARD -p udp --sport 53 --dport 53 -j LOG \
              --log-level $LOGLEVEL --log-prefix "DNS: "
        $IPT -A FORWARD -p udp --sport 123 --dport 123 -j LOG \
              --log-level $LOGLEVEL --log-prefix "NTP: "
        $IPT -A FORWARD -p udp --sport 514 --dport 514 -j LOG \
              --log-level $LOGLEVEL --log-prefix "SYSLOG: "
        $IPT -A FORWARD -p tcp --dport 80 -j LOG \
              --log-level $LOGLEVEL --log-prefix "HTTP: "
        $IPT -A FORWARD -p tcp --dport 443 -j LOG \
              --log-level $LOGLEVEL --log-prefix "HTTPS: "
        $IPT -A FORWARD -p tcp --dport 25 -j LOG \
              --log-level $LOGLEVEL --log-prefix "SMTP: "
        $IPT -A FORWARD -p tcp --dport 143 -j LOG \
              --log-level $LOGLEVEL --log-prefix "IMAP: "
        $IPT -A FORWARD -p tcp --dport 3306 -j LOG \
              --log-level $LOGLEVEL --log-prefix "MySQL: "
        $IPT -A FORWARD -p tcp --dport 636 -j LOG \
              --log-level $LOGLEVEL --log-prefix "LDAPS: "
        $IPT -A FORWARD -p tcp --dport 22 -j LOG \
              --log-level $LOGLEVEL --log-prefix "SSH: "
fi

## End additional logging

# Allow traffic on loopback interface
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT

# Allow IPsec into $EXTIF
# Filter access to IKE Negotiations to only allow authorized clients
if [ "$VPNCLIENT" != "" ]; then
        echo "Loading authorized VPN clients from $VPNCLIENT"
        for host in `grep -v ^# $VPNCLIENT | awk '{print $1}'` ; do
              $IPT -A FORWARD -p udp -i $EXTIF -s $host --sport 500 \
                    --dport 500 -j ACCEPT
              $IPT -A FORWARD -p udp -i $INTIF --sport 500 -d $host \
                    --dport 500 -j ACCEPT
              echo "VPN Client: $host "
        done
fi

# ESP encryption and authentication
$IPT -A FORWARD -p 50 -i $EXTIF -d 39.2.1.50 -j ACCEPT
$IPT -A FORWARD -p 50 -i $INTIF -s 39.2.1.50 -j ACCEPT

# Setup ICMP traffic chain
$IPT -N ICMP_TRAFFIC

# Accept needed ICMP traffic from net for EXTERNAL router
```

```
# Accept destination-unreachable traffic
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 3 -j ACCEPT
# Accept time-exceeded icmp traffic
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 11 -j ACCEPT

# Log traffic by type and drop
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 0 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Echo Reply: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 8 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Echo Request: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 4 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Source Quench: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 5 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Redirect: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 9 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Router Advertisement: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 10 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Router Selection: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 30 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Traceroute: "

$IPT -A ICMP_TRAFFIC -j DROP

# Push all ICMP traffic to ICMP chain
$IPT -A INPUT -p icmp -j ICMP_TRAFFIC
$IPT -A FORWARD -p icmp -j ICMP_TRAFFIC
$IPT -A OUTPUT -p icmp -j ICMP_TRAFFIC

# Setup Blacklist chain
$IPT -N BLACKLIST
$IPT -A BLACKLIST -j LOG --log-level $LOGLEVEL \
      --log-prefix "BLACKLIST: "
$IPT -A BLACKLIST -j DROP

# Populate Blacklist Ruleset using external file.
# Use INPUT/FORWARD/OUTPUT chains to ensure ALL traffic is looked at
if [ "$BLACKLIST" != "" ]; then
      echo "Loading list of blocked hosts from $BLACKLIST"
      for host in `grep -v ^# $BLACKLIST | awk '{print $1}'` ; do
            $IPT -A INPUT -s $host -j BLACKLIST
            $IPT -A INPUT -d $host -j BLACKLIST
            $IPT -A FORWARD -s $host -j BLACKLIST
            $IPT -A FORWARD -d $host -j BLACKLIST
            $IPT -A OUTPUT -s $host -j BLACKLIST
```

```
            $IPT -A OUTPUT -d $host -j BLACKLIST

            echo "Blacklisted: $host "
      done
fi

# IDENT Packets
# Log any packets with header "FW-IDENT:" and limit to 30/minute
$IPT -A INPUT -p tcp -i $EXTIF --dport 113 -m limit --limit \
      30/m -j LOG --log-level $LOGLEVEL --log-prefix "FW-IDENT: "
# Reject any packets with tcp-reset
$IPT -A INPUT -p tcp -i $EXTIF --dport 113 -j REJECT \
      --reject-with tcp-reset

# DNS traffic between External FW and External DNS/NTP server
$IPT -A INPUT -p udp -i $INTIF -s 39.2.1.49 -d 39.2.1.54 \
      --dport 53 -j ACCEPT
$IPT -A OUTPUT -p udp -o $INTIF -s 39.2.1.54 --sport 53 \
      -d 39.2.1.49 -j ACCEPT

# DNS traffic between External DNS/NTP server and ALL
$IPT -A FORWARD -p udp -i $INTIF -s 39.2.1.54 --sport 53 \
      --dport 53 -j ACCEPT
$IPT -A FORWARD -p udp -i $EXTIF --sport 53 -d 39.2.1.54 \
      --dport 53 -j ACCEPT

# HTTP/HTTPS traffic between ALL and Squid server
$IPT -A FORWARD -p tcp -i $EXTIF -d 39.2.1.55 --dport 80 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.55 --sport 80 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $EXTIF -d 39.2.1.55 --dport 443 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.55 --sport 443 \
      -j ACCEPT

# HTTPS traffic between ALL and WebMail server
$IPT -A FORWARD -p tcp -i $EXTIF -d 39.2.1.51 --dport 443 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.51 --sport 443 \
      -j ACCEPT

# SYSLOG traffic between Router and Ext Syslog server
$IPT -A FORWARD -p udp -i $EXTIF --sport 514 -d 39.2.1.62 \
      --dport 514 -j ACCEPT

# SMTP traffic between ALL and Mail Relay
$IPT -A FORWARD -p tcp -i $EXTIF -d 39.2.1.53 --dport 25 \
      -j ACCEPT
$IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.53 --sport 25 \
      -j ACCEPT

# NTP traffic between Router and Ext DNS/NTP server
$IPT -A FORWARD -p udp -i $EXTIF -s 39.2.1.45 -d 39.2.1.54 \
      --dport 123 -j ACCEPT
$IPT -A FORWARD -p udp -i $INTIF -s 39.2.1.54 --sport 123 \
      -d 39.2.1.45 -j ACCEPT
```

```
        # NTP traffic between Ext FW and Ext DNS/NTP server
        $IPT -A OUTPUT -p udp -o $INTIF -s 39.2.1.49 -d 39.2.1.54 \
              --dport 123 -j ACCEPT
        $IPT -A INPUT -p udp -i $INTIF -s 39.2.1.54 --sport 123 \
              -d 39.2.1.49 -j ACCEPT

        # SSH traffic between CIO/Admin Wksn and Ext FW
        $IPT -A INPUT -p tcp -i $INTIF -s 192.168.24.5 -d 39.2.1.49 \
              --dport 22 -j ACCEPT
        $IPT -A INPUT -p tcp -i $INTIF -s 192.168.24.4 -d 39.2.1.49 \
              --dport 22 -j ACCEPT
        $IPT -A OUTPUT -p tcp -o $INTIF -s 39.2.1.49 --sport 22 \
              -d 192.168.24.5 -j ACCEPT
        $IPT -A OUTPUT -p tcp -o $INTIF -s 39.2.1.49 --sport 22 \
              -d 192.168.24.4 -j ACCEPT

        # Zone Transfer between External DNS/NTP server and ISP DNS server
        $IPT -A FORWARD -p tcp -i $EXTIF -s 39.0.1.20 --sport 53 \
              -d 39.2.1.54 --dport 53 -j ACCEPT
        $IPT -A FORWARD -p tcp -i $INTIF -s 39.2.1.54 --sport 53 \
              -d 39.0.1.20 --dport 53 -j ACCEPT

        # LOG all packets that are dropped due to default FORWARD POLICY
        $IPT -A FORWARD -j LOG --log-level $LOGLEVEL \
              --log-prefix "FORWARD DROP:"

        # LOG all packets that are dropped due to default INPUT POLICY
        $IPT -A INPUT -j LOG --log-level $LOGLEVEL \
              --log-prefix "INPUT DROP:"

        # LOG all packets that are dropped due to default OUTPUT POLICY
        $IPT -A OUTPUT -j LOG --log-level $LOGLEVEL \
              --log-prefix "OUTPUT DROP:"

}


stop() {

        DEBUG=0

        echo "DEBUG=$DEBUG"

        # Send message to screen and syslog
        echo "Resetting built-in chains to the default ACCEPT policy"
        logger -p local3.info EXTERNAL NetFilter Firewall Stopped

        ####################################
        # Filter Module Rules
        ####################################

        # Disable syn cookie support
        echo 0 > /proc/sys/net/ipv4/tcp_syncookies

        # Disable logging of Martians (oddball addresses).
        echo 0 > /proc/sys/net/ipv4/conf/all/log_martians
```

```
        # Enable response to ping
        echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_all

        # Disable Dynamic IP Address support.
        echo 0 > /proc/sys/net/ipv4/ip_dynaddr

        # Disable anti spoofing support.
        echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter

        # Enable source routing support.
        echo 1 > /proc/sys/net/ipv4/conf/all/accept_source_route

        # Enable icmp redirect support.
        echo 1 > /proc/sys/net/ipv4/conf/all/accept_redirects

        # Keep IP forwarding enabled
        echo 1 > /proc/sys/net/ipv4/ip_forward


        ##################################
        # Flush,delete and zero existing chains
        ##################################

          $IPT -t filter -F
          $IPT -t nat -F
          $IPT -t mangle -F

        $IPT -X
        $IPT -Z

        ##################################
        # Set up default policies
        ##################################

        $IPT -t nat -P PREROUTING ACCEPT
        $IPT -t nat -P POSTROUTING ACCEPT
        $IPT -t nat -P OUTPUT ACCEPT

        $IPT -t mangle -P PREROUTING ACCEPT
        $IPT -t mangle -P INPUT ACCEPT
        $IPT -t mangle -P FORWARD ACCEPT
        $IPT -t mangle -P OUTPUT ACCEPT
        $IPT -t mangle -P POSTROUTING ACCEPT

        $IPT -t filter -P FORWARD ACCEPT
        $IPT -t filter -P INPUT ACCEPT
        $IPT -t filter -P OUTPUT ACCEPT

        # Script stopped message
        echo "NetFilter Firewall stopped..."

}

debug() {
        # Sets Debug Flag to 1 to turn on Debugging
        echo "Set Debug Flag"
```

```
        logger -p local3.info Set Debug Flag
        DEBUG=1
}

case "$1" in
  start)
        start
        ;;

  stop)
        stop
        ;;

  restart)
        # Basically perform a stop() then start()
        stop
        start
        ;;

  debug)
        # Sets up normal firewall with additional debugging
        stop
        debug
        start
        ;;

  status)

        echo "-----------------------------------"
        echo "NAT table"
        $IPT -t nat -nvxL --line-number
        echo "-----------------------------------"
        echo "Mangle table"
        $IPT -t mangle -vxL --line-number
        echo "-----------------------------------"
        echo "Filter table:"
        $IPT -t filter -nvxL --line-number
        ;;

  panic)

        echo "Initating panic mode"
        echo "Changing target policies to DROP "
        logger -p local3.info EXTERNAL NetFilter Firewall intiated PANIC mode

        $IPT -t filter -P INPUT DROP
        $IPT -t filter -P FORWARD DROP
        $IPT -t filter -P OUTPUT DROP
        $IPT -t nat -P PREROUTING DROP
        $IPT -t nat -P POSTROUTING DROP
        $IPT -t nat -P OUTPUT DROP
        $IPT -t mangle -P PREROUTING DROP
        $IPT -t mangle -P OUTPUT DROP
        $IPT -t mangle -P POSTROUTING DROP
        $IPT -t mangle -P INPUT DROP
        $IPT -t mangle -P FORWARD DROP
```

```
      echo "Flushing all chains"
        $IPT -t filter -F
        $IPT -t nat -F
        $IPT -t mangle -F

      echo "Removing user defined chains"
        $IPT -X

      # Disable IP Forward support.
      echo 0 > /proc/sys/net/ipv4/ip_forward

        ;;
  *)
      echo " Usage: $0 {start|stop|restart|debug|status|panic} "
      exit 1

esac
exit 0
------------------ end of external firewall configuration -----------------
```

Appendix C – Internal Firewall/VPN Gateway Configuration

```
---------- start of internal firewall/VPN gateway configuration ----------
#!/bin/sh
#
# Startup script for NetFilter firewall
# Replacement for the default iptables script
# Requires iptables-1.2.8 from http://www.netfilter.org
#
# 030622 - written by GDW
#
# Setup global variables
IPT=/usr/local/sbin/iptables
LOGLEVEL=notice
DEBUG=0
# Define Interfaces – changes between test and production environments
EXTIF=eth0
SRVIF=vmnet1
SECIF=vmnet2
WEBIF=vmnet3


# Ensure iptables command is available
if [ ! -x $IPT ]; then
      echo "ERROR: Cannot locate iptables command."
      exit 0
fi

# Ensure ipchains command is not running
if  /sbin/lsmod 2>/dev/null |grep -q ipchains ; then
      echo "ERROR: Cannot run both ipchains and iptables"
      exit 0
fi

start() {
```

```
# Log to screen and syslog the start message
echo "Starting NetFilter Firewall"
logger -p local3.info NetFilter Firewall Started

# Enable syn cookie support
echo 1 > /proc/sys/net/ipv4/tcp_syncookies

# Enable logging of Martians (oddball addresses).
echo 1 > /proc/sys/net/ipv4/conf/all/log_martians

# Enable IP Forward support.
echo 1 > /proc/sys/net/ipv4/ip_forward

# Disable response to ping
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all

# Disable response to Broadcasts (to keep from being SMURF amplifier)
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# Enable Dynamic IP Address support.
echo 1 > /proc/sys/net/ipv4/ip_dynaddr

# Enable anti spoofing support.
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter

# Disable source routing support.
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route

# Disable icmp redirect support.
echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects

##################################
# Flush,delete and zero existing chains
##################################

$IPT -t filter -F
$IPT -t nat -F
$IPT -t mangle -F
$IPT -X
$IPT -Z

##################################
# Set up default policies
##################################

$IPT -t nat -P PREROUTING ACCEPT
$IPT -t nat -P POSTROUTING ACCEPT
$IPT -t nat -P OUTPUT ACCEPT

$IPT -t mangle -P PREROUTING ACCEPT
$IPT -t mangle -P INPUT ACCEPT
$IPT -t mangle -P FORWARD ACCEPT
$IPT -t mangle -P OUTPUT ACCEPT
$IPT -t mangle -P POSTROUTING ACCEPT
```

```
$IPT -t filter -P FORWARD DROP
$IPT -t filter -P INPUT DROP
$IPT -t filter -P OUTPUT DROP

###################################
# nat Module Rules
###################################

# This table is consulted when a packet which creates a new
# connection is encountered. Use PREROUTING to drop unwanted new
# traffic before passing on to FORWARD or INPUT Filters.

# Block Reserved/Private Addresses coming in the external interface.
# Data pulled from http://www.iana.org/assignments/ipv4-address-space
# Dropped 39.0.0.0/8 from Reserved list since I have given part of
# this address space to GIAC Enterprises

IANA_RESERVED="0 1 2 5 7 23 27 31 36 37 41 42 58 59 70 71 72 73 \
74 75 76 77 78 79 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 \
99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 \
116 117 118 119 120 121 122 123 124 125 126 127 173 174 175 176 177 \
178 179 180 181 182 183 184 185 186 187 189 190 197 223 240 241 242 \
243 244 245 246 247 248 249 250 251 252 253 254 255"

for NET in $IANA_RESERVED; do
      $IPT -t nat -A PREROUTING -i $EXTIF -s $NET.0.0.0/8 -j DROP
done

IANA_MULTICAST="224 225 226 227 228 229 230 231 232 233 234 235 236 \
237 238 239"
for NET in $IANA_MULTICAST; do
      $IPT -t nat -A PREROUTING -i $EXTIF -s $NET.0.0.0/8 -j DROP
done

# IANA PRIVATE IPv4 address space
# RFC 1918 Private Network
$IPT -t nat -A PREROUTING -i $EXTIF -s 10.0.0.0/8 -j DROP
# RFC 1918 Private Network
$IPT -t nat -A PREROUTING -i $EXTIF -s 172.16/12 -j DROP
# RFC 1918 Private Network
$IPT -t nat -A PREROUTING -i $EXTIF -s 192.168.0.0/16 -j DROP
# RFC 3068 IANA Special Use
$IPT -t nat -A PREROUTING -i $EXTIF -s 192.88.99.0/24 -j DROP

###################################
# Mangle Module Rules.
###################################

# Rules to mangle TOS values of packets routed through the firewall.
# Based on RFC 1060/1349

# TOS stuff: (type: iptables -m tos -h)
# Minimize-Delay 16 (0x10)
# Maximize-Throughput 8 (0x08)
# Maximize-Reliability 4 (0x04)
# Minimize-Cost 2 (0x02)
# Normal-Service 0 (0x00)
```

```
$IPT -t mangle -A PREROUTING -p tcp --dport 22 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 25 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p udp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 80 -j TOS --set-tos 8

###################################
# Filter Module Rules
###################################

# Setup debugging for tracking traffic flows
if [ $DEBUG = 1 ]; then
      echo "Extra Debug logging enabled"
      $IPT -A INPUT -i lo -j LOG --log-level $LOGLEVEL \
            --log-prefix "local-in: "
      $IPT -A OUTPUT -o lo -j LOG --log-level $LOGLEVEL \
            --log-prefix "local-out: "
      $IPT -A INPUT -p 50 -i $EXTIF -j LOG --log-level $LOGLEVEL \
            --log-prefix "ESP-in: "
      $IPT -A OUTPUT -p 50 -o $EXTIF -j LOG --log-level $LOGLEVEL \
            --log-prefix "ESP-out: "
      $IPT -A INPUT -p udp -i $EXTIF --sport 500 --dport 500 -j LOG \
            --log-level $LOGLEVEL --log-prefix "IKE-in: "
      $IPT -A OUTPUT -p udp -o $EXTIF --sport 500 --dport 500 -j LOG \
            --log-level $LOGLEVEL --log-prefix "IKE-out: "
      $IPT –A FORWARD –I ipsec+ -j LOG –log-level $LOGLEVEL \
            --log-prefix "IPSec: "
      $IPT -A FORWARD -p udp --sport 53 --dport 53 -j LOG \
            --log-level $LOGLEVEL --log-prefix "DNS: "
      $IPT -A FORWARD -p udp --sport 123 --dport 123 -j LOG \
            --log-level $LOGLEVEL --log-prefix "NTP: "
      $IPT -A FORWARD -p udp --sport 514 --dport 514 -j LOG \
            --log-level $LOGLEVEL --log-prefix "SYSLOG: "
      $IPT -A FORWARD -p tcp --dport 80 -j LOG \
            --log-level $LOGLEVEL --log-prefix "HTTP: "
      $IPT -A FORWARD -p tcp --dport 443 -j LOG \
            --log-level $LOGLEVEL --log-prefix "HTTPS: "
      $IPT -A FORWARD -p tcp --dport 25 -j LOG \
            --log-level $LOGLEVEL --log-prefix "SMTP: "
      $IPT -A FORWARD -p tcp --dport 143 -j LOG \
            --log-level $LOGLEVEL --log-prefix "IMAP: "
      $IPT -A FORWARD -p tcp --dport 3306 -j LOG \
            --log-level $LOGLEVEL --log-prefix "MySQL: "
      $IPT -A FORWARD -p tcp --dport 636 -j LOG \
            --log-level $LOGLEVEL --log-prefix "LDAPS: "
      $IPT -A FORWARD -p tcp --dport 22 -j LOG \
            --log-level $LOGLEVEL --log-prefix "SSH: "
fi

# Allow traffic on loopback interface
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT

# Allow IPsec into external interface
# IKE Negotiations
$IPT -A INPUT -p udp -i $EXTIF --sport 500 --dport 500 -j ACCEPT
```

```
$IPT -A OUTPUT -p udp -o $EXTIF --sport 500 --dport 500 -j ACCEPT

# ESP encryption and authentication
$IPT -A INPUT -p 50 -i $EXTIF -j ACCEPT
$IPT -A OUTPUT -p 50 -o $EXTIF -j ACCEPT

# Forward traffic on ipsec interfaces
$IPT –A FORWARD –i ipsec+ -j ACCEPT
$IPT –A FORWARD –o ipsec+ -j ACCEPT

# Setup ICMP traffic chain
$IPT -N ICMP_TRAFFIC

# Log traffic by type and drop
$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 0 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Echo Reply: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 8 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Echo Request: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 3 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Dest Unreachable: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 4 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Source Quench: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 5 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Redirect: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 9 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Router Advertisement: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 10 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Router Selection: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 11 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Time Exceeded: "

$IPT -A ICMP_TRAFFIC -p icmp --icmp-type 30 -j LOG \
      --log-level $LOGLEVEL \
      --log-prefix "ICMP-Traceroute: "

$IPT -A ICMP_TRAFFIC -j DROP

# Push all ICMP traffic to ICMP chain
$IPT -A INPUT -p icmp -j ICMP_TRAFFIC
$IPT -A FORWARD -p icmp -j ICMP_TRAFFIC
$IPT -A OUTPUT -p icmp -j ICMP_TRAFFIC
```

```
# DNS traffic between Security subnet and Internal DNS/NTP server
$IPT -A FORWARD -i $SECIF -p udp -s 192.168.24.0/24 -d 192.168.23.6 \
      --dport 53 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.6 -d 192.168.24.0/24 \
      --dport 53 -j ACCEPT

# DNS traffic betwen Internal DNS/NTP server & External DNS/NTP server
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.6 --sport 53 \
      -d 39.2.1.54 --dport 53 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p udp -s 39.2.1.54 --sport 53 \
      -d 192.168.23.6 --dport 53 -j ACCEPT

# NTP traffic between Security subnet and Internal DNS/NTP server
$IPT -A FORWARD -i $SECIF -p udp -s 192.168.24.0/24 -d 192.168.23.6 \
      --dport 123 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.6 –sport 123 \
      -d 192.168.24.0/24 -j ACCEPT

# NTP traffic between Webserver and Internal DNS/NTP server
$IPT -A FORWARD -i $WEBIF -p udp -s 192.168.25.2 -d 192.168.23.6
      --dport 123 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.6 –sport 123 \
      -d 192.168.25.2 -j ACCEPT

# NTP traffic between Internal FW/VPN GW and Internal DNS/NTP server
$IPT -A INPUT -i $SRVIF -p udp -s 192.168.23.6 –sport 123 \
      –d 192.168.23.1 -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p udp -s 192.168.23.1 –d 192.168.23.6 \
      --dport 123 -j ACCEPT

# SYSLOG traffic between syslog server and Server subnet
$IPT -A FORWARD -i $SRVIF -p udp -s 192.168.23.0/24 --sport 514 \
      -d 192.168.24.2 --dport 514 -j ACCEPT

# SYSLOG traffic between syslog server and Webserver
$IPT -A FORWARD -i $WEBIF -p udp -s 192.168.25.2 --sport 514 \
      -d 192.168.24.2 --dport 514 -j ACCEPT

# SYSLOG traffic between Internal FW/VPN GW and syslog server
$IPT -A OUTPUT -o $SECIF -p udp --sport 514 -d 192.168.24.2 \
      --dport 514 -j ACCEPT

# LDAPS traffic between VPN users and LDAP server
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.4 --sport 636 \
      -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.4 --dport 636 \
      -j ACCEPT

# HTTP/HTTPS traffic between VPN users and Intranet server
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.5 --sport 80 \
      -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.5 --dport 80 \
      -j ACCEPT
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.5 --sport 443 \
      -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.5 --dport 443 \
      -j ACCEPT
```

```
# SMTP/IMAP traffic between VPN users and Mail server
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.3 --sport 143 \
      -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.3 --dport 143 \
      -j ACCEPT
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.3 --sport 25 \
      -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.3 --dport 25 \
      -j ACCEPT

# MySQL traffic between VPN users and Database server
$IPT -A INPUT -i $SRVIF -p tcp -s 192.168.23.2 --sport 3306 \
      -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p tcp -d 192.168.23.2 --dport 3306 \
      -j ACCEPT

# NTP/DNS traffic between VPN users and Internal NTP/DNS server
$IPT -A INPUT -i $SRVIF -p udp -s 192.168.23.6 --sport 53 \
      -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p udp -d 192.168.23.6 --dport 53 \
      -j ACCEPT
$IPT -A INPUT -i $SRVIF -p udp -s 192.168.23.6 --sport 123 \
      -j ACCEPT
$IPT -A OUTPUT -o $SRVIF -p udp -d 192.168.23.6 --dport 123 \
      -j ACCEPT

# SSH traffic between VPN users and CIO/Admin workstation
$IPT -A INPUT -i $SECIF -p tcp -s 192.168.24.4 --sport 22 \
      -j ACCEPT
$IPT -A OUTPUT -o $SECIF -p tcp -d 192.168.24.4 --dport 22 \
      -j ACCEPT
$IPT -A INPUT -i $SECIF -p tcp -s 192.168.24.5 --sport 22 \
      -j ACCEPT
$IPT -A OUTPUT -o $SECIF -p tcp -d 192.168.24.5 --dport 22 \
      -j ACCEPT

# LDAPS traffic between webserver and LDAP server
$IPT -A FORWARD -i $WEBIF -p tcp -s 192.168.25.2 \
      -d 192.168.23.4 --dport 636 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.4 --sport 636 \
      -d 192.168.25.2 -j ACCEPT

# MySQL traffic between webserver and database server
$IPT -A FORWARD -i $WEBIF -p tcp -s 192.168.25.2 \
      -d 192.168.23.2 --dport 3306 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.2 --sport 3306 \
      -d 192.168.25.2 -j ACCEPT

# SMTP traffic between Mail server and Mail Relay
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.53 --sport 25 \
      -d 192.168.24.3 --dport 25 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.24.3 --sport 25 \
      -d 39.2.1.53 --dport 25 -j ACCEPT

# LDAPS traffic between WebMail server and LDAP server
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 \
```

```
      -d 192.168.23.4 --dport 636 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.4 --sport 636 \
      -d 39.2.1.51 -j ACCEPT

# SMTP/IMAP traffic between WebMail server and Mailserver
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 --sport 25 \
      -d 192.168.24.3 --dport 25 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.24.3 --sport 25 \
      -d 39.2.1.51 --dport 25 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 \
      -d 192.168.24.3 --dport 143 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.24.3 --sport 143 \
      -d 39.2.1.51 -j ACCEPT

# SSH traffic between CIO/Admin workstations and Server subnet
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 192.168.23.0/24 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.0/24 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 192.168.23.0/24 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $SRVIF -p tcp -s 192.168.23.0/24 --sport 22 \
      -d 192.168.24.5 -j ACCEPT

# SSH traffic between CIO/Admin workstations and webserver
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 192.168.25.2 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $WEBIF -p tcp -s 192.168.25.2 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 192.168.25.2 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $WEBIF -p tcp -s 192.168.25.2 --sport 22 \
      -d 192.168.24.5 -j ACCEPT

# SSH traffic between CIO/Admin workstations and Internal FW/VPN GW
$IPT -A INPUT -i $SECIF -p tcp -s 192.168.24.4 \
      -d 192.168.24.1 --dport 22 -j ACCEPT
$IPT -A OUTPUT -o $SECIF -p tcp -s 192.168.24.1 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A INPUT -i $SECIF -p tcp -s 192.168.24.5 \
      -d 192.168.24.1 --dport 22 -j ACCEPT
$IPT -A OUTPUT -o $SECIF -p tcp -s 192.168.24.1 --sport 22 \
      -d 192.168.24.5 -j ACCEPT

# SSH traffic between CIO/Admin workstations and WebMail server
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.51 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.51 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.51 --sport 22 \
      -d 192.168.24.5 -j ACCEPT

# SSH traffic between CIO/Admin workstations and Squid server
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.55 --dport 22 -j ACCEPT
```

```
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.55 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 22 \
      -d 192.168.24.5 -j ACCEPT

# SSH traffic between CIO/Admin workstations and Mail Relay server
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.53 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.53 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.53 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.53 --sport 22 \
      -d 192.168.24.5 -j ACCEPT

# SSH traffic between CIO/Admin workstations & External DNS/NTP server
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.54 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.54 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.54 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.54 --sport 22 \
      -d 192.168.24.5 -j ACCEPT

# SSH traffic between CIO/Admin workstations and External FW
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.49 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.49 --sport 22 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.49 --dport 22 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.49 --sport 22 \
      -d 192.168.24.5 -j ACCEPT

# HTTP/HTTPS traffic between CIO/Admin workstations and proxy server
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.55 --dport 8080 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.55 –dport 8080 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 8080 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 8080 \
      -d 192.168.24.5 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.4 \
      -d 39.2.1.55 --dport 8080 -j ACCEPT
$IPT -A FORWARD -i $SECIF -p tcp -s 192.168.24.5 \
      -d 39.2.1.55 --dport 8080 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 8080 \
      -d 192.168.24.4 -j ACCEPT
$IPT -A FORWARD -i $EXTIF -p tcp -s 39.2.1.55 --sport 8080 \
      -d 192.168.24.5 -j ACCEPT

# LOG all packets that are dropped due to default FORWARD POLICY
$IPT -A FORWARD -j LOG --log-level $LOGLEVEL \
```

```
            --log-prefix "FORWARD DROP:"

      # LOG all packets that are dropped due to default INPUT POLICY
      $IPT -A INPUT -j LOG --log-level $LOGLEVEL \
            --log-prefix "INPUT DROP:"

      # LOG all packets that are dropped due to default OUTPUT POLICY
      $IPT -A OUTPUT -j LOG --log-level $LOGLEVEL \
            --log-prefix "OUTPUT DROP:"

      # Log to screen NetFilter stopping message
      echo "NetFilter Firewall started successfully"

}

stop() {

      DEBUG=0

      # Send message to screen and syslog
      echo "Resetting built-in chains to the default ACCEPT policy"
      logger -p local3.info NetFilter Firewall Stopped

      ####################################
      # Filter Module Rules
      ####################################

      # Disable syn cookie support
      echo 0 > /proc/sys/net/ipv4/tcp_syncookies

      # Disable logging of Martians (oddball addresses).
      echo 0 > /proc/sys/net/ipv4/conf/all/log_martians

      # Enable response to ping
      echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_all

      # Disable Dynamic IP Address support.
      echo 0 > /proc/sys/net/ipv4/ip_dynaddr

      # Disable anti spoofing support.
      echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter

      # Enable source routing support.
      echo 1 > /proc/sys/net/ipv4/conf/all/accept_source_route

      # Enable icmp redirect support.
      echo 1 > /proc/sys/net/ipv4/conf/all/accept_redirects

      # Keep IP forwarding enabled
      echo 1 > /proc/sys/net/ipv4/ip_forward

      ###################################
      # Flush,delete and zero existing chains
      ###################################

      $IPT -t filter -F
      $IPT -t nat -F
```

```
        $IPT -t mangle -F

        $IPT -X
        $IPT -Z

        ###################################
        # Set up default policies
        ###################################

        $IPT -t nat -P PREROUTING ACCEPT
        $IPT -t nat -P POSTROUTING ACCEPT
        $IPT -t nat -P OUTPUT ACCEPT

        $IPT -t mangle -P PREROUTING ACCEPT
        $IPT -t mangle -P INPUT ACCEPT
        $IPT -t mangle -P FORWARD ACCEPT
        $IPT -t mangle -P OUTPUT ACCEPT
        $IPT -t mangle -P POSTROUTING ACCEPT

        $IPT -t filter -P FORWARD ACCEPT
        $IPT -t filter -P INPUT ACCEPT
        $IPT -t filter -P OUTPUT ACCEPT

        # Script stopped message
        echo "NetFilter Firewall stopped..."

}

debug() {
        # Sets Debug Flag to 1 to turn on Debugging
        echo "Enable troubleshooting via Debug Flag"
        logger -p local3.info NetFilter Firewall Started with Debug Flag
        DEBUG=1

}

case "$1" in
  start)
        start
        ;;

  stop)
        stop
        ;;

  restart)
        # Basically perform a stop() then start()
        stop
        start
        ;;

  debug)
        # Sets up normal firewall with additional debugging
        stop
        debug
        start
        ;;
```

```
  status)
      echo "-----------------------------------"
      echo "NAT table"
      $IPT -t nat -nvxL --line-number
      echo "-----------------------------------"
      echo "Mangle table"
      $IPT -t mangle -vxL --line-number
      echo "-----------------------------------"
      echo "Filter table:"
      $IPT -t filter -nvxL --line-number
      ;;

  panic)
      echo "Initating panic mode"
      echo "Changing target policies to DROP "

      $IPT -t filter -P INPUT DROP
      $IPT -t filter -P FORWARD DROP
      $IPT -t filter -P OUTPUT DROP
      $IPT -t nat -P PREROUTING DROP
      $IPT -t nat -P POSTROUTING DROP
      $IPT -t nat -P OUTPUT DROP
      $IPT -t mangle -P PREROUTING DROP
      $IPT -t mangle -P OUTPUT DROP
      $IPT -t mangle -P POSTROUTING DROP
      $IPT -t mangle -P INPUT DROP
      $IPT -t mangle -P FORWARD DROP

      echo "Flushing all chains"
        $IPT -t filter -F
        $IPT -t nat -F
        $IPT -t mangle -F

      echo "Removing user defined chains"
        $IPT -X

      # Disable IP Forward support.
      echo 0 > /proc/sys/net/ipv4/ip_forward
        ;;

  *)
      echo " Usage: $0 {start|stop|restart|debug|status|panic} "
      exit 1

esac
exit 0

---------- end of internal firewall/VPN gateway configuration -----------
```

Appendix C – Server Firewall Configuration

```
---------- start of Server firewall configuration -----------
#!/bin/sh
#
```

```
# Startup script for NetFilter firewall for Servers
# Replacement for the default iptables script
#
# 030622 - written by GDW
#
# Setup global variables
IPT=/sbin/iptables
LOGLEVEL=notice
DEBUG=0
# Define Interfaces
EXTIF=eth0

# Ensure iptables command is available
if [ ! -x $IPT ]; then
      echo "ERROR: Cannot locate iptables command."
      exit 0
fi

# Ensure ipchains command is not running
if  /sbin/lsmod 2>/dev/null |grep -q ipchains ; then
      echo "ERROR: Cannot run both ipchains and iptables"
      exit 0
fi

start() {

      # Log to screen and syslog the start message
      echo "Starting NetFilter Firewall"
      logger -p local3.info NetFilter Firewall Started

      # Enable syn cookie support
      echo 1 > /proc/sys/net/ipv4/tcp_syncookies

      # Enable logging of Martians (oddball addresses).
      echo 1 > /proc/sys/net/ipv4/conf/all/log_martians

      # Enable IP Forward support.
      echo 1 > /proc/sys/net/ipv4/ip_forward

      # Disable response to ping
      echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all

      # Disable response to Broadcasts (to keep from being SMURF amplifier)
      echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

      # Enable Dynamic IP Address support.
      echo 1 > /proc/sys/net/ipv4/ip_dynaddr

      # Enable anti spoofing support.
      echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter

      # Disable source routing support.
      echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route

      # Disable icmp redirect support.
      echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects
```

```
#################################
# Flush,delete and zero existing chains
#################################

  $IPT -t filter -F
  $IPT -t nat -F
  $IPT -t mangle -F

$IPT -X
$IPT -Z

#################################
# Set up default policies
#################################

$IPT -t nat -P PREROUTING ACCEPT
$IPT -t nat -P POSTROUTING ACCEPT
$IPT -t nat -P OUTPUT ACCEPT

$IPT -t mangle -P PREROUTING ACCEPT
$IPT -t mangle -P INPUT ACCEPT
$IPT -t mangle -P FORWARD ACCEPT
$IPT -t mangle -P OUTPUT ACCEPT
$IPT -t mangle -P POSTROUTING ACCEPT

$IPT -t filter -P FORWARD DROP
$IPT -t filter -P INPUT DROP
$IPT -t filter -P OUTPUT ACCEPT

#################################
# nat Module Rules
#################################

# This table is consulted when a packet which creates a new
# connection is encountered. Use PREROUTING to drop unwanted new
# traffic before passing on to FORWARD or INPUT Filters.


#################################
# Mangle Module Rules.
#################################

# Rules to mangle TOS values of packets routed through the firewall.
# Based on RFC 1060/1349

# TOS stuff: (type: iptables -m tos -h)
# Minimize-Delay 16 (0x10)
# Maximize-Throughput 8 (0x08)
# Maximize-Reliability 4 (0x04)
# Minimize-Cost 2 (0x02)
# Normal-Service 0 (0x00)

$IPT -t mangle -A PREROUTING -p tcp --dport 22 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 25 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p udp --dport 53 -j TOS --set-tos 16
$IPT -t mangle -A PREROUTING -p tcp --dport 80 -j TOS --set-tos 8
```

```
###################################
# Filter Module Rules
###################################

# Setup debugging for tracking traffic flows
if [ $DEBUG = 1 ]; then
      echo "Extra Debug logging enabled"
      $IPT -A INPUT -i lo -j LOG --log-level $LOGLEVEL \
            --log-prefix "local-in: "
      $IPT -A OUTPUT -o lo -j LOG --log-level $LOGLEVEL \
            --log-prefix "local-out: "
      $IPT -A INPUT -p udp --sport 53 --dport 53 -j LOG \
            --log-level $LOGLEVEL --log-prefix "DNS: "
      $IPT -A INPUT -p udp --sport 123 --dport 123 -j LOG \
            --log-level $LOGLEVEL --log-prefix "NTP: "
      $IPT -A INPUT -p udp --sport 514 --dport 514 -j LOG \
            --log-level $LOGLEVEL --log-prefix "SYSLOG: "
      $IPT -A INPUT -p tcp --dport 636 -j LOG \
            --log-level $LOGLEVEL --log-prefix "LDAPS: "
      $IPT -A INPUT -p tcp --dport 22 -j LOG \
            --log-level $LOGLEVEL --log-prefix "SSH: "
fi

## End additional logging

# Allow traffic on loopback interface
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT

# Allow incoming traffic if is part of an previous connection
$IPT -A INPUT -p TCP -m state --state ESTABLISHED,RELATED -j ACCEPT

# DNS traffic between LDAP server and Internal DNS/NTP server
$IPT -A OUTPUT -o $EXTIF -p udp -s 192.168.23.4 -d 192.168.23.6 \
      --dport 53 -j ACCEPT
$IPT -A INPUT -i $EXTIF -p udp -s 192.168.23.6 --sport 53 \
      -d 192.168.23.4 -j ACCEPT

# NTP traffic between LDAP server and Internal DNS/NTP server
$IPT -A OUTPUT -o $EXTIF -p udp -s 192.168.23.4 -d 192.168.23.6 \
      --dport 123 -j ACCEPT
$IPT -A INPUT -i $EXTIF -p udp -s 192.168.23.6 --sport 123 \
      -d 192.168.23.4 -j ACCEPT

# SYSLOG traffic between LDAP server and Syslog server
$IPT -A OUTPUT -o $EXTIF -p udp -s 192.168.23.4 -d 192.168.24.2 \
      --dport 514 -j ACCEPT

# LDAPS traffic for LDAP server
$IPT -A INPUT -i $EXTIF -p tcp -d 192.168.23.4 \
      --dport 636 -j ACCEPT
$IPT -A OUTPUT -o $EXTIF -p tcp -s 192.168.23.4 --sport 636 \
      -j ACCEPT

# SSH traffic between LDAP server and CIO/Admin workstation
$IPT -A INPUT -i $EXTIF -p tcp -s 192.168.24.4 -d 192.168.23.4 \
```

```
            --dport 22 -j ACCEPT
      $IPT -A OUTPUT -o $EXTIF -p tcp -s 192.168.23.4 --sport 22 \
            -d 192.168.24.4 -j ACCEPT
      $IPT -A INPUT -i $EXTIF -p tcp -s 192.168.24.5 -d 192.168.23.4 \
            --dport 22 -j ACCEPT
      $IPT -A OUTPUT -o $EXTIF -p tcp -s 192.168.23.4 --sport 22 \

      # LOG all packets that are dropped due to default FORWARD POLICY
      $IPT -A FORWARD -j LOG --log-level $LOGLEVEL \
            --log-prefix "FORWARD DROP:"

      # LOG all packets that are dropped due to default INPUT POLICY
      $IPT -A INPUT -j LOG --log-level $LOGLEVEL \
            --log-prefix "INPUT DROP:"

      # Log to screen NetFilter stopping message
      echo "NetFilter Firewall started successfully"

}

stop() {

      DEBUG=0

      # Send message to screen and syslog
      echo "Resetting built-in chains to the default ACCEPT policy"
      logger -p local3.info NetFilter Firewall Stopped

      ####################################
      # Filter Module Rules
      ####################################

      # Disable syn cookie support
      echo 0 > /proc/sys/net/ipv4/tcp_syncookies

      # Disable logging of Martians (oddball addresses).
      echo 0 > /proc/sys/net/ipv4/conf/all/log_martians

      # Enable response to ping
      echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_all

      # Disable Dynamic IP Address support.
      echo 0 > /proc/sys/net/ipv4/ip_dynaddr

      # Disable anti spoofing support.
      echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter

      # Enable source routing support.
      echo 1 > /proc/sys/net/ipv4/conf/all/accept_source_route

      # Enable icmp redirect support.
      echo 1 > /proc/sys/net/ipv4/conf/all/accept_redirects

      # Keep IP forwarding enabled
      echo 1 > /proc/sys/net/ipv4/ip_forward
```

```
        ##################################
        # Flush,delete and zero existing chains
        ##################################

          $IPT -t filter -F
          $IPT -t nat -F
          $IPT -t mangle -F

        $IPT -X
        $IPT -Z

        ##################################
        # Set up default policies
        ##################################

        $IPT -t nat -P PREROUTING ACCEPT
        $IPT -t nat -P POSTROUTING ACCEPT
        $IPT -t nat -P OUTPUT ACCEPT

        $IPT -t mangle -P PREROUTING ACCEPT
        $IPT -t mangle -P INPUT ACCEPT
        $IPT -t mangle -P FORWARD ACCEPT
        $IPT -t mangle -P OUTPUT ACCEPT
        $IPT -t mangle -P POSTROUTING ACCEPT

        $IPT -t filter -P FORWARD ACCEPT
        $IPT -t filter -P INPUT ACCEPT
        $IPT -t filter -P OUTPUT ACCEPT

        # Script stopped message
        echo "NetFilter Firewall stopped..."

}

debug() {
        # Sets Debug Flag to 1 to turn on Debugging
        echo "Enable troubleshooting via Debug Flag"
        logger -p local3.info NetFilter Firewall Started with Debug Flag
        DEBUG=1
}

case "$1" in
  start)
        start
        ;;

  stop)
        stop
        ;;

  restart)
        # Basically perform a stop() then start()
        stop
        start
        ;;

  debug)
```

```
        # Sets up normal firewall with additional debugging
        stop
        debug
        start
        ;;

    status)

        echo "-----------------------------------"
        echo "NAT table"
        $IPT -t nat -nvxL --line-number
        echo "-----------------------------------"
        echo "Mangle table"
        $IPT -t mangle -vxL --line-number
        echo "-----------------------------------"
        echo "Filter table:"
        $IPT -t filter -nvxL --line-number
        ;;

    panic)

        echo "Initating panic mode"
        echo "Changing target policies to DROP "

        $IPT -t filter -P INPUT DROP
        $IPT -t filter -P FORWARD DROP
        $IPT -t filter -P OUTPUT DROP
        $IPT -t nat -P PREROUTING DROP
        $IPT -t nat -P POSTROUTING DROP
        $IPT -t nat -P OUTPUT DROP
        $IPT -t mangle -P PREROUTING DROP
        $IPT -t mangle -P OUTPUT DROP
        $IPT -t mangle -P POSTROUTING DROP
        $IPT -t mangle -P INPUT DROP
        $IPT -t mangle -P FORWARD DROP

        echo "Flushing all chains"
          $IPT -t filter -F
          $IPT -t nat -F
          $IPT -t mangle -F

        echo "Removing user defined chains"
          $IPT -X

        # Disable IP Forward support.
        echo 0 > /proc/sys/net/ipv4/ip_forward

          ;;
    *)
        echo " Usage: $0 {start|stop|restart|debug|status|panic} "
        exit 1

esac
exit 0
---------- end of Server firewall configuration -----------
```

References

Bauer, Mick. "Stealthful Sniffing, Intrusion Detection and Logging." Linux Journal October 2002:34-40. (June 28, 2003)

Dunston, Duane. "Remote Syslog with MySQL and PHP." LinuxSecurity.Com. 13 February 2003 URL:http://www.linuxsecurity.com/feature_stories/feature_story-138.html. (June 28, 2003)

Berk, Vincent and Marion Bates. HOWTO-Jeanne, redirector for Squid Reverse Proxy. 2001 URL: http://www.ists.dartmouth.edu/IRIA/projects/jeanne/howto.pdf. (July 5, 2003)

Nathan, Jeff and Brian Caswell. Full Duplex Tap for Span Port. URL:http://www.snort.org/docs/100Mb_tapping1.pdf (July 3, 2003)

[1] Diagram reproduced using Visio and pulled from:
Novak, Judy. TCP/IP for Firewalls Course Material – Firewalls, Perimeter Protection and VPNs Track. SANS Institute. 2003. page 2-27.

Russell, Rusty. Linux netfilter Hacking HOWTO. 04 May 2001. URL:http://www.linuxguruz.com/iptables/howto/netfilter-hacking-HOWTO.html (June 30, 2003)

Novak, Judy. TCP/IP for Firewalls Course Material – Firewalls, Perimeter Protection and VPNs Track. SANS Institute. 2003.

Brenton, Chris, Tanya Baccam and Stephen Northcutt. Packet Filters Course Material – Firewalls, Perimeter Protection and VPNs Track. SANS Institute. 2003.

Flickenger, Rob. Linux Server Hacks. January 2003. URL: http://hacks.oreilly.com/pub/h/45. (July 8, 2003)

Andreasson, Oskar. Iptables Tutorial 1.1.19. 2001. URL:http://www.faqs.org/docs/iptables/index.html. (July 8, 2003)

Naidu, Krishni. Firewall Checklist. URL:http://www.sans.org/score/checklists/FirewallChecklist.doc (July 10, 2003)

Antoine, Vanessa, Patricia Boamajian, Daniel Duesterhaus, et.al. Router Security Guidance Activity of the System and Network Attack Center (SNAC). National Security Agency. 2002.

Spitzner, Lance. Auditing Your Firewall Setup. December 12, 2000. URL: http://www.spitzner.net/audit.html (July 12, 2003)

Curtin, Matt and Marcus Ranum. Internet Firewalls: Frequently Asked Questions. December 1, 2000. URL: http://www.interhack.net/pubs/fwfaq/ (July 12, 2003)

Graham, Robert. FAQ: Firewall Forensics (What am I seeing?). January 2003. URL: http://www.robertgraham.com/pubs/firewall-seen.html (July 12, 2003)

Wack, John, Ken Cutler and Jamie Pole. Guidelines on Firewalls and Firewall Policy. Gaithersburg: National Institute of Standards and Technology, 2002.

Carlson, James. "Don't Let Your Cookies Crumble." SANS GIAC Certified Firewall Analyst (GCFW) Practical Assignment. 12 April 2003. URL: http://www.giac.org/practical/GCFW/James_Carlson_GCFW.pdf (1 August 2003).

[2] Diagram reproduced using Visio and pulled from:
Carlson, James. "Don't Let Your Cookies Crumble." SANS GIAC Certified Firewall Analyst (GCFW) Practical Assignment. 12 April 2003. URL: http://www.giac.org/practical/GCFW/James_Carlson_GCFW.pdf page 7.

Barlow, Jason and Woody Thrower. "TFN2K – An Analysis." 7 March 2000. URL: http://www.ussrback.com/docs/distributed/TFN2k_Analysis-1.3.txt. (1 August 2003).

General resource websites (many found in paper)
http://www.rootkit.com
http://www.linuxguruz.com/iptables/
http://www.cisco.com
http://www.insecure.org/nmap/