



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

GCFW PRACTICAL ASSIGNMENT

Firewalls, Perimeter Protection and VPN's

Version 1.8

Renett Chan

February 2003

Abstract.....	1
Assignment 1: Security Architecture.....	1
1.1 Current Network and Security Architecture	1
1.2 Business Operations Overview.....	3
1.2.1 Customers	3
1.2.2 Suppliers	3
1.2.3 Partners	4
1.2.4 GIAC Enterprises employees located on GIAC Enterprise's network	4
1.2.5 GIAC Enterprises mobile sales force and teleworkers	5
1.3 Technical Architecture	5
1.3.1 IP Addressing Scheme	5
1.3.2 Network Diagram	7
1.3.3 Border Routers	8
1.3.4 Firewalls	9
1.3.5 VPN	13
1.3.6 Other Security Considerations	14
1.3.7 Hardware and Software Costs	14
Assignment 2: Security Policies and Tutorials.....	16
2.1 Cisco Border Router Security Policy	16
2.1.1 Ingress Filter	18
2.1.2 Egress Filter	22
2.1.3 Interface-specific commands	23
2.2 Netfilter Security Policy and Tutorial.....	23
2.2.1 Syntax	24
2.2.2. Primary Firewall Policies and Rules	25
2.3 VPN Policy	34
Assignment 3: Security Audit.....	36
3.1 Audit Plan.....	36
3.2 Test Environment for First Audit	37
3.2.1 Traffic going to the Service Network	37
3.2.2 Traffic from the Service Network to the external or internal network	39
3.3 Auditing the Firewall.....	40
3.3.1 Auditing the External Interface	40
3.3.2 Auditing the Internal Interface	42
3.3.3 Auditing the Service Network Interface	42
3.3.4 Auditing the VPN Interface	42
3.4 Auditing the Rulebase.....	43
3.5 Audit Conclusion.....	50
Assignment 4: Design Under Fire	52
4.1 Internal System Compromise.....	53
Countermeasures	54
4.2 Attack Against the Firewall	54
Countermeasures	56

4.3 DoS Attack	56
Countermeasures	56
References	57
Appendix A: Dell.com Shopping Cart	59
Appendix B: rc.firewall script	63
Appendix C: Chris Brenton's Sample Netfilter Files	70
Appendix D: Netfilter Log Entries for the Firewall Audit	77

© SANS Institute 2003, Author retains full rights.

Abstract

GIAC Enterprises is a small company, with 45 employees, and has been in business for three years. It is an e-business, dealing with the online sale of fortune cookie sayings, so all business transactions are conducted through the Internet.

The IT director has brought security consultants in to propose recommendations for improving their network security and perimeter protection. In addition, a secure VPN solution is required.

With the recent economic downturn, GIAC is very conservative about spending money for new purchases unless there is a justified return on investment. (There is a budget of \$7,500 for hardware and software for this project.) Therefore the design goal is to provide a layered security design, also known as defense-in-depth, that is easy to maintain and manage.

Assignment 1: Security Architecture

1.1 Current Network and Security Architecture

GIAC Enterprises' network is currently protected by an IPChains firewall which was configured by a security consultant when the company was first established.

The two system/network/security administrators are both MCSE's but use Red Hat Linux at home so they are quite comfortable with Microsoft and Linux technologies. All network and server equipment is located in the computer room, which has monitored and restricted access. Servers on the internal network run Windows 2000 server software and use Active Directory Services. All the servers have been hardened as well. In addition, Internet Explorer has been removed from all the servers due to the obvious security risks and ongoing vulnerabilities in the software. There are monthly scheduled maintenance windows to ensure that the latest patches are installed on all devices and servers on the network.

There is a border router connecting their DMZ to the ISP. However, no access control lists (ACL's) were defined. The IPChains firewall only has two interfaces, one to the DMZ and one to the internal network. The web server, secure ftp server, SMTP relay server and Infowave Wireless Business Engine (for mobile workers) are all located within the DMZ. The current placement of these servers exposes them to direct attacks from the Internet.

At the moment, the web server experiences about 6 million customer hits per month. All DNS queries are passed through to the internal Microsoft Dynamic DNS servers.

The SMTP Relay server also has RAV's Antivirus for Sendmail installed to scan and clean mail messages. Because not all email originates from the Internet, McAfee's GroupShield is also installed on the Exchange Server.

All the Windows 2000 servers also has McAfee's NetShield installed and the virus pattern files are updated once a week.

GIAC Enterprises' current security architecture is shown in Diagram 1.1.

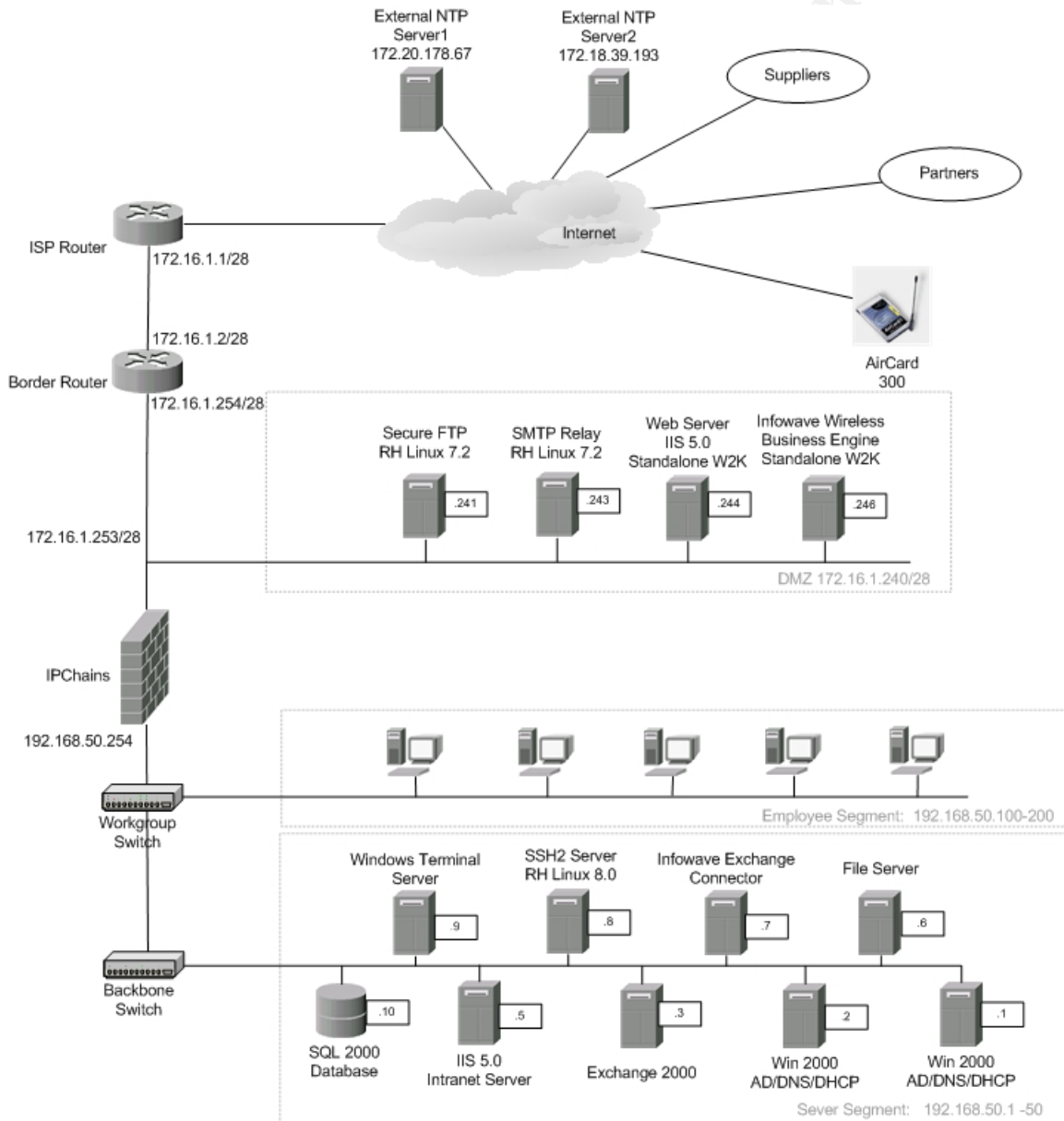


Diagram 1.1: GIAC Enterprises' current security architecture

All desktops are identical in hardware configuration and use Windows 2000 Professional. The standard software configuration consists of:

- Windows 2000 w/ Windows Update installed to download the latest security patches and fixes.
- Norton's Anti-virus w/ automatic downloads of the latest virus pattern files
- Microsoft Office 2000
- Host level firewall software: ZoneAlarm Pro (also provides ad-blocking to minimize unnecessary internet traffic)

All IT staff and mobile employees are provided with standardized laptops with Windows 2000 Professional. In addition, to the standard desktop software configuration, Infowave's Exchange Connector 4.4 client is installed on these laptops.

The Windows Terminal server is currently being used by developers to compile and test code.

Currently, there is no full VPN solution in place – there is only wireless (CDPD) remote access to email through the Infowave solution using Sierra Wireless' Aircard 300.

1.2 Business Operations Overview

1.2.1 Customers

The company's website only sells fortunes in English, however, for all other languages, the customer will be redirected to a partner's website.

Secure access to purchase bulk online fortunes is mandatory. Therefore the actual online purchase, containing customer and credit card information, will be encrypted with SSL.

Since anyone may be a potential customer, all inbound traffic from the Internet will require the following access:

- Email GIAC employees through SMTP on port 25 on the SMTP relay server.
- Access to GIAC Enterprises' website through HTTP on port 80. The purchase process will be performed through SSL on TCP port 443.

1.2.2 Suppliers

These companies supply GIAC Enterprises with their fortune cookie sayings. They are permitted to upload their new batch of fortune cookie sayings in CSV format to a secure ftp server. Each supplier has a unique userid and password

for the secure ftp server and is only permitted to upload into their home directories. All uploads are logged in the server's log files.

When new files are uploaded, an email message is automatically sent to the fortune cookie checkers. Their job is to verify the number of fortune cookie sayings in each batch, ensure that the fortune cookie sayings are within the company's acceptable guidelines policy, check it against the PO and release the PO for payment. Afterwards, the fortune cookie checker uploads the files to the fortune cookies database on the SQL 2000 server.

Suppliers will be required to provide the IP addresses that they'll be connecting from. Once these IP addresses have been authorized on the firewall, they can upload their fortunes to the secure FTP server, in the screened service network, on port 22. Each supplier will have their own directory and will only be able to upload fortunes.

1.2.3 Partners

These are international companies that translate GIAC Enterprises' fortune cookie sayings and then resells them on their own non-English websites. If their customers wish to purchase fortunes in English, they will be redirected from their website to GIAC Enterprises' website.

Since GIAC Enterprises' currently only has a T1 Internet link, it was not feasible to allow partners direct access into their database. In addition, some partners used different types of databases so it was more logical for GIAC Enterprises to automatically export their fortune cookie sayings from their database into a universal format, such as Excel, on a weekly basis. Each partner would then download the exported Excel file from the secure ftp server, using a unique userid and password; and then import the file into their own databases.

Partners are required to provide the IP addresses that they'll be connecting from and these will be authorized on the firewall.

1.2.4 GIAC Enterprises employees located on GIAC Enterprise's network

Internet access for all employees is for business purposes only. They will need Internet access to browse the Web, as well as ftp, therefore they will need outbound access through ports 80, 443, and 21. All internal DNS queries will be resolved by the internal Windows 2000 Dynamic DNS server.

All employees will require the ability to send and respond to emails from customers, partners and suppliers.

1.2.5 GIAC Enterprises mobile sales force and teleworkers

The mobile sales force is the bread and butter of the company, therefore it is crucial that they have access to their email especially when they are on the road, so the company has already equipped them with wireless modem cards like Sierra Wireless' Aircard 300 (CDPD).

All mobile workers are required to use the Infowave Exchange Connector client to access their email on an MS Exchange server, via Infowave's Wireless Business Engine, because it significantly optimizes and compresses the information over a wireless link as well as providing a secure email tunnel to GIAC Enterprises' network. (The software can be used over any Internet link like cable modem or ADSL as well.)

Access to the company's files, some of them are quite large, and custom applications will be required. Due to the limitations of the speed of Internet access while on the road and from home (most employees only have dialup, ADSL or cable modem Internet access), its more feasible to access these files and applications from the Windows Terminal Server, located on the internal network. Therefore, the VPN will restrict access to the Windows Terminal Server only. In addition, no additional software, besides the VPN client, needs to be installed on the laptops. Cisco's Secure VPN client, version 3.6.2, will be installed on all the laptops.

All teleworkers and mobile sales employees require inbound access to:

- Infowave's Wireless Business Engine, on the Screened Service Network, through port 2085
- VPN access to the Cisco VPN 3005 Concentrator (IPSEC, tcp port 50 (ESP) and UDP port 500 (IKE))

To ensure that IT staff can manage and troubleshoot the firewalls remotely, without having to go through the VPN, the firewall will only allow ssh access from defined IP address. This will prevent unauthorized login attempts, into a critical security device from unauthorized hosts.

1.3 Technical Architecture

1.3.1 IP Addressing Scheme

As per the practical instructions, non-routable addresses will be used. Hence, 172.16.1.240/28 will be used for GIAC Enterprises' public addresses and 172.16.0.0 for the rest of the Internet addresses used by suppliers, partners and other external entities on the Internet. The IP addresses are listed in Table 1.3.1.

Device Name	Interface	IP Address
Primary firewall	External (eth0)	172.16.1.253/28
	Screened Services (eth1)	192.168.40.254/24
	Internal (eth2)	192.168.50.254/24
	VPN (eth3)	192.168.60.254/24
Secondary firewall	Employee LAN	192.168.50.253/24
	Secure LAN	192.168.70.253/24
External Network 172.16.1.240/28		
ISP Border Router	External (e0)	172.16.1.2/28
	Internal (e1)	172.16.1.254/28
Cisco VPN 3005 Concentrator	Public/untrusted	172.16.1.252/28
	Private/trusted	192.168.60.253/24
Screened Services Network 192.168.40.0/254		
	Internal IP Address	Public IP Address
SFTP server	192.168.40.201	172.16.1.241
NTP Server	192.168.40.201	172.16.1.253 [#]
External DNS	192.168.40.202	172.16.1.253 [#]
SMTP Relay	192.168.40.203	172.16.1.253 [#]
Reverse Web Proxy	192.168.40.204	172.16.1.253 [#]
Web Proxy	192.168.40.205	n/a
Infowave Wireless Business Engine	192.168.40.206	172.16.1.253 [#]
[#] Port forwarding will be used for these services.		
Internal Network 192.168.50.0/24		
Win 2000/AD/DNS/DHCP server1		192.168.50.1
Win 2000/AD/DNS/DHCP server2		192.168.50.2
Exchange Server		192.168.50.3
Web Server		192.168.50.4
Intranet Server		192.168.50.5
File Server		192.168.50.6
Infowave Exchange Connector		192.168.50.7
SSH2 Management Server		192.168.50.8
Microsoft Windows Terminal Server		192.168.50.9
Secure Network 192.168.70.0/24		
SQL 2000 Database		192.168.70.100
Syslog server		192.168.70.101
Partners		
Emporio Fortuna		172.16.33.39
Chateau de Fortunes		172.16.79.106
Suppliers		
Sayings 'R US		172.16.92.5
Karma Surprises		172.16.47.245
4Tunes		172.16.17.35

Table 1.3.1: IP Addressing Scheme

The IT staff also have fixed IP addresses and since they registered all 5 cards at the same time, they were fortunate enough to obtain consecutive IP addresses (172.16.45.45 to 172.16.45.49).

1.3.2 Network Diagram

The proposed security architecture is depicted in Diagram 1.3.2.

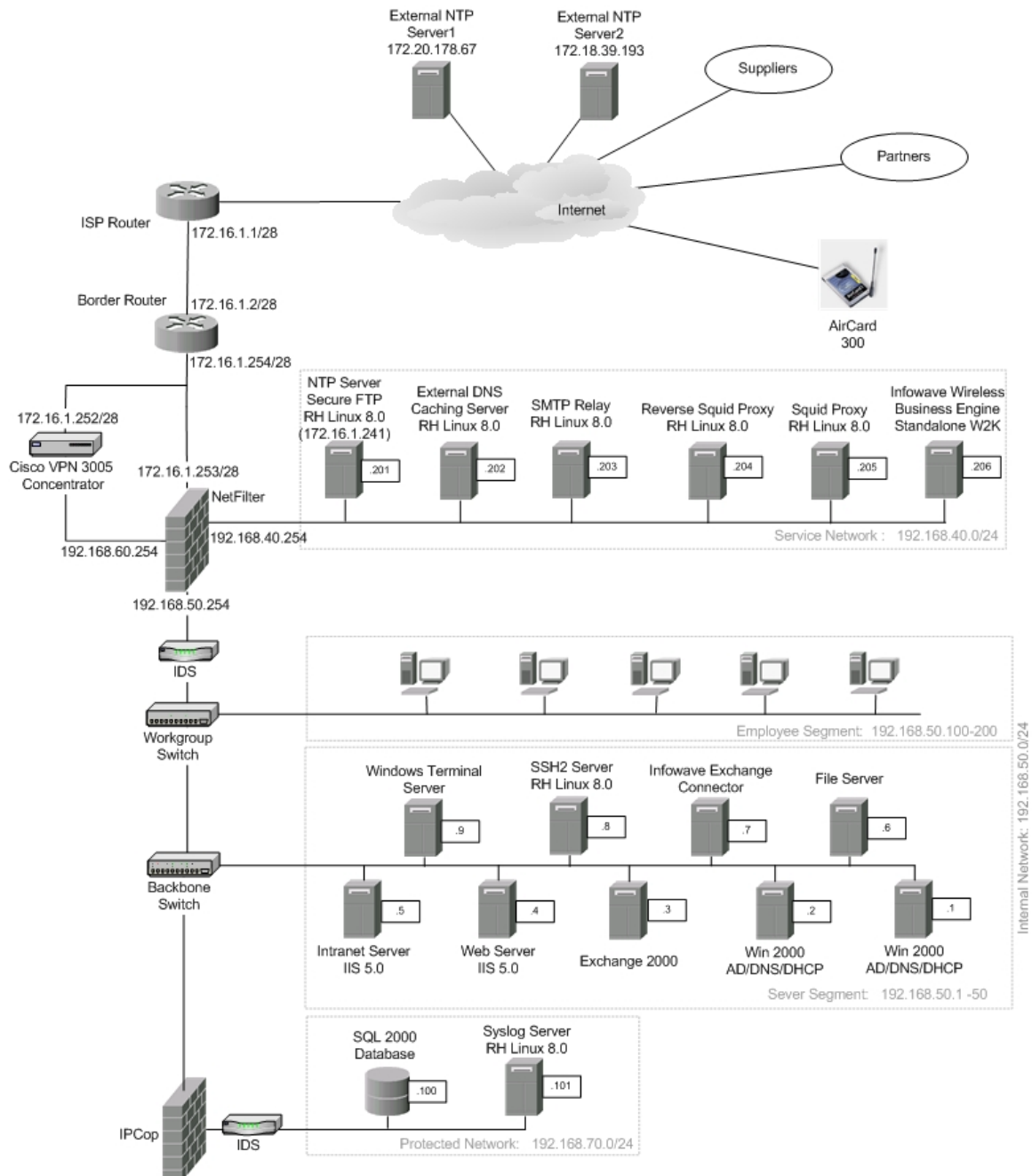


Diagram 1.3.2 Proposed Security Architecture

1.3.3 Border Routers

Since all of GIAC Enterprises' business transactions are conducted through the Internet, it is important to ensure that the border routers and Internet link would not be the primary source of failure because that would translate into lost revenue. Usually, problems with an ISP's internet connectivity is not within the control of IT staff, however hardware/software failure of GIAC devices on their own network are within manageable limits. Recommendations has been included below to improve the high availability of the Internet connection. There are three options available:

1. Purchase a second Cisco 2691 router (about \$5,000) and configure it to use HSRP (Hot Standby Routing Protocol) to provide real-time failover. However, problems with the ISP will be a single source of failure.
2. Provision a second low speed Internet link to act as a backup. A low cost option would be a business DSL link using a Cisco 673 (about \$600) with a monthly charge of about \$80 per month for a 256k SDSL internet link. However, this would involve a lot of coordination with both ISP's as well as BGP configuration on the routers which translates into higher maintenance.
3. Provision a second low speed link (same as option 2) and use Radware's LinkProof to ensure uninterrupted link traffic management. An entry level 2-port LinkProof appliance is about \$8,250, however minimal configuration maintenance is required.

Radware's Linkproof product would provide continuous Internet access by monitoring the defined routers and re-route traffic from an overloaded (or failed) router to the other under-utilized (or functioning) router. The LinkProof device would be defined as the default router for all egress traffic and would also act as the Authoritative DNS for resolution of GIAC's publicly-accessible addresses. In addition, there is a performance gain because the effective network bandwidth to the Internet is the sum of the bandwidths available through each Internet access link, i.e. the links would be used collectively.

However, due to conservative budget constraints, these options and redundancy for all security components will not be implemented until more funds become available.

The existing border router is a Cisco 2691 and its primary purpose is to route traffic between the Internet and GIAC Enterprises' network. This router will be leveraged as a static packet filter as well and will be our first line of defense in our security design. The Cisco routers are running IOS version 12.2(13a).

As mentioned in the SANS Firewalls 101: Perimeter Protection with Firewalls course notes, these border routers will be used for "absolute" filtering, i.e. prevent unwanted traffic from entering or leaving the network under any conditions.

1.3.4 Firewalls

The existing IPChains firewall will be replaced by a new Netfilter firewall. When the Netfilter firewall has been proven to be stable, the IPChains firewall can be upgraded to Red Hat Linux 8.0 and be re-deployed as one of the IDS systems.

Note: Even though it is not depicted in the logical diagram, the firewalls are sandwiched by hubs and these hubs are connected to switches. This allows traffic to be easily sniffed coming in and out of the firewall. Even though the ports on the switches can be mirrored, not all the traffic may be duplicated to the mirrored port when the traffic load is high/heavy.

The hardware for both firewalls are Dell PowerEdge 600SC P4 2.4GHz servers with 512MB RAM, and IDE RAID 1 configuration for the hard drives. The installed OS is Red Hat Linux 8.0, hardened with Bastille.

Primary Firewall

Netfilter has been chosen as the primary firewall due to its extensive and flexible logging capabilities. In addition, the IT staff is already familiar with the IPChains, a precursor to NetFilter.

The primary firewall has three interfaces: one for the external network, one for the service network and one for the internal network. Its main purpose is to protect the servers on the service network as well as the internal network; and restrict vpn access to certain parts of the network.

Secondary Firewall

A different firewall is recommended because if the primary firewall is compromised, the attacker won't be able to compromise the secondary firewall with the same vulnerability. Due to the limited budget, we will be using another open source firewall called IPCop, to provide an additional layer of protection for the database and syslog servers.

The secondary firewall has two interfaces: one for the internal network and the other for the protected network. This firewall will only allow:

- Syslog traffic to the Syslog Server originating from the border router, primary firewall, VPN gateway and IDS systems
- Traffic between the web server and the SQL 2000 database only
- DNS traffic from the DDNS servers
- Windows 2000 Active Directory traffic
- Access to the database from authorized IP addresses on the internal network only.

(Unfortunately, a detailed configuration of the secondary firewall is not covered due to time constraints for completing this practical.)

Service Network

For all hosts on the service network, identical hardware (Dell 600SC Celeron) is used. Red Hat Linux 8.0 was chosen as the operating system because it is cheaper than buying MS Windows 2000 server licenses. Recovery is also simpler as well. Bastille was used to lock down all hosts and all unnecessary services have been disabled. We have chosen to use host-based IDS, Tripwire on the service network since the majority of hosts are Linux-based and there's no added maintenance of a network-based IDS server. (Tripwire for Linux is free.)

No data is stored on any of the servers in this network. The configuration settings for each of these servers is documented and stored on the file server, in a directory that is only accessible by IT staff, and backed up on a nightly basis. All the servers on the service network will be managed through SSH (v2).

External and Internal DNS

Split DNS is implemented so that two versions of the DNS databases are available: one for internal users only and the other for external Internet users only. This minimizes the exposure of information about internal hosts.

The external DNS server is a Red Hat Linux 8.0 installation running BIND version 9.2.1 and is configured as a caching-only DNS server. The external DNS server will only provide name resolution for publicly accessible services. Zone transfers from the external DNS server is limited to the ISP's DNS server only.

The internal DNS uses the Microsoft Windows 2000 Dynamic DNS (DDNS) services and is only accessible from the internal network. No zone transfers are permitted. It is configured to perform recursion for internal queries so if it receives non-authoritative requests, the DNS query will be forwarded to the external DNS server for resolution. Internal users will only have access to these DDNS servers. By default, Windows 2000 DHCP clients can register their resource records with the DDNS servers. This will be disabled and only DHCP servers will be allowed to register resource records with the DDNS servers.

Prior to placing the new DNS in place, the system/network/security administrators must update their domain name records to reflect the new DNS server information. The ISP will also need to be informed as well.

External Mail Server

The external mail server will be upgraded to Red Hat Linux 8.0 and retain its configuration as the SMTP relay for the Exchange Server. It will be hardened and run sendmail version 8.12.5 with all the latest patches installed.

In addition, SpamAssassin 2.43 will be installed to help reduce the amount of spam emails entering GIAC Enterprises' system.

Squid Reverse Proxy

This will act as a gateway for incoming HTTP and SSL requests to GIAC's web server. The advantages of using a reverse proxy are:

1. Provides a layer of protection for the web server.
2. Cache's GIAC's static web content and therefore frees up the web server's CPU cycles to do other tasks like serving up dynamic web content.
3. Allows easy replacement of the backend web server in cases of hardware or software failures on that web server.

When the reverse proxy receives a request, it forwards it to the Web server; and if required, the Web server retrieves information from the SQL 2000 database and returns the information back to the reverse proxy.

Squid Proxy

This will act as a gateway between the browser clients on the internal network and the Internet by sending HTTP, HTTPS and FTP requests on their behalf. Some of the advantages of using a proxy server are:

1. Conserves Internet bandwidth usage by caching visited web sites.
2. Hides the client's actual IP address and uses the squid proxy's IP address instead. (Masquerading does not have to be done at the firewall for client's on GIAC's internal network.)
3. Can be used for URL filtering if the need arises.

NTP/SFTP Server

The current Secure FTP server will be upgraded to Red Hat Linux 8.0 with all the latest patches installed. In addition, this server will also function as the NTP server to ensure that the time on all logging hosts, with the exception of the border routers and firewalls, in the GIAC network is synchronized so that timestamps on log entries are consistent across the network. Since the border routers and firewalls are crucial elements of GIAC's network security, they will be configured manually. After testing with several public time servers for stability

and latency, two different upstream NTP servers (172.16.178.67 and 172.18.39.193) were selected to provide time synchronization for GIAC's NTP server.

Infowave Wireless Business Engine

The Infowave solution is already implemented in the current security architecture but its worth describing since it can be perceived as a "VPN" solution. It consists of three components:

- Infowave Exchange Connector client
- Infowave Wireless Business Engine
- Infowave Exchange Connector Server

Note: There are other Infowave components available but GIAC currently only has the Exchange Connector implementation.

The Wireless Business Engine (WBE) is the core product and every Infowave connector plugs into this component. Essentially, the Wireless Business Engine creates a secure tunnel for the mobile wireless user to the corporate network through the wireless network (CDPD, CDMA, GSM/GPRS, etc). The Infowave solution provides a complete security model and provides for authentication, authorization, encryption and data integrity.

From the Infowave whitepaper, it describes how other components work with the WBE:

All Infowave products that plug into the Wireless Business Engine use Windows NT/2000 Challenge/Response (NTLM) authentication. Users are authenticated to the NT/2000 domain using username, password, and domain to ensure that the person attempting to log in is a recognized system user. To prevent the potential capture of user information, the Wireless Business Engine encrypts all NTLM tokens before they are transferred over the wireless data network.

The Wireless Business Engine is typically a standalone server while the other plug-in products, like the Infowave Exchange Connector server, are members of the domain.

The Wireless Business Engine Technical Overview whitepaper describes Infowave's encryption scheme as "a combination of public key (Certicom's 163-bit Elliptical Curve Cryptography (ECC) algorithm) and symmetric key cryptography (120-bit DESX for symmetric key encryption). The lengthier and slower ECC key is only used for the initial authentication and session key exchange. Thereafter, all data is encrypted with the DESX key."

GIAC is running the most current version of the software and at the time of writing, the latest version is 4.4. The default client connect port has been changed during configuration from port 2082 to port 2085.

Internal Network

There is a scheduled maintenance window every month to ensure that servers and network devices have the latest patches installed.

All servers on the internal network are backed up on a nightly basis.

Infowave Exchange Connector

The Infowave Exchange Connector Server is located on the internal network, behind the firewall, and provides access to the Exchange Server. This server is part of the Windows domain. The Web phone component has been disabled.

Web Server

Since the number of hits to the web server was not excessively high and the firewall is able to handle these hits effortlessly, it was feasible to relocate it to the internal network. This server's exposure to attack is hugely minimized with the reverse Squid proxy and the more protected location of the internal network.

Syslog Server

This server will be used as a central repository for all security logs. It will allow the system/network/security administrators to grep and correlate logfiles on one server, rather than obtaining the information from multiple logfiles on multiple devices. In addition, storing all logfiles on a remote logging server makes it a little harder for a network intruder to cover their tracks.

1.3.5 VPN

A Cisco VPN 3005 Concentrator, running software version 3.6.7, will be used for the VPN gateway since it is very easy to configure and maintain. This device would cost around \$2,800. This VPN device can support up to 100 simultaneous connections and is designed for small- to medium-sized businesses with bandwidth requirements up to full-duplex T1. The Cisco VPN 3005 Concentrator fits very neatly into the current requirements.

As mentioned previously, VPN users will only be allowed to access the Windows Terminal Server on the network. This configuration will allow mobile and teleworkers to access their email as well, however, all their email can not be stored locally on the laptops because Windows Terminal Server does not allow

users to copy data from a Windows Terminal Server session onto the user's local hard drive. (That functionality would require the purchase of the Citrix MetaFrame product.) The Infowave solution allows these workers to have an ost file on their laptops and the Sales Team often access their email offline.

1.3.6 Other Security Considerations

Social engineering is a major unaddressed security concern and it is strongly recommended that the IT staff develop a security briefing to inform all employees what it is and how not to become victims of social engineering. A brief social engineering demo would definitely motivate them to be more security conscious about giving out information ☺. The security briefing should be included as part of employee orientation as well.

1.3.7 Hardware and Software Costs

The proposed cost for hardware, in addition to the current infrastructure already in place, is summarized in Table 1.3.7-1.

Description	Hardware	Price
Primary Firewall	Dell PowerEdge 600SC P4 2.4 GHz w/ 512K Cache, 512MB RAM, 2x 40GB IDE hard drives in an IDE RAID1 configuration; 3Yr Same Day 4Hr response parts + onsite labor	\$1,085
Secondary Firewall	Dell PowerEdge 600SC P4 2.4 GHz w/ 512K Cache, 512MB RAM, 2x 40GB IDE hard drives in an IDE RAID1 configuration; 3Yr Same Day 4Hr response parts + onsite labor	\$1,085
External DNS server	Dell 600SC Celeron 1.7GHz w/ 256K Cache, 256MB DDR 200MHZ DIMM, 40GB IDE Hard Drive	\$449
Squid Reverse Proxy	Dell 600SC Celeron 1.7GHz w/ 256K Cache, 256MB DDR 200MHZ DIMM, 40GB IDE Hard Drive	\$449
Squid Proxy	Dell 600SC Celeron 1.7GHz w/ 256K Cache, 256MB DDR 200MHZ DIMM, 40GB IDE Hard Drive	\$449
Syslog server	Dell 600SC Celeron 1.7GHz w/ 256K Cache, 128MB DDR 200MHZ DIMM, 120GB IDE Hard Drive	\$693
IDS (Snort)	Dell 600SC Celeron 1.7GHz w/ 256K Cache, 256MB DDR 200MHZ DIMM, 40GB IDE Hard Drive	\$449
VPN Gateway	Cisco VPN 3005 Concentrator	\$2,800
Total		\$7,459

For a more detailed configuration of the Dell servers, please refer to Appendix E: Dell.com Shopping Cart.

Table 1.3.7-1: Summarized Hardware Costs

The proposed software (and costs) are summarized in the Table 1.3.7-2.

Software	Price
Red Hat Linux 8.0	Free
Netfilter Software	Free
Squid software	Free
Tripwire for Linux	Free
SpamAssasin	Free
Cisco Secure VPN Client	Included in purchase of VPN Concentrator
Snort	Free
Total	\$0

Table 1.3.7-1: Summarized Software Costs

In total, we're well within budget ☺.

© SANS Institute 2003, Author retains full rights.

Assignment 2: Security Policies and Tutorials

2.1 Cisco Border Router Security Policy

It's assumed that the Cisco router is only configured with the assigned IP address and basic passwords are configured. Let's assign a hostname to the router. We will use a non-descriptive name.

```
hostname shredder
```

Next, we'll add a warning banner to clarify that this router is for authorized use only.

```
banner login /  
INFORMATION IN THIS DEVICE BELONGS TO GIAC ENTERPRISES AND MAY NOT  
BE COPIED (IN WHOLE OR IN PART) IN ANY MANNER WITHOUT EXPRESS  
WRITTEN AUTHORIZATION. This device may be used only for the authorized  
business purposes of GIAC Enterprises. Anyone found using this device or its  
information for any unauthorized purpose or personal use will be subject to  
disciplinary action and/or prosecution./
```

A copy of the router configuration will be stored off-line (in a restricted directory on the company's file server) and regularly backed up. Cisco passwords are usually stored in plain text and can be viewed when the configuration is saved or listed. The passwords can be displayed as an MD5 hash with the following command.

```
service password-encryption
```

We'll also establish an encrypted password that users must enter to access the privileged mode.

```
enable secret <password>
```

Next, we'll turn off unneeded services. These "small servers", also known as minor services, on the Cisco router include TCP, UDP, HTTP, BOOTP, and Finger. Even though these services are disabled by default, we want to be certain that they are turned off.

```
no service tcp-small-servers  
no service udp-small-servers  
no ip bootp server  
no service finger
```

Even though, GIAC's Cisco routers include the patches for HTTP vulnerabilities with the web administration interface, we will adhere to the command line interface. (Administrators are more likely to make configuration errors with the easier to use GUI interface.)

no ip http server

No network management application is being used to administer these routers so SNMP traffic can be disabled.

no snmp

Using CDP, you can view information about all the Cisco devices directly attached to this router. We don't want a potential attacker to gain useful information about the network structure through CDP, particularly if the company is a Cisco shop, so we'll disable CDP.

no cdp run

Identd, also known as the auth protocol, is an insecure and useless protocol (for GIAC Enterprises) that allows any client to initiate a TCP session, the host then responds with "Who's there?" and the client replies with its identity.

no ip identd

Source routing is a technique where the sender can specify the route a packet should take in order to reach a target network. This technique could be used to reach networks with a private address space as well as bypass access lists and firewalls so this option will be disabled.

no ip source-route

Disable the auxillary port, which is commonly used for a modem connection.

line aux 0
transport input none

All versions of Cisco IOS 12.0.5 S and above include an SSHv1 server. Even though there are several vulnerabilities with SSHv1, it is more secure than using telnet to manage the border router. With ssh, all traffic to the border router will be encrypted. Local usernames and authentication will be used.

!--- aaa new-model causes the local username/password on the router
!--- to be used in the absence of other aaa statements.

aaa new-model
username admin-giac password 0 4tun8ly

Next we'll generate an SSH key

crypto key generate rsa

Limit the amount of time the ssh connection remains valid

```
ip ssh time-out 60
```

Limit the number of authentication retries to two.

```
ip ssh authentication retries 2
```

Note: To view the generated key, issue the *show cry key mypubkey rsa* from the command prompt.

Limit the router to ssh connections only, i.e. disable telnet access, and only allow access from specific IP addresses. We're going to limit access and only allow the IT Aircards and SSH2 Management Server to ssh in to this router.

```
line vty 0 4  
transport input ssh  
access-class 1 in  
login
```

```
! Allow access from the SSH2 Management Server  
access-list 1 permit 192.168.50.8  
! Allow access from the IT Staff's Aircard's  
access-list 1 permit 172.16.45.45  
access-list 1 permit 172.16.45.46  
access-list 1 permit 172.16.45.47  
access-list 1 permit 172.16.45.48  
access-list 1 permit 172.16.45.49
```

It's important to log significant events and console messages.

```
Logging 192.168.70.101  
Logging trap debug  
Logging console emergencies
```

2.1.1 Ingress Filter

This filter refers to traffic coming into GIAC's network from the Internet and will be applied to the external interface (e0) with the following commands:

```
interface e0  
ip access-group 101 in
```

Since we will want to screen traffic based on address, port and protocols, we'll be using extended access lists. All Cisco access lists are processed in sequence so the most important access will be selectively placed at the top of the list. In addition, the most accessed traffic patterns are placed near the top to improve throughput.

1. We want to prevent spoofed packets from entering the GIAC network:

```
! Deny source IP's with GIAC's external address space
!  
access-list 101 deny ip 172.16.1.240 0.0.0.15 any log
!  
! Deny source IP's with the loopback address
!  
access-list 101 deny ip 127.0.0.0 0.255.255.255 any log
!  
! Deny source IP's with private addresses
access-list 101 deny ip 192.168.0.0 0.0.255.255 any log
!  
! Note: we have not denied 172.16.0.0/12 because is used to represent the  
public  
! Internet address in this practical  
! space
!  
access-list 101 deny ip 10.0.0.0 0.255.255.255 any log
!  
! Deny source IP's with unallocated IANA addresses  
http://www.iana.org/assignments/ipv4-address-space
!  
access-list 101 deny ip 0.0.0.0 255.255.255.255 log
access-list 101 deny ip 1.0.0.0 255.255.255.255 log
access-list 101 deny ip 2.0.0.0 255.255.255.255 log
access-list 101 deny ip 5.0.0.0 255.255.255.255 log
access-list 101 deny ip 7.0.0.0 255.255.255.255 log
access-list 101 deny ip 10.0.0.0 255.255.255.255 log
access-list 101 deny ip 23.0.0.0 255.255.255.255 log
access-list 101 deny ip 27.0.0.0 255.255.255.255 log
access-list 101 deny ip 31.0.0.0 255.255.255.255 log
access-list 101 deny ip 36.0.0.0 255.255.255.255 log
access-list 101 deny ip 37.0.0.0 255.255.255.255 log
access-list 101 deny ip 39.0.0.0 255.255.255.255 log
access-list 101 deny ip 41.0.0.0 255.255.255.255 log
access-list 101 deny ip 42.0.0.0 255.255.255.255 log
access-list 101 deny ip 58.0.0.0 255.255.255.255 log
access-list 101 deny ip 59.0.0.0 255.255.255.255 log
access-list 101 deny ip 60.0.0.0 255.255.255.255 log
....
access-list 101 deny ip 255.0.0.0 255.255.255.255 log
!  
! Deny packets with multicast addresses
!  
access-list 101 deny ip 224.0.0.0 7.255.255.255 any log
```

Block traffic to GIAC's public broadcast address to prevent some forms of probing as well as being a SMURF amplifier.

```
access-list 101 deny ip any host 172.16.1.255 log
```

2. If there are problems with the GIAC network and IT staff need to fix the problem remotely, they will need to get access to the network before the customers. Allow restricted access for IT Aircards for remote management.

```
! Permit SSH access to the firewall for IP addresses in the range  
! 172.16.45.45 – 172.16.45.49  
access-list 101 permit tcp 172.16.45.45 0.0.0.4 host 172.16.1.253 eq 22 log  
! Permit HTTPS access to VPN Concentrator for IP addresses in the  
! range 172.16.45.45 – 172.16.45.49  
access-list 101 permit tcp 172.16.45.45 0.0.0.4 host 172.16.1.252 eq 22 log
```

3. Allow NTP traffic from external NTP servers

```
! Permit external ntp servers  
!  
access-list 101 permit udp host 172.16.178.67 host 172.16.1.241 eq 123 log  
access-list 101 permit udp host 172.16.39.193 host 172.16.1.241 eq 123 log
```

4. The volume of traffic from customer access would be higher than regular employee traffic so allow incoming access to GIAC's publicly accessible services.

```
! Permit HTTP and SSL access to the web proxy server  
!  
access-list 101 permit tcp any host 172.16.1.253 eq 80  
access-list 101 permit tcp any host 172.16.1.253 eq 443  
!  
! Permit smtp access to the external SMTP server  
!  
access-list 101 permit tcp any host 172.16.1.253 eq 25 log  
!  
! Allow external DNS queries through  
!  
access-list 101 permit udp any host 172.16.1.253 eq 53 log  
!  
! Allow zone transfer to the secondary DNS on ISP  
!  
access-list 101 permit tcp host 172.16.1.200 host 172.16.1.253 eq 53
```

5. Allow supplier and partner access to Secure FTP server.

```
! Permit supplier and partner access to secure FTP server  
!  
access-list 101 permit tcp any host 172.16.1.241 eq 22 log
```

6. Connections initiated from the inside need to be able to pass through the router.

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


7. Allow access for mobile sales force and teleworkers

```
! Allow access to Infowave Wireless Business Engine on port 2085
!  
access-list 101 permit udp any host 172.16.1.253 eq 2085  
!  
! Permit VPN traffic to the VPN 3005 Concentrator  
!  
access-list 101 permit esp any host 172.16.1.252 log  
access-list 101 permit udp any eq 500 host 172.16.1.252 eq 500 log
```

8. Deny access to ports that are most frequently probed
(<http://www.sans.org/y2k/ports.htm>) and log this information:

```
! Drop all telnet traffic  
!  
access-list 101 deny tcp any any eq 23 log  
!  
! Drop all ftp traffic  
!  
access-list 101 deny tcp any any eq 21 log  
!  
! Drop all tftp traffic  
!  
access-list 101 deny udp any any eq 69 log  
!  
! Drop all Netbios traffic  
!  
access-list 101 deny tcp any any range 135 139 log  
access-list 101 deny udp any any range 135 139 log  
access-list 101 deny tcp any any 445 log  
access-list 101 deny udp any any 445 log  
!  
! Drop all other SSH access  
!  
access-list 101 deny tcp any any eq 22 log  
!  
! Prevent PCAnywhere traffic  
!  
access-list 101 deny udp any any eq 22 log  
access-list 101 deny tcp any any eq 5631 log  
access-list 101 deny udp any any eq 5632 log  
!  
! Drop all other SMTP traffic  
!  
access-list 101 deny tcp any any eq 25 log  
!  
! Deny POP traffic  
!  
access-list 101 tcp any any range 109 110 log  
!  
! Deny IMAP traffic
```

```

!
access-list 101 deny tcp any any eq 143 log
!
! Deny SNMP traffic
!
access-list 101 deny tcp any any eq 161 log
!
! Deny all UNIX RPC traffic
!
access-list 101 deny tcp any any eq 111 log
access-list 101 deny udp any any eq 111 log
!
! Deny UNIX rlogin, rexec, cmd, spooler traffic
!
access-list 101 deny tcp any any range 512 515 log
!
! Deny UNIX who traffic
!
access-list 101 deny udp any any 513 log
!
! Deny NFS mount service traffic
!
access-list 101 deny tcp any any eq 635 log
!
! Deny Network File System traffic
!
access-list 101 deny tcp any any eq 2000 log
!
! Drop all nntp traffic
!
access-list 101 deny tcp any any eq 119 log
!
! Deny ICQ traffic
!
access-list 101 deny udp any any eq 4000 log
!
! Deny IRC traffic
!
access-list 101 deny udp any any range 6665 6669 log
access-list 101 deny tcp any any range 6665 6669 log
!

```

9. Deny all other traffic and log:

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

2.1.2 Egress Filter

This filter refers to traffic leaving GIAC's network to the Internet and will be applied to the internal interface (e1) with the following commands:

```
interface e1
ip access-group 102 in
```

To prevent spoofed IP attacks from an internal hacker or compromised system, only packets with a valid source IP address will be permitted from leaving the network therefore a standard access list is adequate.

```
! Allow IP addresses in the range 172.16.1.241 - 172.16.1.254
access-list 102 permit ip 172.16.1.241 0.0.0.13
```

A great way to hunt down the sender of spoofed IP packets from GIAC's network is to ensure that unauthorized packets are logged with the log-input option.

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

2.1.3 Interface-specific commands

We want to be a good Internet neighbour and not be an smurf amplifier.

```
no ip direct-broadcast
```

We'll also limit the amount of information given out by the router based on ICMP error messages.

```
no ip unreachable
```

The time on the border router will be configured manually so the NTP protocol will not be used.

```
ntp disable
```

2.2 Netfilter Security Policy and Tutorial

Netfilter, also referred to as iptables, is exceptionally good at controlling and logging traffic flow. It is included in the Linux kernel. The iptables utility is used to configure the Netfilter security policies so Netfilter is often referred to as iptables as well. Under Netfilter, there are: tables, chains and rules.

1. **Tables:** These provide a certain functionality and there are three default tables: filter, nat and mangle. The filter table never alters packets, it just filters them. The NAT table is used for source NAT, destination NAT, masquerading and transparent proxying. The mangle table is used to modify the packet itself or some of the out-of-band data attached to the packet.

2. **Chains:** There are three built-in/default IP filtering chains: INPUT, OUTPUT and FORWARD. These chains are a checklist of rules. The INPUT chain controls traffic destined for the Netfilter local host. The OUTPUT chain controls traffic leaving the Netfilter local host, i.e. traffic generated on this host. And finally, the FORWARD chain controls packets destined for other hosts. User-defined chains can also be added as well.
3. **Rules:** These define conditions that specify whether packets are accepted, rejected, dropped or logged.

2.2.1 Syntax

The general syntax of the iptables command is as follows:

iptables [-t table] command [match] [-j target]

Firewalling with Netfilter Configuration and Syntax (<http://www.knowplace.org/netfilter/syntax.html>) does an excellent job of describing the syntax and I've quoted it below:

- **A table** (-t *table_name*). If omitted, this will default to the filter table. Other built-in tables are nat and mangle.
- **Command** to perform on a chain to either append (-A) , delete (-D) or replace (-R) or insert (-I) followed by the name of a chain in this table
 - Built-in chains for the filter table are INPUT, FORWARD and OUTPUT
 - Built-in chains for the nat table are PREROUTING, POSTROUTING and OUTPUT
 - Built-in chains for the mangle table are PREROUTING and OUTPUT
- **Match** - Available matches are (this list is incomplete):
 - -p (protocol: TCP, UDP, ICMP, etc - can be numeric - see /etc/protocols)
 - -s (source address)
 - -d (destination address)
 - -i (incoming interface)
 - -o (outgoing interface - note that this specifies the interface to send the packet; this does not match the interface origin of the packet)
 - --fragment (will only attempt to match second and further fragments of fragmented packets since there is no way to tell the source and destination ports, or ICMP type of such a packet)
 - --dport (destination port)
 - --sport (source port)
 - --port (source and destination ports are equal to the specified value)
 - --mark (nfmark value)
 - --tcp-flags (allows TCP flags based matching)
 - --syn (shorthand for --tcp-flags SYN,RST,ACK SYN)
 - --tcp-option (examines the numeric value of the TCP option in a TCP header - will drop packet if the TCP header is incomplete)
 - --state (NEW, ESTABLISHED, RELATED, INVALID)
 - --mac-source (source MAC address)

- `-m unclean` (this actually loads a module to attempt to match unusual or malformed packets - EXPERIMENTAL)
- `--tos` (TOS value of the packet)
- `--ttl` (ttl value of the packet)
- **General rate limit section**
- `--limit` (match rate limiting - value can be give in seconds, minute, hour, or day)
- `--limit-burst` (maximum burst number before the imposing the set rate limit)
- **IP based rate limit section (TCP)** - experimental
- `--iplimit-above` - allows you to restrict the number of parallel TCP connections by IP address or address block
- **Owner matching section**
- `--uid-owner userid` (matches if the packet was created by a process with the given effective numeric user ID)
- `--gid-owner groupid` (matches if the packet was created by a process with the given effective numeric group ID)
- `--pid-owner processid` (matches if the packet was created by a process with the given process ID)
- `--sid-owner sessionid` (matches if the packet was created by a process in the given session group)
- **Port scan detection section** - experimental
- `--psd-weight-threshold` (threshold)
- `--psd-delay-threshold` (delay)
- `--psd-lo-ports-weight` (weight)
- `--psd-hi-ports-weight` (weight)
- **Target** if it matches. Available targets are:
 - `ACCEPT` - accept the packet
 - `DROP` - silently drop the packet
 - `REJECT` - drop the packet and inform the sender
 - `LOG` - log the packet via syslogd and continue traversal
 - `ULOG` - send the packet to an userspace logging process (experimental)
 - `MIRROR` - swap the source/destination IP address and resend the packet (experimental)
 - `QUEUE` - queue the packet to an userspace process - if there is no userspace process, the packet is eventually dropped
 - `RETURN` - return to previous (calling) chain
 - `Name_of_a_custom_chain` - user defined, typically in lowercase but can be in uppercase as well

2.2.2. Primary Firewall Policies and Rules

The full ruleset script will be stored in `/rc.firewall` and will be configured to load within the startup scripts. (The full `rc.firewall` script is included in Appendix B.)

Since packets will go through the checklist of firewall rules sequentially until a match is found, the order is important. We want to allow essential traffic to pass through, like remote management, and then the most high volume traffic, followed by medium volume traffic, then low volume traffic.

Firstly, since the firewall is stateful, we want to allow established sessions through without checking the rule base again. Secondly, we want to make sure that no spoofed packets and banned IP addresses are entering the network.

Thirdly, the firewall needs to be remotely managed by the IT staff. Fourthly, high volume traffic should be near the top of the list and these are DNS packets, incoming HTTP and SSL packets and email (SMTP packets). Fifthly, employees need to access the Internet (outgoing internet email, HTTP, HTTPS and FTP). Lastly, remote employee access is required.

Let's now look at the rc.firewall script.

We'll need to enable IP forwarding so that packets can be forwarded between the various interfaces:

```
# Enable IP forwarding  
echo 1 > /proc/sys/net/ipv4/ip_forward
```

The firewall will be hardened as follows:

```
for interface in /proc/sys/net/ipv4/conf/*; do  
    #Block ICMP redirects for all interfaces  
    echo 0 > $interface/accept_redirects  
    # Block IP Source Routing for all interfaces  
    echo 0 > $interface/accept_source_route  
    # Block IP spoofing for all interfaces  
    echo 1 > $interface/rp_filter  
done  
# Block ICMP broadcasts  
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts  
# Enable tcp SYN cookies protection  
echo 1 > /proc/sys/net/ipv4/tcp_syncookies  
# Ignore ICMP Bogus error messages  
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses  
# Log packets with oddball addresses  
echo 1 > /proc/sys/net/ipv4/conf/all/log_martians
```

Even though the Netfilter startup script does flush all current rules and user defined chains, it's still a good idea to flush them again in the firewall script:

```
# Flush all rules for all chains on restart  
iptables -v -F
```

In addition, we want to ensure that our other tables are flushed as well:

```
# Flush all established connections in the nat and mangle tables  
iptables -t nat -v -F  
iptables -t mangle -v -F
```

Once the chains have been flushed, any user-defined chains can now be deleted:

```
iptables -X
```

When a packet has been checked against the rules in a chain and there are no more rules left, the system will check that chain's policy to determine the fate of the packet. So we want to set the default policies to drop all packets:

```
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
```

We will define several environment variables in the rc.firewall script so that any changes in IP addresses, due to a change in ISP's or addressing scheme, won't need to be redefined for every rule.

```
eth3_IP="192.168.60.254"
eth2_IP="192.168.50.254"
eth1_IP="192.168.40.254"
eth0_IP="172.16.1.253"
ALL_INTERFACES="eth0 eth1 eth2 lo"

INTERNAL_NET="192.168.50.0/24"
SERVICE_NET="192.168.40.0/24"
WAN_NET="172.16.1.240/28"
VPN_NET="192.168.60.0/24"
ALL_INT_NET="192.168.40.0/24 192.168.50.0/24 192.168.60.0/24
192.168.70.0/24"
SPOOFED_ADDR="10.0.0.0/8 192.168.0.0/16 0.0.0.0/32 127.0.0.0/8
224.0.0.0/8 240.254.0.0/16"

ROUTER="172.16.1.254"

EXTERNAL_NTP_SERVERS="172.16.178.67 172.16.39.193"
PARTNER_IPs="172.16.33.39 172.16.79.106"
SUPPLIER_IPs="172.16.92.5 172.16.47.245 172.16.245.35"
IT_AIRCARDS="172.16.45.45 172.16.45.46 172.16.45.47 172.16.45.48
172.16.45.49"
MOBILE_WORKERS="172.16.45.123 172.16.45.134 172.16.45.76"
BADGUYS="172.16.200.201 172.16.205.56 "
ISP_DNS="172.16.1.200"
SFTP="172.16.1.241"
SYSLOGGER="192.168.70.101"

SERVICE_NTP="192.168.40.201"
SERVICE_SFTP="192.168.40.201"
SERVICE_DNS="192.168.40.202"
SERVICE_SMTP="192.168.40.203"
SERVICE_RPROXY="192.168.40.204"
SERVICE_PROXY="192.168.40.205"
WBE="192.168.40.206"

EXCHANGE_SERVER="192.168.50.3"
```

```
WEB_SERVER="192.168.50.4"
INFOWAVE_XCONNECTOR="192.168.50.7"
DDNS1="192.168.50.1"
DDNS2="192.168.50.2"
MNGT_IP="192.168.50.8"
WIN_TSERVER="192.168.50.9"
```

Netfilter is capable of maintaining state, i.e. established connections do not pass through the firewall rulebase but are checked against a state table instead. We'll allow state matches through:

```
iptables -A FORWARD -m state --state ESTABLISH, RELATED -j ACCEPT
iptables -A INPUT -m state --state ESTABLISH, RELATED -j ACCEPT
iptables -A OUTPUT -m state --state ESTABLISH, RELATED -j ACCEPT
```

Allow traffic on the loopback interface so that local services can function properly:

```
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
```

We do not want the world to know our addressing scheme behind the firewall so it's important to translate the service network IP addresses back to the external network IP addresses before being forwarded back to the Internet. For all service network IP addresses, except the SFTP server, they will be translated to the IP address of the firewall's external interface.

```
iptables -t nat -A POSTROUTING -s $SERVICE_NET -o eth0 -j SNAT --to-source $eth0_IP
iptables -t nat -A POSTROUTING -s $SERVICE_SFTP -o eth0 -j SNAT --to-source $SFTP
```

Block and log all spoofed source IP traffic:

```
for x in $SPOOFED_ADDR; do
    iptables -A FORWARD -i eth0 -p tcp -s $x -d 0/0 -j LOG --log-prefix "Address spoofing: "
    iptables -A FORWARD -i eth0 -p tcp -s $x -d 0/0 -j DROP
done
```

Occasionally, there are IP addresses that we wish to block from our network:

```
for x in $BADGUYS; do
    iptables -A FORWARD -I eth0 -s $x -d 0/0 -j LOG --log-level info --log-prefix "BADGUY "
    iptables -A FORWARD -I eth0 -s $x -d 0/0 -j DROP
done
```

Pass syslog traffic, originating from the firewall:

```
iptables -A OUTPUT -o eth2 -m state --state NEW,ESTABLISHED -p udp -o eth2 -d $SYSLOGGER --dport 514 -j ACCEPT
```


Allow syslog traffic from the router to the syslog server:

```
iptables -A FORWARD -i eth0 -o eth2 -m state --state NEW -p udp -s $ROUTER  
-d $SYSLOGGER --dport 514 -j ACCEPT
```

Allow firewall to be managed from the management server:

```
iptables -A INPUT -m state --state NEW -p tcp -s $MNGT_IP -d $eth2_IP --dport  
22 -j ACCEPT
```

Allow firewall to be managed from IT AirCards:

```
for x in $IT_AIRCARDS; do  
iptables -A INPUT -m state --state NEW -p tcp -s $x -d $eth0_IP --dport 22 -j  
LOG --log-level info --log-prefix " IT Remote management "  
iptables -A INPUT -m state --state NEW -p tcp -s $x -d $eth0_IP --dport 22 -j  
ACCEPT  
done
```

Allow SSH management traffic to service network:

```
iptables -A FORWARD -m state --state NEW,ESTABLISHED -p tcp -i eth2 -s  
$MNGT_IP_IP -o eth1 -d $SERVICE_NET --dport 22 -j ACCEPT
```

The unclean module is still in the experimental stages and is meant to detect any suspicious activity. This option will only be used for auditing purposes and not on the production firewall so it will be commented:

```
# iptables -A FORWARD -m unclean -j DROP
```

DNS Traffic:

DNS queries normally use udp/53 so everyone will need incoming access to this port on the DNS server on the DNS. Sometimes DNS queries need to use tcp/53 for resolving large responses but this doesn't happen very often. Zone transfers will require access to tcp/ 53 so this will be restricted to the ISP's DNS server, serving as GIAC's secondary DNS.

Port forward from external interface for udp and tcp port 53 to DNS server on DMZ:

```
iptables -t nat -A PREROUTING -i eth0 -p udp -d $eth0_IP --dport 53 -j DNAT --  
to $SERVICE_DNS:53  
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 53 -j DNAT --to  
$SERVICE_DNS:53
```

Log and accept port forwarded traffic to external DNS server on the service network:

```
iptables -A FORWARD -i eth0 -o eth1 -p udp -d $SERVICE_DNS --dport 53 -j LOG
--log-level info --log-prefix " Ext DNS query : "
iptables -A FORWARD -i eth0 -o eth1 -p udp -d $SERVICE_DNS --dport 53 -j
ACCEPT
```

Log and accept zone transfers between ISP DNS and external DNS:

```
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -s $ISP_DNS -d
$SERVICE_DNS --dport 53 -j LOG --log-prefix " Zone xfer : "
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -s $ISP_DNS -d
$SERVICE_DNS --dport 53 -j ACCEPT
```

Log and accept outgoing DNS traffic from DNS server on the service network:

```
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p udp -s
$SERVICE_DNS -d 0/0 --dport 53 -j LOG --log-prefix " Outgoing DNS traffic: "
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p udp -s
$SERVICE_DNS -d 0/0 --dport 53 -j ACCEPT
```

Log and accept DNS traffic between DDNS servers and Service DNS server:

```
iptables -A FORWARD -i eth2 -o eth1 -p udp -s $DDNS1 -d $SERVICE_DNS --
dport 53 -j LOG --log-prefix "DNS Recursion: "
iptables -A FORWARD -i eth2 -o eth1 -p udp -s $DDNS1 -d $SERVICE_DNS --
dport 53 -j ACCEPT
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p udp -s $DDNS2 -d
$SERVICE_DNS --dport 53 -j LOG --log-prefix "DNS Recursion: "
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p udp -s $DDNS2 -d
$SERVICE_DNS --dport 53 -j ACCEPT
```

Incoming HTTP and SSL traffic:

Port forward tcp/80 and tcp/443 to the reverse Squid Proxy server:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 80 -j DNAT --to
$SERVICE_RPROXY:80
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 443 -j DNAT --
to $SERVICE_RPROXY:443
```

Instead of defining two separate lines to pass port forwarded HTTP and SSL traffic to the reverse Squid Proxy, we will use the `-m multiport` option to specify ports 80 and 443:

```
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -d
$SERVICE_RPROXY -m multiport --dport 80,443 -j LOG --log-prefix " Incoming
HTTP, SSL: "
```

```
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -d
$SERVICE_RPROXY -m multiport --dport 80,443 -j ACCEPT
```

Log and allow HTTP and SSL traffic to pass between the Reverse Web Proxy and the Web Server:

```
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s
$SERVICE_RPROXY -d $WEB_SERVER -m multiport --dport 80,443 -j LOG --log-
prefix " HTTP, HTTPS to Web: "
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s
$SERVICE_RPROXY -d $WEB_SERVER -m multiport --dport 80,443 -j ACCEPT
```

SMTP Traffic:

Port forward tcp/25 from Incoming Internet connection to SMTP relay server:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 25 -j DNAT --to
$SERVICE_SMTP:25
```

Log and accept port forwarded SMTP traffic through to SMTP Relay Server on the service network:

```
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -d
$SERVICE_SMTP --dport 25 -j LOG --log-prefix " Incoming SMTP: "
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -d
$SERVICE_SMTP --dport 25 -j ACCEPT
```

Log and accept SMTP traffic from SMTP Relay server to the Exchange Server:

```
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s
$SERVICE_SMTP -d $EXCHANGE_SERVER --dport 25 -j LOG --log-level info --log-
prefix " SMTP-Xchange: "
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s
$SERVICE_SMTP -d $EXCHANGE_SERVER --dport 25 -j ACCEPT
```

Log and accept SMTP traffic from Exchange Server to SMTP relay

```
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s
$SERVICE_SMTP -d $EXCHANGE_SERVER --dport 25 -j LOG --log-level info --log-
prefix " Xchange-SMTP: "
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s
$SERVICE_SMTP -d $EXCHANGE_SERVER --dport 25 -j ACCEPT
```

Log and accept smtp traffic from SMTP Relay Server to the Internet:

```
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s
$SERVICE_SMTP -d 0/0 --dport 25 -j LOG --log-prefix " SMTP Relay: "
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s
$SERVICE_SMTP -d 0/0 --dport 25 -j ACCEPT
```

Outgoing Employee Squid Proxy traffic:

Log and accept all internal requests to Squid Proxy on Service Network on port 8008:

```
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s  
$INTERNAL_NET -d $SERVICE_PROXY --dport 8008 -j ACCEPT  
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s  
$INTERNAL_NET -d $SERVICE_PROXY --dport 8008 -j LOG --log-level info --log-  
prefix "Int Squid traffic : "
```

Allow Squid Proxy to access everything on ports 80,443 and 21 on the Internet:

```
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s  
$SERVICE_PROXY -d 0/0 -m multiport --dport 80,443,21 -j LOG --log-prefix "  
Squid Proxy: "  
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s  
$SERVICE_PROXY -d 0/0 -m multiport --dport 80,443,21 -j ACCEPT
```

Secure FTP Traffic:

Destination NAT public SFTP address to the IP address of the Secure FTP server on the service network for incoming connections from the Internet:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $SFTP --dport 22 -j DNAT --to  
$SERVICE_SFTP:22
```

Log and allow Partners to access SFTP sever through port 22:

```
for x in $PARTNER_IPs; do  
iptables -A FORWARD -i eth0 -o eth1 -p tcp -s $x -d $SERVICE_SFTP --dport  
22 -j LOG --log-level info --log-prefix " Partner SFTP: "  
iptables -A FORWARD -i eth0 -o eth1 -p tcp -s $x -d $SERVICE_SFTP --dport  
22 -j ACCEPT  
done
```

Log and allow Suppliers to access SFTP sever through port 22:

```
for x in $SUPPLIER_IPs; do  
iptables -A FORWARD -i eth0 -p tcp -s $x -d $SERVICE_SFTP --dport 22 -j  
LOG --log-level info --log-prefix " Supplier SFTP: "  
iptables -A FORWARD -i eth0 -p tcp -s $x/32 -d $SERVICE_SFTP --dport 22 -j  
ACCEPT  
done
```

Allow internal users to download/upload files to Secure FTP server:

```
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s  
$INTERNAL_NET -d $SERVICE_SFTP --dport 22 -j LOG --log-level info --log-prefix  
"Internal SFTP: "
```

```
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s
$INTERNAL_NET -d $SERVICE_SFTP --dport 22 -j ACCEPT
```

NTP Traffic:

Allow NTP server to talk to the upstream external NTP servers:

```
for x in $EXTERNAL_NTP_SERVERS; do
    iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s
    $SERVICE_NTP -d $x --dport 123 -j ACCEPT
done
```

Log and accept all internal NTP queries:

```
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s
$INTERNAL_NET -d $SERVICE_NTP --dport 123 -j LOG --log-level info --log-
prefix " NTP: "
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s
$INTERNAL_NET -d $SERVICE_NTP --dport 123 -j ACCEPT
```

Infowave Traffic:

Port forward tcp/2085 to Infowave Wireless Business Engine (WBE) server:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 2085 -j DNAT --
to $WBE:2085
```

Log and allow authorized Infowave IP addresses to the WBE:

```
for x in $MOBILE_WORKERS; do
    # Log and pass authorized natted Infowave traffic through to DMZ
    iptables -A FORWARD -m state --state NEW -p tcp -s $x -d $WBE --dport
    2085 -j LOG --log-level info --log-prefix "Incoming Infowave: "
    iptables -A FORWARD -m state --state NEW -p tcp -s $x -d $WBE --dport
    2085 -j ACCEPT
done
```

Pass Infowave traffic between WBE and Infowave Exchange Connector:

```
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s $WBE -d
$INFOWAVE_XCONNECTOR --dport 2085 -j LOG --log-level info --log-prefix
"WBE-XConnector: "
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s $WBE -d
$INFOWAVE_XCONNECTOR --dport 2085 -j ACCEPT
```

VPN Traffic:

Log and allow VPN traffic to Windows Terminal Server only:

```
iptables -A FORWARD -i eth3 -o eth2 -m state --state NEW -p tcp -d
$WIN_TSERVER --dport 3389 -j LOG --log-level info --log-prefix "VPN-WTS: "
iptables -A FORWARD -i eth3 -o eth2 -m state --state NEW -p tcp -d
$WIN_TSERVER --dport 3389 -j ACCEPT
```

Logging for other dropped packets:

Catch all rule, all other packets are logged:

```
iptables -A FORWARD -j LOG --log-level info --log-prefix " Default Drop (FWD):"
iptables -A INPUT -j LOG --log-level info --log-prefix " Default Drop (IN):"
iptables -A OUTPUT -j LOG --log-level info --log-prefix " Default Drop (OUT):"
```

Currently, there is a Netfilter NAT vulnerability that may reveal information on how port forwarding is done with unfiltered ICMP messages (http://www.linuxsecurity.com/advisories/redhat_advisory-2057.html). The following workaround is recommended:

```
# Filter out untracked local icmp packets
iptables -A OUTPUT -m state -p icmp --state INVALID -j DROP
```

2.3 VPN Policy

Three tunneling protocols are available on the VPN 3005 Concentrator: PPTP, L2TP and IPSec. We will be using IPSec since it is more secure than the other two protocols.

We will be using the VPN Concentrator's internal authentication server since we have a small user base. So VPN users will have a username and password separate from their Active Directory username and password. All users will be members of the base group.

All the various built-in servers, except SSL, for performing network and system management functions will be disabled.

Split tunneling is disabled by default on the VPN Concentrator and client. Only the VPN Concentrator can enable split tunneling – once it's been enabled on the VPN Concentrator, Mode Configuration is used to push it to and enable it on the IPSec client. For security purposes, split tunneling will not be enabled.

The system policies for the base group are defined in Table 2.3.:

General Parameters	
Access Hours	No restrictions
Simultaneous Logins	2
Minimum Password Length	8
Idle Timeout	30 (minutes)
Maximum Connect Time	400 (minutes)
Filter	None
Tunneling Protocol	IPSec
IPSec Parameters	
IPSec SA	ESP/IKE-3DES-MD5*
IKE Keepalives	Enabled
Tunnel Type	Remote Access
Authentication	Internal
Group Lock	Enabled
Client FW Parameters	
Firewall Setting	Firewall Required
Firewall	Zone Labs ZoneAlarm Pro

Table 2.3: Base Group System Policies

*3DES 168-bit data encryption will be used for the IKE tunnel and IPSec traffic, while ESP/MD5/HMAC-128 authentication is for IPSec traffic and MD4/HMAC-128 authentication for the IKE tunnel.

© SANS Institute 2003, Author retains full rights.

Assignment 3: Security Audit

Now that the firewall policies and rules have been defined, we want to be certain that the firewall is secure and that the firewall policies have been correctly implemented. Based on Lance Spitzner's audit procedures, we need to first define our expectations.

Firstly, we expect the firewall to be secure. Secondly, we expect that the rulebase is permitting legitimate traffic and dropping/ rejecting illegitimate traffic as defined in the technical architecture section.

Lance's audit procedures also recommends digging deeper, i.e. auditing or vulnerability assessing other network devices. However, this falls beyond the scope of this practical and will therefore not be covered.

3.1 Audit Plan

Since, there is an IPChains firewall in place at the moment, we have the luxury of auditing this newly implemented firewall in a test environment. Both system administrators will be involved in the audit as part of the knowledge transfer process. In addition, two non-standard laptops (Windows XP and Linux) are available so additional test hardware is not required. The most time-consuming part of the audit will be the port scans and it is estimated to take 3 days. Verification of the firewall rules will take about 2 days. So in total, 5 working days is recommended. Since the audit will be conducted in a test environment, separate from the production network, no prior approval will be required.

However, when the new Netfilter firewall is in production, a second audit is recommended after the firewall has proven to be stable. Ideally, the second audit should take place during work hours to determine if the firewall is able to handle normal traffic volume in addition to the audit traffic. To mitigate the risk of congesting the network with audit traffic, the network will be closely monitored so that the audit traffic can be slowed down or stopped. In addition, there is a small risk that a real attack may not be detected during this audit period. This risk can be alleviated by not making the audit period public knowledge, closely monitoring the logs and be cognizant that not all suspicious traffic is part of the audit. Since a preliminary audit has been conducted, the second audit will require less time to complete and an estimate of two working day should be sufficient.

Prior to the second audit, we need to inform the CTO of our plans and obtain the necessary approval to proceed. In addition, the ISP should be informed as well so that the audit activity is not inadvertently reported to the authorities.

We'll be using nmap for our security audit. Nmap is a free tool that supports a large number of scanning techniques including ICMP, UDP, TCP connec, TCP

SYN, FIN, etc scans. In addition, other features like OS fingerprinting (-O option), decoy scanning, etc. are available when the appropriate options are specified.

The results of an nmap scan will include a list of interesting ports. The ports will have a status of either open, filtered or unfiltered. An open port means that the target machine is accepting connections on that port. A filtered port means that a filtering device is blocking the port and nmap is unable to determine its status. An unfiltered port means that nmap knows that the port is closed and that there is no filtering device.

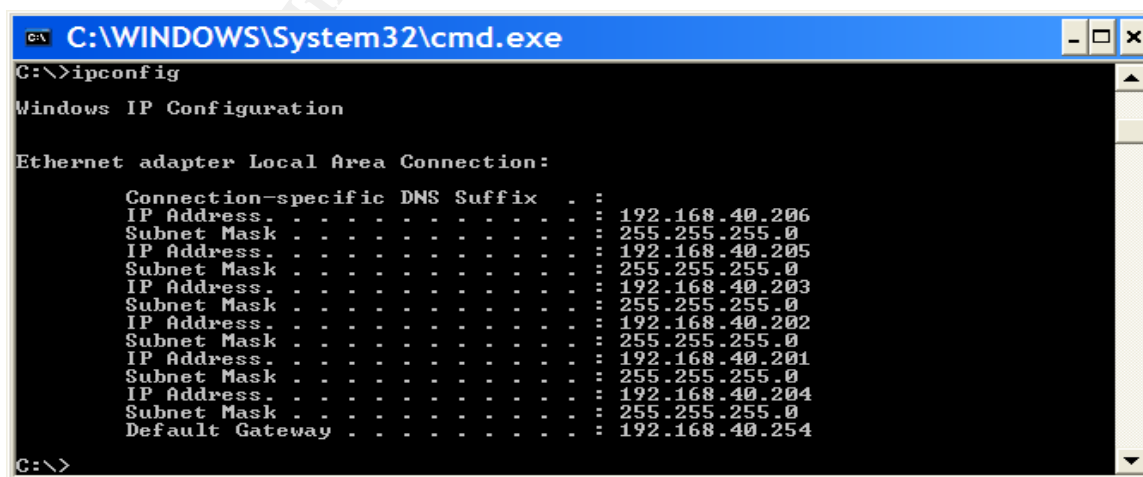
Overall, there are no additional hardware or software costs for the audits. The only additional cost would be the security consultants time for 5 days for the first audit. Since knowledge on how to conduct an audit will be transferred during the first audit, the security consultant's services will not be required for the second audit.

3.2 Test Environment for First Audit

Since the first audit will be conducted in a test environment, we may not have all services available for testing, e.g. web server, SMTP relay, Exchange Server, etc. So, we will be using the netcat tool to "emulate" these services. NetCat is described as the tcp/ip "Swiss Army Knife" that can be run as a listener (-l option) on any specified port (-p option). It's an excellent tool for auditing firewall policies.

3.2.1 Traffic going to the Service Network

To test traffic destined to the servers on the Service Network, the Windows XP will have multiple addresses bound to its network interface card with multiple netcat listeners to emulate all the services on the service network. Running ipconfig from a command prompt shows the multiple addresses bound to a single nic:



```
C:\WINDOWS\System32\cmd.exe
C:\>ipconfig

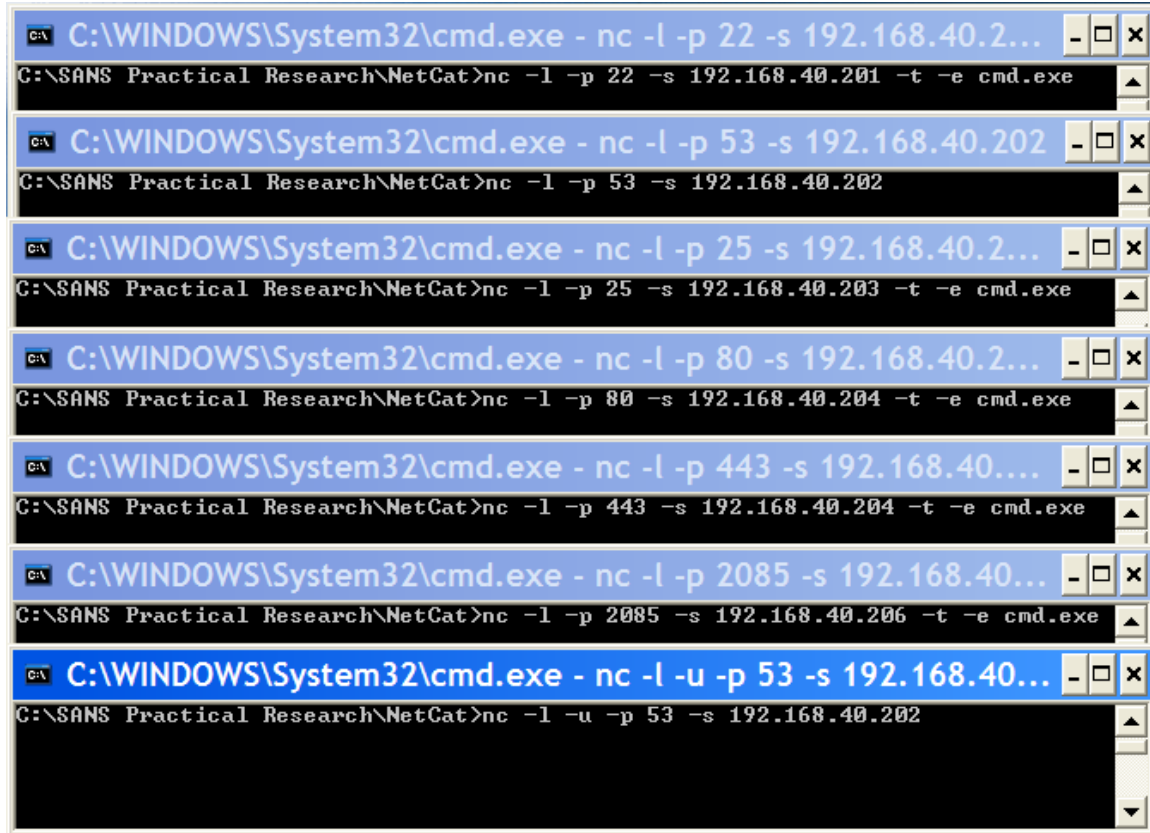
Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    IP Address. . . . .               : 192.168.40.206
    Subnet Mask . . . . .             : 255.255.255.0
    IP Address. . . . .               : 192.168.40.205
    Subnet Mask . . . . .             : 255.255.255.0
    IP Address. . . . .               : 192.168.40.203
    Subnet Mask . . . . .             : 255.255.255.0
    IP Address. . . . .               : 192.168.40.202
    Subnet Mask . . . . .             : 255.255.255.0
    IP Address. . . . .               : 192.168.40.201
    Subnet Mask . . . . .             : 255.255.255.0
    IP Address. . . . .               : 192.168.40.204
    Subnet Mask . . . . .             : 255.255.255.0
    Default Gateway . . . . .         : 192.168.40.254

C:\>
```

For each netcat listener, we'll run it from separate command prompt windows:



The netcat utility uses the following options:

- `-l` to run in listen mode
- `-u` use udp (If `-u` is not specified, tcp is used.)
- `-p <port number>` to specify the port number
- `-s <IP address>` listen for the specified service on the specified IP address
- `-t -e cmd.exe` open up a command prompt when a client successfully connects

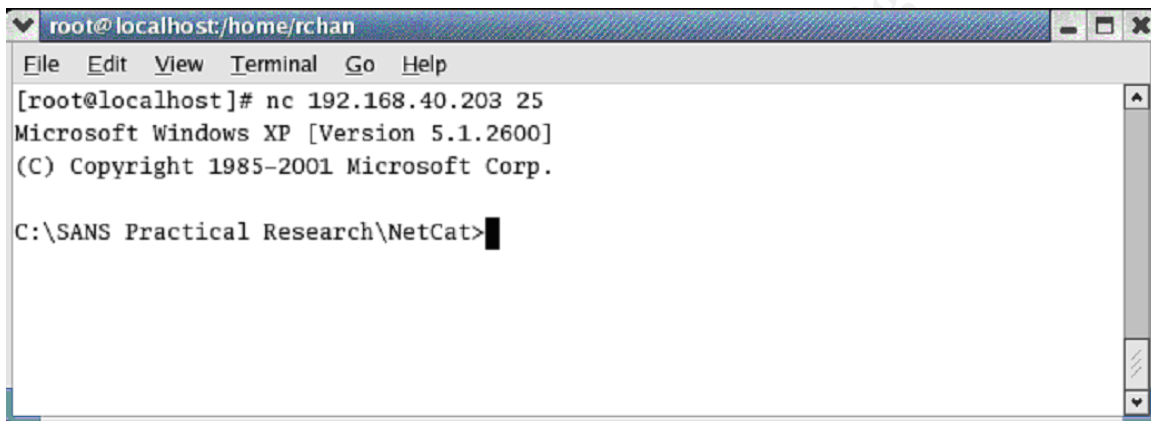
We will have eight netcat listeners for:

- TCP/22 emulating the ssh service on the Secure FTP server
- TCP/53 and UDP/53 emulating the external DNS server
- TCP/25 emulating the SMTP Relay server
- TCP/80 and TCP/443 emulating the HTTP and SSL service on the reverse Squid Proxy server
- TCP/8008 emulating the Squid Proxy server for internal users
- TCP/2085 emulating the Infowave Wireless Business Engine server

The Linux laptop will travel between the external and internal network, depending on the test, and either run netcat or nmap, depending on the rule we're verifying. On the client laptop, to use netcat to connect to the listener, we would use the following command:

```
nc <IP address of listener> <port number>
```

When the netcat client successfully connects to the netcat listener, we have a Windows command prompt on our Linux laptop and the screen capture is shown below:



The screenshot shows a terminal window titled 'root@localhost:/home/rchan'. The terminal output is as follows:

```
[root@localhost]# nc 192.168.40.203 25
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\SANS Practical Research\NetCat>
```

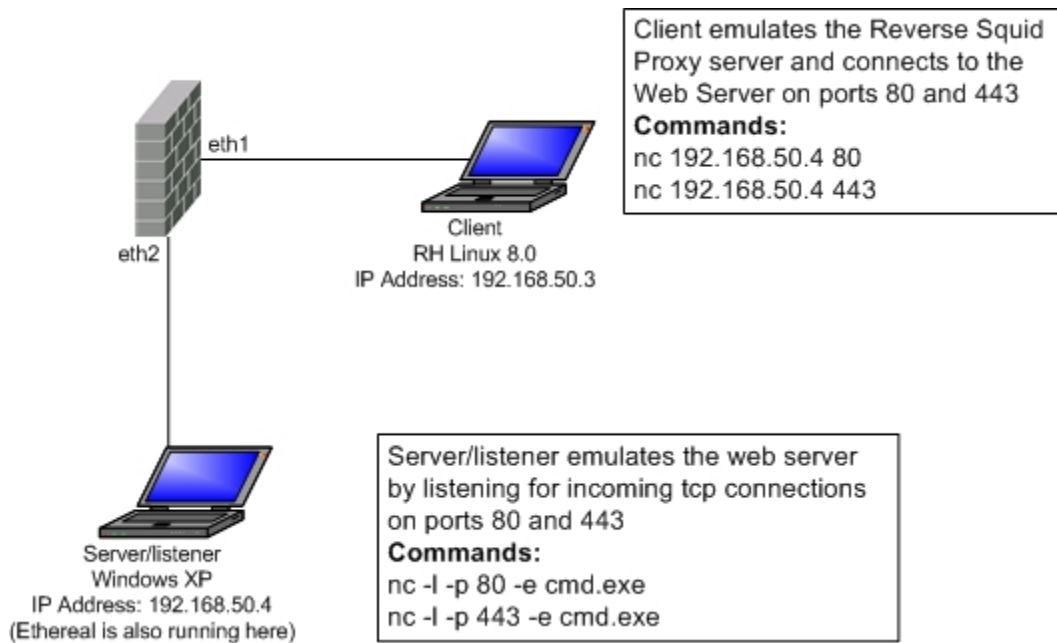
Netcat, as a client, has a `-u` option to specify a connection to a udp port instead, however, with this option netcat works in conjunction with ICMP packets. Since ICMP is being blocked by the firewall, we cannot use the netcat client for rules specifying specific udp ports. Instead, we'll use nmap with the `-sU` and `-P0` option, to check if the port is open.

For rules requiring specific source IP addresses on the client side, we will use nmap with the `-S` option, to spoof the source IP address.

3.2.2 Traffic from the Service Network to the external or internal network

To test traffic leaving the Service Network and going to the external or internal networks, the Linux laptop will be connected to the Service Network interface of the firewall and act as the client.

The Windows XP laptop will travel between the external and internal networks to emulate the services required by the servers on the Service Network. For example, to test outgoing HTTP and SSL traffic from the Reverse Squid Proxy server to the Web server, we would set up the laptops as depicted in the diagram on the following page:



3.3 Auditing the Firewall

We will port scan the firewall (icmp, tcp and udp), from a non-privileged IP address, to check for open ports on all segments to determine if the firewall itself is secure. Only the first audit of the primary firewall is covered in this section.

3.3.1 Auditing the External Interface

Firstly, we'll start the firewall audit from the external interface, from a non-privileged IP address, with the most basic scan, a simple ping to the firewall's public IP address of 172.16.1.253. We didn't get any response back. The following firewall log entry indicates that the ping (ICMP echo request, PROTO=ICMP TYPE=8) packet was dropped by the INPUT chain's default drop policy.

```
Feb  9 19:19:33 Obfuscate kernel:  Default Drop (IN): IN=eth0 OUT=
MAC=00:09:5b:09:fa:33:00:60:97:8b:ea:d2:08:00 SRC=172.16.1.251
DST=172.16.1.253 LEN=84 TOS=0x00 PREC=0x00 TTL=64 ID=0 DF PROTO=ICMP
TYPE=8 CODE=0 ID=26397 SEQ=16640
```

We will now do a tcp scan and the nmap command will be using the `-sT` option, to specify a TCP Connect scan, `-p 1-65535`, to specify all ports, `-P0` to disable ping before scanning. A screen capture of the command and results is included below.

```
root@localhost:~  
File Edit View Terminal Go Help  
[root@localhost root]# nmap -sT -p 1-65535 -PO 172.16.1.253  
  
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )  
Interesting ports on (172.16.1.253):  
(The 65532 ports scanned but not shown below are in state: filtered)  
Port      State      Service  
25/tcp    open       smtp  
80/tcp    open       http  
443/tcp   open       https  
  
Nmap run completed -- 1 IP address (1 host up) scanned in 5545 seconds
```

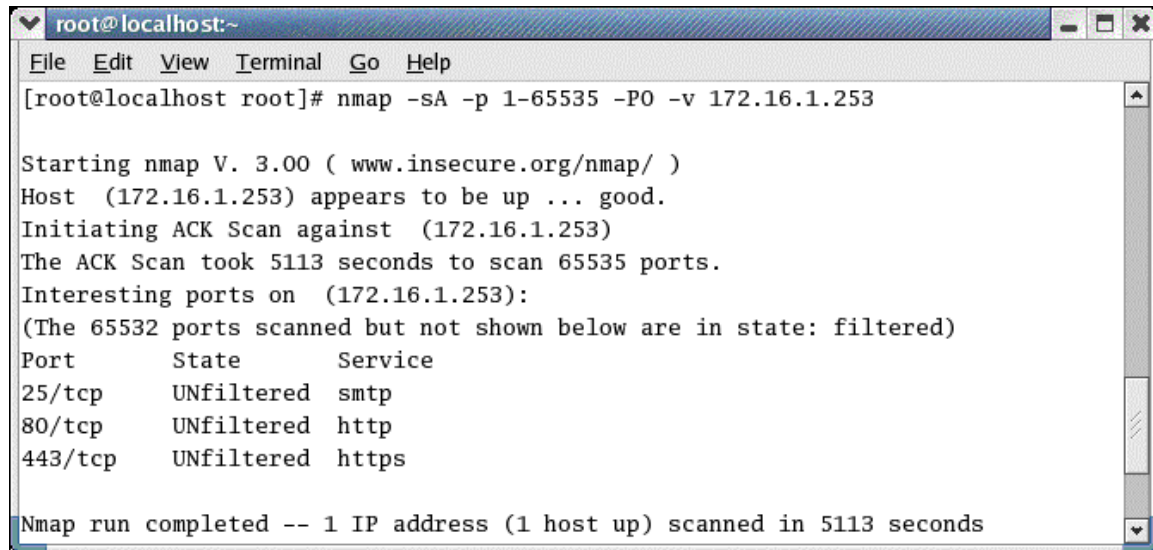
The tcp scan shows that ports 25, 80 and 443 are open, i.e. accepting connections on those ports, but the rest of the ports are filtered, i.e. blocked by the firewall. This is expected since we are port forwarding on those ports for all incoming connections from the Internet.

However, when we run the scan (*`nmap -S 172.16.1.45.45 -e eth0 -sT -p 1-65535 -PO 172.16.1.253`*) using the IP address of an IT AirCard, authorized partner or reseller, we see that tcp port 22 is open. Port scanning from a mobile worker's IP address (*`nmap -S 172.16.45.134 -e eth0 -sT -p 1-65535 -PO 172.16.1.253`*), shows that port 2085 is open. Port scanning from the ISP's DNS server address (*`nmap -S 172.16.1.200 -e eth0 -sT -p 1-65535 -PO 172.16.1.253`*), shows that tcp port 53 is open. This is expected since we are restricting access to those ports from authorized IP addresses only.

We now try a UDP scan, -sU option. From the screen capture on the following page, we can see that the scan took a significantly longer time to complete – 40.5 hours, almost two days! This is due to the fact that every udp port had to time out before the scan tried another udp port. As expected only udp port 53 is open, while the rest of the udp ports are being dropped by the firewall.

```
root@localhost:~  
File Edit View Terminal Go Help  
[root@localhost root]# nmap -sU -PO -p 1-65535 172.16.1.253  
  
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )  
Interesting ports on (172.16.1.253):  
(The 65534 ports scanned but not shown below are in state: filtered)  
Port      State      Service  
53/udp    open       domain  
  
Nmap run completed -- 1 IP address (1 host up) scanned in 145712 seconds  
[root@localhost root]#
```

Lastly, we try an ACK scan, -sA option, on the firewall:



```
root@localhost:~  
File Edit View Terminal Go Help  
[root@localhost root]# nmap -sA -p 1-65535 -P0 -v 172.16.1.253  
  
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )  
Host (172.16.1.253) appears to be up ... good.  
Initiating ACK Scan against (172.16.1.253)  
The ACK Scan took 5113 seconds to scan 65535 ports.  
Interesting ports on (172.16.1.253):  
(The 65532 ports scanned but not shown below are in state: filtered)  
Port      State      Service  
25/tcp    UNfiltered  smtp  
80/tcp    UNfiltered  http  
443/tcp   UNfiltered  https  
  
Nmap run completed -- 1 IP address (1 host up) scanned in 5113 seconds
```

The ACK scan results show that ports 25/tcp, 80/tcp and 443/tcp are unfiltered, i.e. nmap knows that those ports are closed and there is no firewall/filter preventing nmap from determining this fact. This verifies that the firewall is not accepting non-established ACK connections, i.e. the firewall is stateful.

3.3.2 Auditing the Internal Interface

The IP address of the firewall's internal interface was pinged and there was no response back. The firewall logs indicated that the ICMP echo-request packet was dropped and this was verified with Ethereal.

A tcp scan was done on the firewall's internal interface and all ports were filtered. The udp and ack port scans also showed that all ports were filtered.

3.3.3 Auditing the Service Network Interface

ICMP, udp and tcp port scanning on the service network interface, from a non-privileged IP address, showed that all ports were filtered.

3.3.4 Auditing the VPN Interface

ICMP, udp and tcp port scanning on the VPN interface, from a non-privileged IP address, showed that all ports were filtered as well.

3.4 Auditing the Rulebase

We will ensure that the rulebase is correctly implemented by scanning each segment on the firewall to validate that:

- Legitimate packets are being accepted by the firewall
- All dropped and rejected packets, in the rulebase, are not forwarded

We'll be using a combination of tools (netcat, nmap and Ethereal) to verify the rules. The firewall management rule does not require a listener to be setup so we can test this rule directly with just one laptop, acting as the client.

Firewall Rule Management (tcp 22 on the Netfilter firewall)			
Test Case	Client command (Linux)	Firewall Log	Status
SSH access from an IT AirCard (from eth0)	<code>nmap -sS -S 172.16.45.45 -e eth0 -p 22 -v -P0 172.16.1.253</code>	Logged and tagged as "IT Remote Management"	Accepted
SSH access from non-authorized external IP address	<code>nmap -sS -S 172.16.44.45 -e eth0 -p 22 -v -P0 172.16.1.253</code>	Logged and tagged as "Default Drop (IN)"	Dropped
SSH access from the management server	<code>nmap -sS -S 192.168.50.8 -e eth0 -p 22 -v -P0 192.168.50.254</code>	Logged and tagged as "Internal FW Management"	Accepted
SSH access from another internal IP address	<code>nmap -sS -S 192.168.50.100 -e eth0 -p 22 -v -P0 192.168.50.254</code>	Logged and tagged as "Default Drop (IN)"	Dropped

The nmap options used are:

-sS	Use a TCP SYN scan
-S 172.16.45.45	Specify the source address as 172.16.45.45
-e eth0	Send and receive packets on eth0
-p 22	Specify port 22 only
-P0	Do not ping target before scanning

The log entry for the first test case shows that the packet was accepted by the IT Remote Management rule:

Feb 5 13:21:16 Obfuscate kernel: IT Remote Management IN=eth0 OUT=MAC=00:09:5b:09:fa:33:00:60:97:8b:ea:d2:08:00 SRC=172.16.45.45 DST=172.16.1.253 LEN=40 TOS=0x00 PREC=0x00 TTL=49 ID=34999 PROTO=TCP SPT=57884 DPT=22 WINDOW=2048 RES=0x00 SYN URGP=0

The log entry for the second test case shows that the packet was dropped by the default drop rule:

Feb 5 13:26:57 Obfuscate kernel: Default Drop (IN): IN=eth0 OUT=MAC=00:09:5b:09:fa:33:00:60:97:8b:ea:d2:08:00 SRC=172.16.44.45 DST=172.16.1.253 LEN=40 TOS=0x00 PREC=0x00 TTL=47 ID=30220 PROTO=TCP SPT=60826 DPT=22 WINDOW=4096 RES=0x00 SYN URGP=0

The firewall management rule is accepting and dropping packets as expected.

The firewall rules related to employee, customer, partner and supplier rules will be verified first to ensure that all traffic essential to the business operations of GIAC Enterprises is correctly implemented. All the corresponding log entries verifying acceptance or denial of packets are included in Appendix D: Netfilter Log Entries for the Firewall Policy Audit. Verification of these firewall rules are listed in the following table:

Service Network Management (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to internal network)					
Test Case	Client's IP address	Netcat client command	Firewall Log	Status	Ethereal
TCP/22 from authorized management server	192.168.50.8	nc 192.168.40.202 22	Logged and tagged as "Service Net Management"	Accepted	Packets passed and captured
TCP/22 from non-authorized IP	192.168.50.8	nmap -sS -S 192.168.50.100 -e eth0 -P0 -p 22 192.168.40.202	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Internal Employee Access (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to internal network)					
Test Case	Client's IP address	Netcat client command	Firewall Log	Status	Ethereal
Access to Squid Proxy on DMZ on port 8008	192.168.50.188	nc 192.168.40.205 8008	Logged and tagged as "Int Squid Traffic"	Accepted	Packets passed and captured
Access to Squid Proxy on DMZ on port 80	192.168.50.188	nc 192.168.40.205 80	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Access to Squid Proxy on DMZ on port 21	192.168.50.188	nc 192.168.40.205 21	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Access to any other DMZ address on port 8008	192.168.50.188	nc 192.168.40.206 8008	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Access to SMTP relay on DMZ on port 25	192.168.50.188	nc 192.168.40.203 25	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Access to external IP address on port 80	192.168.50.188	nc 172.16.1.250 80	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Squid Proxy traffic to Internet (Server, 172.16.1.251, w/ netcat listener for port 80 and 443, connected to external network; Client connected to service network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal

Squid Proxy access to tcp/80	192.168.40.205	nc 172.16.1.251 80	Logged and tagged with "Squid Proxy"	Accepted	Packets passed and captured
Squid Proxy access to tcp/443	192.168.40.205	nc 172.16.1.251 443	Logged and tagged with "Squid Proxy"	Accepted	Packets passed and captured
Squid Proxy access to tcp/21	192.168.40.205	nc 172.16.1.251 21	Logged and tagged with "Squid Proxy"	Accepted	Packets passed and captured
Squid Proxy access to tcp/23	192.168.40.205	nc 172.16.1.251 23	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Incoming DNS query (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to external network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
External DNS Query (udp/53)	172.16.1.251	nmap -sU -P0 -p 53 172.16.1.253	Logged and tagged with "Ext DNS Query"	Accepted	Packets passed and captured
Incoming Zone Transfer (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to external network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Zone transfer from authorized IP	172.16.1.251	nmap -S 172.16.1.200 -e eth0 -sS -P0 -p 53 172.16.1.253	Logged and tagged with "Zone Xfer: "	Accepted	Packets passed and captured
Zone transfer from non-authorized IP	172.16.1.251	nc 172.16.1.253 53	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
DNS Recursion (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to internal network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
From DDNS1	192.168.50.1	nmap -sU -P0 -p 53 192.168.40.202	Logged and tagged with "DNS Recursion: "	Accepted	Packets passed and captured
From DDNS2	192.168.50.1	nmap -sU -S 192.168.50.2 -e eth0 -P0 -p 53 192.168.40.202	Logged and tagged with "DNS Recursion: "	Accepted	Packets passed and captured
From non-authorized IP	192.168.50.1	nmap -sU -S 192.168.50.2 -e eth0 -P0 -p 53 192.168.40.202	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
DNS Traffic from external DNS server (Server, 172.16.1.251, w/ netcat listener for port 80 and 443, connected to external network; Client connected to service network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal

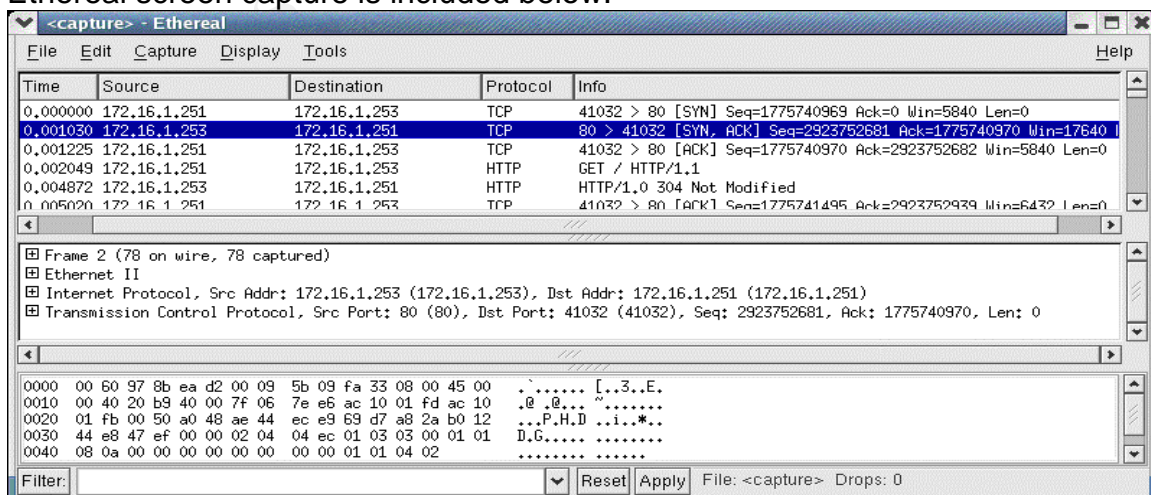
UDP/53 from DNS server on service network to Internet	192.168.40.202	nmap -sU -P0 -p 53 172.16.1.251	Logged and tagged with "Outgoing DNS traffic: "	Accepted	Packets passed and captured
UDP/53 from other IP address to Internet	192.168.40.202	nmap -sU -S 192.168.40.205 -e eth0 -P0 -p 53 172.16.1.251	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Incoming HTTP and HTTPS traffic (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to external network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Incoming tcp/80 to Inverse Squid Proxy	172.16.1.251	nc 172.16.1.253 80	Logged and tagged with "Incoming HTTP,SSL: "	Accepted	Packets passed and captured
Incoming tcp/443 to Inverse Squid Proxy	172.16.1.251	nc 172.16.1.253 443	Logged and tagged with "Incoming HTTP,SSL: "	Accepted	Packets passed and captured
Incoming tcp/21 to Inverse Squid Proxy	172.16.1.251	nc 172.16.1.253 21	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Inverse Squid Proxy traffic to Web Server (Server, 192.168.50.4, w/ netcat listener for port 80 and 443, connected to internal network; Client connected to service network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Tcp/80 from Inverse Proxy to Web Server	192.168.40.204	nc 192.168.50.4 80	Logged and tagged with "HTTP, HTTPS to Web: "	Accepted	Packets passed and captured
Tcp/443 from Inverse Proxy to Web Server	192.168.40.204	nc 192.168.50.4 443	Logged and tagged with "HTTP, HTTPS to Web: "	Accepted	Packets passed and captured
Incoming SMTP traffic (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to external network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Incoming smtp to tcp/25	172.16.1.251	nc 172.16.1.253 25	Logged and tagged with "Incoming SMTP: "	Accepted	Packets passed and captured
Incoming smtp to tcp/10056	172.16.1.251	nc 172.16.1.253 10056	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
SMTP Relay traffic to Internet (Server, 172.16.1.251, w/ netcat listener for port 25, connected to external network; Client connected to service network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
TCP/25 from SMTP Relay	192.168.40.203	nc 172.16.1.251 25	Logged and tagged with	Accepted	Packets passed

server			"SMTP Relay: "		and captured
TCP/25 from other IP address	192.168.40.203	nmap -sS -S 192.168.40.100 -e eth0 -P0 -p 25 172.16.1.251	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
SMTP Relay to Exchange Server (Server, 192.168.50.3, w/ netcat listener for port 25, connected to internal network; Client connected to service network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Tcp/25 from SMTP relay to Exchange Server	192.168.40.203	nc 192.168.40.203 25	Logged and tagged with "SMTP-Xchange: "	Accepted	Packets passed and captured
Incoming Secure FTP traffic (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to external network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
SFTP from Partner IP	172.16.1.251	nmap -sS -S 172.16.33.39 -e eth0 -P0 -p 22 172.16.1.253	Logged and tagged with "Partner SFTP: "	Accepted	Packets passed and captured
SFTP from Supplier IP	172.16.1.251	nmap -sS -S 172.16.245.35 -e eth0 -P0 -p 22 172.16.1.253	Logged and tagged with "Partner SFTP: "	Accepted	Packets passed and captured
SFTP from non-privileged IP	172.16.1.251	nmap -sS -S 172.16.100.100 -e eth0 -P0 -p 22 172.16.1.253	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Secure FTP from internal network (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to internal network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Tcp/22 to Secure FTP server	192.168.50.188	nc 192.168.40.201 22	Logged and tagged with "Internal SFTP: "	Accepted	Packets passed and captured
NTP Queries from Internal networks (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to internal network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Tcp/123 to NTP server	192.168.50.188	nc 192.168.40.201 123	Logged and tagged with "NTP: "	Accepted	Packets passed and captured
Incoming Infowave traffic (Server, w/ multiple addresses and netcat listeners, connected to Service Network; Client connected to external network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Tcp/2085 from authorized IP in \$MOBILE_WORKERS	172.16.1.251	nmap -sS -S 172.16.45.123 -e eth0 -P0 -p 2085 172.16.1.253	Logged and tagged with "Incoming Infowave: "	Accepted	Packets passed and captured

Tcp/2085 from non-authorized IP	172.16.1.251	nmap -sS -S 172.16.100.100 -e eth0 -P0 -p 2085 172.16.1.253	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through
Infowave WBE to Exchange Connector Traffic (Server, 192.168.50.7, w/ netcat listener for port 25, connected to internal network; Client connected to service network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Tcp/2085 from WBE to Infowave Exchange Connector	192.168.40.206	nc 192.168.50.7 2085	Logged and tagged with "Infowave-Xconnector: "	Accepted	Packets passed and captured
Incoming VPN traffic (Server, 192.168.50.9, w/ netcat listener for port 3389, connected to internal network; Client connected to VPN network)					
Test Case	Client's IP address	Client command	Firewall Log	Status	Ethereal
Connect to the Windows Terminal Server using tcp/3389	192.168.60.60	nc 192.168.50.9 3389	Logged and tagged with "VPN-WTS: "	Accepted	Packets passed and captured
Connect to the Intranet Server on tcp/80	192.168.60.60	nc 192.168.50.5 80	Logged and tagged with "Default Drop (FWD)"	Dropped	No packets went through

Now that all the business operations traffic has been verified, we need to verify that the IP addresses on the service network is hidden and properly translated to the firewall's external address, or in the case of the secure ftp server, to its public secure ftp server address.

We'll use a freeware application called TinyWeb, developed by Maxim Masiutin, to accept and respond to http requests. TinyWeb is a web server with an extremely small footprint – only 53K. TinyWeb will be running on the Windows XP laptop (192.168.50.204), connected to the service network. The Linux laptop, 172.16.1.251, connected to the external network, will run Ethereal and connect to the TinyWeb application, http://172.16.1.253, through its web browser. The Ethereal screen capture is included below:



From the Ethereal screen capture, the first packet is a SYN request to tcp port 80 from the Linux laptop to the TinyWeb application. The second highlighted packet is an SYN ACK from the TinyWeb application to the Linux laptop. The details of the second packet are displayed in the middle window. You'll notice that the source address (Src Addr: 172.16.1.253) is the external IP address of the firewall, and not the IP address of the TinyWeb application. This verifies that the IP addresses on the service network are being correctly translated to the external IP address of the firewall. Similarly, a test can be run to verify that the Secure FTP server is correctly translated as well.

We'll now verify the rest of our rules: dropping spoofed packets and dropping traffic from banned IP addresses. For these tests, we'll leave the TinyWeb application running and try to connect from spoofed and banned addresses using nmap and the -S option to specify the source address.

For the spoofed addresses, on the client, we use the following command:

```
nmap -S 10.10.10.10 -e eth0 -p 80 172.16.1.253
```

The results show that port 80 is blocked for the spoofed address:



```
root@localhost:~  
File Edit View Terminal Go Help  
[root@localhost root]# nmap -S 10.10.10.10 -e eth0 -p 80 -PO 172.16.1.253  
  
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )  
Interesting ports on (172.16.1.253):  
Port      State      Service  
80/tcp    filtered  http  
  
Nmap run completed -- 1 IP address (1 host up) scanned in 37 seconds  
[root@localhost root]#
```

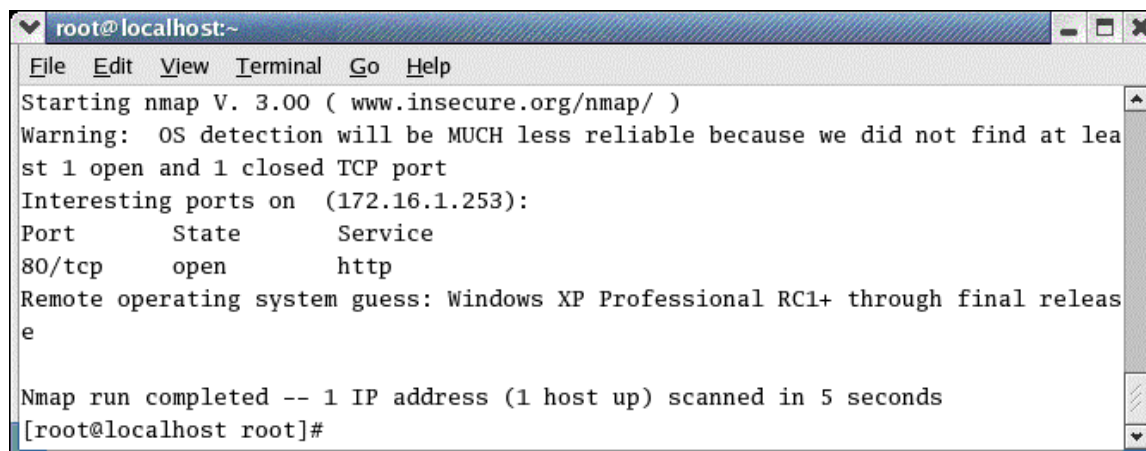
The packet is logged, in the firewall log, as being dropped:

```
Feb  9 19:09:45 Obfuscate kernel: Address spoofing: IN=eth0 OUT=eth1  
SRC=10.10.10.10 DST=192.168.40.204 LEN=40 TOS=0x00 PREC=0x00 TTL=54  
ID=1797 PROTO=TCP SPT=38416 DPT=80 WINDOW=4096 RES=0x00 SYN URGP=0
```

Note that the destination is specified as 192.168.40.204. This is due to the fact that the packet has already been port forwarded and the IP address 172.16.1.253:80 has already been translated to 192.168.40.204:80 prior to it being routed/forwarded onto the service network interface.

Similar tests were performed for other spoofed and banned addresses with the same results.

Since ports 80, 443 and 25 are open to all incoming Internet connections, except spoofed or banned source addresses, it would make OS fingerprinting of the servers on the Service Network quite easy as shown in the screen capture on the following page:



```
root@localhost:~
File Edit View Terminal Go Help
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Warning: OS detection will be MUCH less reliable because we did not find at least 1 open and 1 closed TCP port
Interesting ports on (172.16.1.253):
Port      State      Service
80/tcp    open       http
Remote operating system guess: Windows XP Professional RC1+ through final release
Nmap run completed -- 1 IP address (1 host up) scanned in 5 seconds
[root@localhost root]#
```

Lastly, we ran some non-standard packets (FIN, XMAS and NULL scan) through the firewall to see if this would be detected by the rule to drop and log suspicious activity. The firewall detected it as an unclean packet and issued the following message on the console:

```
ipt_unclean: TCP flags bad: 1
ipt_unclean: TCP flags bad: 14
```

In addition, the default drop policy dropped all the packets from the FIN, XMAS and NULL scans. These packets were also logged.

3.5 Audit Conclusion

The icmp, tcp and udp scans from a non-privileged IP address, against the firewall itself, on all interfaces, showed that only ports tcp/25, tcp/80, tcp/443 and udp/53 is open for everyone, except spoofed or banned source IP addresses. While ports tcp/22, tcp/53 and tcp/2085 are open for authorized source IP addresses only. Each firewall rule was tested and proven that legitimate traffic was accepted and illegitimate traffic was blocked.

One of the scans highlighted the fact that the systems on the service network can be easily fingerprinted because we are using port forwarding for most of the

publicly accessible services. So we need to be very vigilant about applying the latest patches to these systems.

The performance and efficiency of the firewall could also be increased by creating user-defined chains and mapping traffic to these smaller chains instead. In addition, these user-defined chains would be easier to maintain.

Overall, the firewall is correctly implementing the security policies defined in Assignment 2.

© SANS Institute 2003, Author retains full rights.

Assignment 4: Design Under Fire

Alfred Ho's, http://www.giac.org/practical/Alfred_Ho_GCFW.doc, practical assignment will be used to demonstrate that security is an ongoing effort and that the design structure needs to be reviewed and updated on a regular basis.

The network design is using Cisco PIX version 6.2.6 with SSH enabled. Access to ssh is only permitted from the internal 10.0.0.0 network. I've chosen to exploit the SSH Malformed Packet Vulnerability

http://www.cisco.com/en/US/products/sw/iosswrel/ps1835/products_security_advisory09186a008011c3b4.shtml since it is the most recent vulnerability and the PIX is most likely not patched yet. We'll first compromise an internal system in order to launch an attack on the firewall itself.

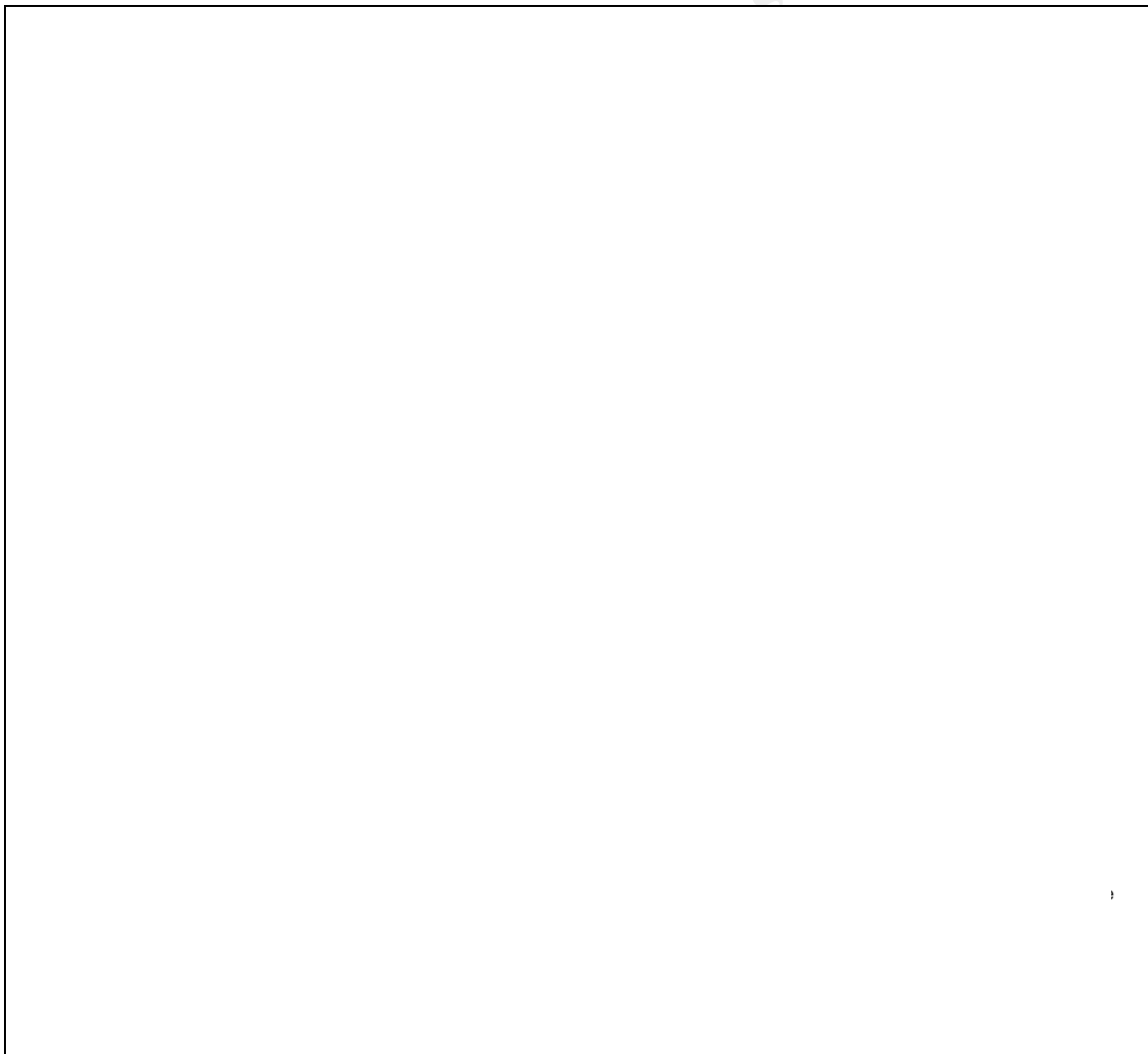


Diagram 4: Alfred Ho's Security Architecture for GIAC Enterprises

4.1 Internal System Compromise

Most companies rely on securing their networks by designing a layered security network architecture and physically securing the routers, switches, servers, etc., yet social engineering is the biggest unaddressed security hole. Social engineering exploits the natural human tendency to trust and help colleagues; and hackers can use this weakness to gain useful information.

So we'll begin our exercise with information gathering through this technique ☺. We'll call GIAC Enterprises and ask the receptionist for the name(s) and number(s) of the system administrator(s) that we could contact regarding a technical information survey. We could get information like John Adams is the system administrator and he can be reached at 555-7635 or jadams@giac.com.

The Sales team usually has to travel to customer sites and would most likely have VPN access into GIAC Enterprises' network. We'll visit the company's web page and click on "Contact Us" for an email address, then we'll request information through a free email account like Yahoo or Hotmail:

I was looking for a cookie fortune vendor and found your website through a Google search. I'd like to get more information about volume purchasing for different locations across North America and would like to get the names, numbers and email addresses of sales people that I could contact directly.

We will very likely get back an email with one or two sales people from GIAC Enterprises, for example:

You can contact one of the following sales reps:
Rob Smith 555-6723 rsmith@giac.org
Mark Simpson 555-6738 msimpson@giac.org

There is also a lot of information that we can gather from a company's website as well, like the name of the CEO, CFO and COO. Let's say that GIAC Enterprises' CEO is David Anders. We'll setup a free email account on Yahoo called David_Anders@yahoo.com.

Now, we'll call Rob asking for his help with a VPN problem on the CEO's computer and the conversation could progress as follows:

Me: Hi Rob, my name is Tom and I'm working with John from the IT department. John is tied up with a critical server upgrade at the moment and there seems to be a problem with David Anders computer, especially his VPN connection. I was wondering if you could help us out?

Rob: Sure, what do you need me to do?

Me: How do you start up your VPN software?

Rob: I click on Start, Run, Cisco Systems VPN Client Software, VPN Dialer.

(We've just learnt that they are using a Cisco VPN Concentrator for VPN access.)

Me: Please start up the VPN Dialer and click on the yellow icon on the tile bar and select "About VPN Client". What version does it display?

Rob: It says version 3.6.

Me: Ok, it looks like David's software is the same version. I need to get your password to see if I can log in from David's computer. Is your network login rsmith?

Rob: Yes and my password is \$money\$.

Me: Ok, I'm logging in as you. ...Mmmm. It doesn't look like it's working either. We'll need to try something else. Please go to your c:\program files\Cisco Systems\Profiles directory.

Rob: Ok, I'm in the specified directory.

Me: You'll need to email all the .pcf files in that directory to David's Yahoo account – he's also having problems with accessing his work email. His email address is David.Anders@yahoo.com.

Rob: No problem. Here are the files.

Me: I've received the files and will need to tweak them. I'll call you again if there's still a problem. Thanks Rob, you've been very helpful and I know that David appreciates your help too.

We've just been able to get Rob's password and connection profile for VPN access!

Countermeasures

- Provide training for all employees on how to prevent social engineering attacks as well as basic network security (including how to select good passwords, not leaving passwords on a sticky, not sharing passwords, etc).

4.2 Attack Against the Firewall

Now that we've successfully established a VPN session into the network, we can get a pretty good idea of how the network is designed. We can figure out the internal IP address of the firewall by doing a ping out to the Internet to see what the last IP address is before icmp echo-request packets are dropped since the firewall would be the only device filtering all icmp packets. When the internal IP address of the firewall has been determined, we can fingerprint the firewall by using nmap with the -O.

The firewall is configured to allow all SSH sessions from the internal network. If we didn't have knowledge of how the PIX was configured, this can be easily determined with a port scan for port 22 on the firewall with nmap:

```
C:\ nmap-3.00>nmap -sT -p22 -PO -v -O 10.2.4.21
```

```
Starting nmap V. 3.00 ( www.insecure.org/nmap )
```

```
Host (10.2.4.21) appears to be up ... good.
```

```
Initiating Connect() Scan against (10.2.4.21)
```

```
Adding open port 22/tcp
```

```
The Connect() Scan took 0 seconds to scan 1 ports.
```

```
Interesting ports on (10.2.4.21):
```

```
Port      State      Service
```

```
22/tcp    open       ssh
```

```
Remote operating system guess: Cisco PIX 515 or 525 running 6.2 (1)
```

The nmap results indicates that port 22 is available for connection and that the firewall may be a Cisco PIX. The above port scan would most likely not be logged because most firewalls are setup to protect internal resources from incoming attacks from the Internet.

After checking Cisco's security advisories, we find that there is an SSH Malformed packet vulnerability that can cause the Cisco PIX to reload. After a few searches on Google, we find that there are already tools to generate malformed ssh packets like SSHredder suite.

Below is a quote of Cert's description (CA-2002-36) of Rapid7's SSHredder suite:

[Rapid7](#) has developed a suite (SSHredder) of test cases that examine the connection initialization, key exchange, and negotiation phase (KEX, KEXINIT) of the SSH transport layer protocol. The suite tests the way an SSH transport layer implementation handles invalid or incorrect packet and string lengths, padding and padding length, malformed strings, and invalid algorithms.

The test suite has demonstrated a number of vulnerabilities in different vendors' SSH products. These vulnerabilities include buffer overflows, and they occur before any user authentication takes place.

SSHredder is a collection of PDU files that can be delivered through netcat to exploit the SSH vulnerability. For example, to use the 0000001.pdu file with netcat with IP address, 10.2.4.21, on port 22, as the target, you would type in:

```
nc 10.2.4.21 22 < 0000001.pdu
```

From Cisco's security advisory, several of these test cases cause the PIX firewall to reload. Since, we're not sure which one will result in the reload, we'll write a script with an infinite loop to go through all the test cases. A sample perl script, assuming that the PDU files are located in /tmp/sshredder, is included below:

```
#!/usr/bin/perl

use File::Listing;
@dir_listing = parse_dir(`ls -l /tmp/sshredder-files`);
$x = 1;
while ( $x > 0)
{
    foreach $file_ref (@dir_listing) {
        ($name, $type, $size, $mtime, $mode) = @$file_ref;
        nc -v 10.2.4.21 22 < $name;
    } ## foreach
} ## while
```

In effect, we have created a DoS attack on the PIX by relentlessly pushing these packets onto the PIX and causing it to reload repeatedly.

Countermeasures

- Ensure that the latest security patches are installed on a timely basis.
- Only allow management access to security devices from secure or trusted workstations on the internal network.

4.3 DoS Attack

We have 50 compromised cable modem/DSL systems at our disposal. I will be using the same technique that Alfred Ho used for his DoS attack – the TFN2K (Tribal Flood Network 2000) available from Packet storm (<http://packetstormsecurity.com/distributed/tfn2k.tgz>). Even though Alfred has recommended countermeasures such as rate-limiting the ISP's router and limiting the number of concurrent sessions to the Web server, they do not seem to be implemented in his network security design so I believe that this attack will succeed.

The 50 compromised cable modem/DSL systems will have the TFN2K agent installed and will be listening for instructions from the master TFN2K server. The IP addresses of the TFN2K agents will be stored in a file called slaves.txt. So, in order for the master server to instruct all the TFN2K agents to launch a SYN flood attack (-c 5 option) against the target GIAC web server 142.32.1.35, (using the -i option) on target port 80 (using the -p option):

```
./tfn -f slaves.txt -c 5 -i 142.32.1.35 -p 80
```

We can check if our attack is successful by trying to connect to the web server to see if there is a response back.

Countermeasures

In addition to Alfred Ho's recommendations of negotiating with the ISP to provide rate-limiting on their routers and limiting the number of concurrent sessions to the web server, we could also:

- Install Psionic Technologies' PortSentry product on the external web server. PortSentry protects the system from targeted system attacks, as well as reconnaissance probes and auto-scanners, by blocking the host in real-time when a predefined user threshold has been reached.
- Install a web proxy server to intercept all HTTP and SSL traffic before it reaches the web server.
- Install a load balancer, like Cisco's LocalDirector, in front of a web server farm and limit the number of concurrent sessions to each web server.

References

"Track 2 – Firewalls, Perimeter Protection, and VPNs" Coursebooks from the SANS Institute.

Assignment 1 References

- *"A Reverse Proxy is a Proxy By Any Other Name"* by Art Stricek, January 10, 2002 http://www.sans.org/rr/web/reverse_proxy.php
- *"Introducing LinkProof"* whitepaper <http://www.radware.com/library/whtpaper/linkproof.pdf>
- *"Infowave Wireless Business Engine – Technical Overview"* whitepaper http://www.infowave.com/pages/Technical_Overview_WBEv4_rev031401.pdf

Assignment 2 References

- *"Cisco IOS 12.2 Command Reference"*: <http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122mindx/crfindx.htm>
- *"Configuring Secure Shell on Cisco IOS Routers"*: http://www.cisco.com/en/US/tech/tk583/tk209/technologies_tech_note09186a00800949e2.shtml#before
- *"SYCTL Options"*: <http://www.tldp.org/HOWTO/Security-Quickstart-HOWTO/appendix.html#SYCTL>
- *"Locking Down the Firewall During the Boot Process"*: <http://www.islandsoft.net/veerapen.html>
- *"Linux 2.4 Packet Filtering HowTo"*: <http://www.netfilter.org/documentation/HOWTO/packet-filtering-HOWTO.txt>
- *"Netfilter and IPTables (2.4 Kernels)"*: <http://www.tldp.org/LDP/nag2/x-087-2-firewall.future.html>
- *"Firewalling with Netfilter Configuration and Syntax"*: <http://www.knowplace.org/netfilter/syntax.html>
- *"Peter Vestergaard's GCFW Practical"*: http://www.giac.org/practical/Peter_Vestergaard_GCFW.zip
- Sample Netfilter script from Chris Brenton: Appendix B
- *"VPN 3000 Series Concentrator Reference Volume I: Configuration"*: http://www.cisco.com/univercd/cc/td/doc/product/vpn/vpn3000/3_6/config/cfg.pdf

Assignment 3 References

- *"Testing a router or firewall"*: <http://rootprompt.org/article.php3?article=2317>
- *"Auditing your Firewall Setup by Lance Spitzner"*: <http://www.spitzner.net/audit.html>

- “TinyWeb” by Maxim Masiutin: <http://www.ritlabs.com/tinyweb/index.html>

Assignment 4 References

- “Alfred Ho’s GCFW Practical”:
http://www.giac.org/practical/Alfred_Ho_GCFW.doc
- “SSH Malformed Packet Vulnerability”:
http://www.cisco.com/en/US/products/sw/iosswrel/ps1835/products_security_advisory09186a008011c3b4.shtml
- “Social Engineering Fundamentals, Part I: Hacker Tactics” by Sarah Granger
<http://online.securityfocus.com/infocus/1527>
- Cert Advisory CA-2002-36: <http://www.cert.org/advisories/CA-2002-36.html>

© SANS Institute 2003, Author retains full rights.

SMALL BUSINESS

[Computers](#)
[Software & Peripherals](#)
[Service & Support](#)

Buy Online or Call
1-800-295-3355

Purchase Assistance

[Common Questions](#)
[Security & Privacy](#)
[Limited Warranty](#)

MY CART

Have you also considered?

Low monthly payments with Small Business Financing

As low as \$109/mo. (46 payments)² No payment for 60 days to qualified customers. [Apply Today](#)

Shopping Cart

Detailed View

Save cart

Print cart

Email cart

Cart items to buy now	Qty.	Unit Price	Total	
PowerEdge 600SC PowerEdge 600SC, Intel Celeron, 1.7GHz w/256K Cache, includes floppy	<div>3</div> <div>Update</div>	\$499.00	\$1,497.00	<div>Move item to buy later</div> <div>Remove item</div>
<div>System Details Reconfigure</div>				
PowerEdge 600SC PowerEdge 600SC, Intel Pentium 4, 1.8GHz w/512K Cache	<div>2</div> <div>Update</div>	\$1,085.00	\$2,170.00	<div>Move item to buy later</div> <div>Remove item</div>
<div>System Details Reconfigure</div>				
PowerEdge 600SC PowerEdge 600SC, Intel Celeron, 1.7GHz w/256K Cache, includes floppy	<div>1</div> <div>Update</div>	\$693.00	\$693.00	<div>Move item to buy later</div> <div>Remove item</div>
<div>System Details Reconfigure</div>				

Coupon Entry

Enter coupon number

OR

Select a coupon from your account

[Apply coupon](#)

Discounts and Coupons	Total	Sub-total	\$4,360.00
Save \$50 on PowerEdge 600SC through the Small Business division.	- \$300.00	Discounts	- \$300.00
Details		Shipping ¹	\$234.00
		Shipping Discount ¹	- \$234.00
		Tax ¹	Unavailable
		Total Price ¹	Unavailable

Shipping Discounts and Coupons

Total

Small Business customers receive free ground shipping on select Systems and Software and Peripherals items.

- \$234.00

[Details](#)

As low as \$109/mo. (46 payments)²

No payment for 60 days - Feature available to qualified customers

[Financing Details](#)

Description

PowerEdge 600SC

PowerEdge 600SC, Intel Celeron, 1.7GHz w/128K Cache, includes floppy

Qty: 3

Price: \$1,497.00

Date:

Wednesday,
February 05, 2003
1:15:10 AM CDT

Catalog Number:

4 04

PowerEdge 600SC:	PowerEdge 600SC, Intel Celeron, 1.7GHz w/128K Cache, includes floppy	617CSC	[221-0773]
Memory:	256MB DDR, 200MHz, for the Price of 128MB(1x128) (mail-in rebate not available)	256DYM	[460-5915]
Keyboard:	Standard Windows Keyboard, Gray	S	[310-1676]
Monitor:	No Monitor Option	N	[320-0058]
First Hard Drive:	40GB 7.2K RPM IDE Hard Drive	40GB	[340-2243]
Operating System:	No Operating System, Other	NOOS/O	[420-4106]
Mouse:	Logitech System Mouse, Gray	L	[310-3776]
1st Network Adapter:	Onboard NIC	OB	[460-6604]
CD ROM/DVD ROM:	48X IDE Internal CD ROM Drive	CD48X	[313-2602]
Documentation:	Electronic Documentation, PowerEdge 600SC	EDOCS	[310-0438]
Hard Drive Configurations:	Motherboard IDE, Ascending Order	MIDE	[340-6930]
Hardware Support Services:	1Yr Parts + Onsite Labor (Next Business Day) + 1Yr Technical Support	W111YOS	[900-9054] [950-3040]
Installation Support Services:	No Installation	NOINSTL	[900-9997]

© SANS I

PowerEdge 600SC

PowerEdge 600SC, Intel Pentium 4, 1.8GHz w/512K Cache

Qty: 2

Price: \$2,170.00

Date:		Wednesday, February 05, 2003 1:15:10 AM CDT	
Catalog Number:		4 04	
PowerEdge 600SC:	PowerEdge 600SC, Intel Pentium 4, 1.8GHz w/512K Cache	6018SC	[221-0774]
Memory:	512MB DDR,200MHz,for the Price of 256MB(2x128) (mail-in rebate not available)	512DYM	[460-5916]
Keyboard:	Standard Windows Keyboard, Gray	S	[310-1676]
Monitor:	No Monitor Option	N	[320-0058]
First Hard Drive:	40GB 7.2K RPM IDE Hard Drive	40GB	[340-2243]
Hard Drive Controller:	IDE RAID Controller, ATA100 / 4 Channel	IDE4CH	[340-5918]
Operating System:	No Operating System, Other	NOOS/O	[420-4106]
Mouse:	Logitech System Mouse, Gray	L	[310-3776]
1st Network Adapter:	Onboard NIC	OB	[460-6604]
CD ROM/DVD ROM:	48X IDE Internal CD ROM Drive	CD48X	[313-2602]
Documentation:	Electronic Documentation, PowerEdge 600SC	EDOCS	[310-0438]
Second Hard Drive:	40GB 7.2K RPM IDE Additional Hard Drive	40GBADD	[340-2246]
Hard Drive Configurations:	C5, Add-In Raid Card,RAID 1 with a 2 Hard Drive Configuration	IAR12HD	[340-6935]
Hardware Support Services:	1Yr Parts + Onsite Labor (Next Business Day) + 1Yr Technical Support	W111YOS	[900-9054] [950-3040]
Installation Support Services:	No Installation	NOINSTL	[900-9997]

© SANS

PowerEdge 600SC

PowerEdge 600SC, Intel Celeron, 1.7GHz w/128K Cache, includes floppy

Qty: **1**Price: **\$693.00**

Date: Wednesday,
February 05, 2003
1:15:10 AM CDT

Catalog Number:		4 04	
PowerEdge 600SC:	PowerEdge 600SC, Intel Celeron, 1.7GHz w/128K Cache, includes floppy	617CSC	[221-0773]
Memory:	256MB DDR,200MHz,for the Price of 128MB(1x128) (mail-in rebate not available	256DYM	[460-5915]
Keyboard:	Standard Windows Keyboard, Gray	S	[310-1676]
Monitor:	No Monitor Option	N	[320-0058]
First Hard Drive:	40GB 7.2K RPM IDE Hard Drive	40GB	[340-2243]
Operating System:	No Operating System, Other	NOOS/O	[420-4106]
Mouse:	Logitec System Mouse, Gray	L	[310-3776]
1st Network Adapter:	Onboard NIC	OB	[460-6604]
CD ROM/DVD ROM:	48X IDE Internal CD ROM Drive	CD48X	[313-2602]
Documentation:	Electronic Documentation, PowerEdge 600SC	EDOCS	[310-0438]
Hard Drive Configurations:	Motherboard IDE, Ascending Order	MIDE	[340-6930]
Hardware Support Services:	1Yr Parts + Onsite Labor (Next Business Day) + 1Yr Technical Support	W111YOS	[900-9054] [950-3040]
Installation Support Services:	No Installation	NOINSTL	[900-9997]
Fourth Hard Drive:	50% OFF THE UPGRADE TO A 120GB 7.2K RPM IDE Additional Hard Drive	120GBAH	[460-6855]

Appendix B: rc.firewall script

```
# Enable IP forwarding
echo 1 > /proc/sys/net/ipv4/ip_forward

for interface in /proc/sys/net/ipv4/conf/*; do
    #Block ICMP redirects for all interfaces
    echo 0 > $interface/accept_redirects
    # Block IP Source Routing for all interfaces
    echo 0 > $interface/accept_source_route
    # Block IP spoofing for all interfaces
    echo 1 > $interface/rp_filter
done

# Block ICMP broadcasts
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
# Enable tcp SYN cookies protection
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
# Ignore ICMP Bogus error messages
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
# Log packets with oddball addresses

# Flush all rules for all chains on restart
iptables -v -F

# Flush all established connections in the nat and mangle tables
iptables -t nat -v -F
iptables -t mangle -v -F

# Delete any user-defined chains
iptables -v -X

# -----
# Define the environment Variables
# -----
eth3_IP="192.168.60.254"
eth2_IP="192.168.50.254"
eth1_IP="192.168.40.254"
eth0_IP="172.16.1.253"
ALL_INTERFACES="eth0 eth1 eth2 lo"

INTERNAL_NET="192.168.50.0/24"
SERVICE_NET="192.168.40.0/24"
WAN_NET="172.16.1.240/28"
VPN_NET="192.168.60.0/24"
ALL_INT_NET="192.168.40.0/24 192.168.50.0/24 192.168.60.0/24 192.168.70.0/24"
SPOOFED_ADDR="10.0.0.0/8 192.168.0.0/16 0.0.0.0/32 127.0.0.0/8 224.0.0.0/8
240.254.0.0/16"

ROUTER="172.16.1.254"

EXTERNAL_NTP_SERVERS="172.16.178.67 172.16.39.193"
PARTNER_IPs="172.16.33.39 172.16.79.106"
SUPPLIER_IPs="172.16.92.5 172.16.47.245 172.16.245.35"
IT_AIRCARDS="172.16.45.45 172.16.45.46 172.16.45.47 172.16.45.48 172.16.45.49"
```

```
MOBILE_WORKERS="172.16.45.123 172.16.45.134 172.16.45.76"
BADGUYS="172.16.200.201 172.16.205.56 "
ISP_DNS="172.16.1.200"
SFTP="172.16.1.241"
SYSLOGGER="192.168.70.101"
```

```
SERVICE_NTP="192.168.40.201"
SERVICE_SFTP="192.168.40.201"
SERVICE_DNS="192.168.40.202"
SERVICE_SMTP="192.168.40.203"
SERVICE_RPROXY="192.168.40.204"
SERVICE_PROXY="192.168.40.205"
WBE="192.168.40.206"
```

```
EXCHANGE_SERVER="192.168.50.3"
WEB_SERVER="192.168.50.4"
INFOWAVE_XCONNECTOR="192.168.50.7"
DDNS1="192.168.50.1"
DDNS2="192.168.50.2"
MNGT_IP="192.168.50.8"
WIN_TSERVER="192.168.50.9"
```

```
# -----
# Allow state matches to pass
# -----
iptables -A FORWARD -m state --state ESTABLISH,RELATED -j ACCEPT
iptables -A INPUT -m state --state ESTABLISH,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state ESTABLISH,RELATED -j ACCEPT

# -----
# Allow all local connections through
# -----
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -i lo -j ACCEPT

# -----
# NAT from DMZ to Internet
# -----
iptables -t nat -A POSTROUTING -s $SERVICE_NET -o eth0 -j SNAT --to-source $eth0_IP
iptables -t nat -A POSTROUTING -s $SERVICE_SFTP -o eth0 -j SNAT --to-source $SFTP

# -----
# Block and log all spoofed source IP traffic:
# -----
for x in $SPOOFED_ADDR; do
    iptables -A FORWARD -i eth0 -p tcp -s $x -d 0/0 -j LOG --log-prefix " Address spoofing: "
    iptables -A FORWARD -i eth0 -p tcp -s $x -d 0/0 -j DROP
done

# -----
# Block and log banned IP's:
# -----
for x in $BADGUYS; do
    iptables -A FORWARD -i eth0 -s $x -d 0/0 -j LOG --log-level info --log-prefix " BADGUY "
    iptables -A FORWARD -i eth0 -s $x -d 0/0 -j DROP
done
```

```

# -----
# Pass Syslog traffic, originating from the firewall
# -----
iptables -A OUTPUT -o eth2 -m state --state NEW -p udp -d $SYSLOGGER --dport 514 -j
ACCEPT

# -----
Allow syslog traffic from the router to the syslog server:
# -----
iptables -A FORWARD -i eth0 -o eth2 -m state --state NEW -p udp -s $ROUTER -d
$SYSLOGGER --dport 514 -j ACCEPT

# -----
# Allow firewall to be managed from the management server
# -----
iptables -A INPUT -m state --state NEW -p tcp -s $MNGT_IP -d $eth2_IP --dport 22 -j LOG --log-
prefix " Internal FW Management: "
iptables -A INPUT -m state --state NEW -p tcp -s $MNGT_IP -d $eth2_IP --dport 22 -j ACCEPT

# -----
# Allow firewall to be managed from IT AirCards
# -----
for x in $IT_AIRCARDS; do
iptables -A INPUT -m state --state NEW -p tcp -s $x -d $eth0_IP --dport 22 -j LOG --log-level info
--log-prefix " IT Remote Management "
iptables -A INPUT -m state --state NEW -p tcp -s $x -d $eth0_IP --dport 22 -j ACCEPT
done

# -----
# Log and pass SSH traffic to service network from Management Server
# -----
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW,ESTABLISHED -p tcp -s $MNGT_IP -
d $SERVICE_NET --dport 22 -j LOG --log-prefix " Service Net Management: "
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW,ESTABLISHED -p tcp -s $MNGT_IP -
d $SERVICE_NET --dport 22 -j ACCEPT

# -----
# Drop any suspicious activity – FOR AUDIT PURPOSES ONLY
# Do not use on a production firewall system!
# -----
#iptables -A FORWARD -m unclean -j DROP

# -----
# FOR AUDIT PURPOSES ONLY
# -----
# Log Scans
#iptables -A FORWARD -p tcp --tcp-flags ALL FIN,PSH,URG -j LOG --log-prefix " Xmas Scan : "
#iptables -A FORWARD -p tcp --tcp-flags ALL FIN -j LOG --log-prefix " FIN Scan: "
#iptables -A FORWARD -p tcp --tcp-flags ALL NONE -j LOG --log-prefix " Null Scan: "

# -----
# Port forward from external interface for udp and udp port 53 to Service DNS server
# -----
iptables -t nat -A PREROUTING -i eth0 -p udp -d $eth0_IP --dport 53 -j DNAT --to
$SERVICE_DNS:53

```

```

iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 53 -j DNAT --to
$SERVICE_DNS:53

# -----
# Pass port forward incoming TCP/53, from Internet, to Service DNS server
# -----
iptables -A FORWARD -i eth0 -o eth1 -p udp -d $SERVICE_DNS --dport 53 -j LOG --log-level info
--log-prefix " Ext DNS query : "
iptables -A FORWARD -i eth0 -o eth1 -p udp -d $SERVICE_DNS --dport 53 -j ACCEPT

# -----
# Allow zone transfer between ISP DNS and SERVICE DNS
# -----
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -s $ISP_DNS -d
$SERVICE_DNS --dport 53 -j LOG --log-prefix " Zone xfer : "
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -s $ISP_DNS -d
$SERVICE_DNS --dport 53 -j ACCEPT

# -----
# Pass DNS traffic from Service DNS server
# -----
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p udp -s $SERVICE_DNS -d 0/0 --
dport 53 -j LOG --log-prefix " Outgoing DNS traffic: "
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p udp -s $SERVICE_DNS -d 0/0 --
dport 53 -j ACCEPT

# -----
# Allow DNS traffic between DDNS servers and Service DNS server
# -----
iptables -A FORWARD -i eth2 -o eth1 -p udp -s $DDNS1 -d $SERVICE_DNS --dport 53 -j LOG --
log-prefix "DNS Recursion: "
iptables -A FORWARD -i eth2 -o eth1 -p udp -s $DDNS1 -d $SERVICE_DNS --dport 53 -j
ACCEPT
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p udp -s $DDNS2 -d
$SERVICE_DNS --dport 53 -j LOG --log-prefix "DNS Recursion: "
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p udp -s $DDNS2 -d
$SERVICE_DNS --dport 53 -j ACCEPT

# -----
# DNAT Reverse Proxy Server for ports 80 and 443
# -----
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 80 -j DNAT --to
$SERVICE_RPROXY:80
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 443 -j DNAT --to
$SERVICE_RPROXY:443

# -----
# Pass natted HTTP and SSL traffic through to DMZ
# -----
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -d $SERVICE_RPROXY -m
multiport --dport 80,443 -j LOG --log-prefix " Incoming HTTP, SSL: "
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -d $SERVICE_RPROXY -m
multiport --dport 80,443 -j ACCEPT

# -----
# Allow traffic to pass between the Reverse Web Proxy and the Web Server

```

```

# -----
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s $SERVICE_RPROXY -d
$WEB_SERVER -m multiport --dport 80,443 -j LOG --log-prefix " HTTP, HTTPS to Web: "
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s $SERVICE_RPROXY -d
$WEB_SERVER -m multiport --dport 80,443 -j ACCEPT

# -----
# Port forward tcp/25 from Incoming Internet connection to SMTP relay server
# -----
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 25 -j DNAT --to
$SERVICE_SMTP:25

# Allow natted SMTP traffic through to DMZ
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -d $SERVICE_SMTP --dport
25 -j LOG --log-prefix " Incoming SMTP: "
iptables -A FORWARD -i eth0 -o eth1 -m state --state NEW -p tcp -d $SERVICE_SMTP --dport
25 -j ACCEPT

# -----
# Pass SMTP traffic from SMTP Relay server to Exchange Server
# -----
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s $SERVICE_SMTP -d
$EXCHANGE_SERVER --dport 25 -j LOG --log-level info --log-prefix " SMTP-Xchange: "
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s $SERVICE_SMTP -d
$EXCHANGE_SERVER --dport 25 -j ACCEPT

# -----
# Pass SMTP traffic from Exchange Server to SMTP relay
# -----
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s $SERVICE_SMTP -d
$EXCHANGE_SERVER --dport 25 -j LOG --log-level info --log-prefix " Xchange-SMTP: "
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s $SERVICE_SMTP -d
$EXCHANGE_SERVER --dport 25 -j ACCEPT

# -----
# Allow smtp traffic from SERVICE_SMTP to the Internet
# -----
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s $SERVICE_SMTP -d 0/0 --
dport 25 -j LOG --log-prefix " SMTP Relay: "
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s $SERVICE_SMTP -d 0/0 --
dport 25 -j ACCEPT

# -----
# Allow all Internal requests to Squid Proxy on Service Network
# -----
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s $INTERNAL_NET -d
$SERVICE_PROXY --dport 8008 -j ACCEPT
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s $INTERNAL_NET -d
$SERVICE_PROXY --dport 8008 -j LOG --log-level info --log-prefix "Int Squid traffic : "

# -----
# Allow Squid Proxy to access everything on ports 80,443 and 21
# -----
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s $SERVICE_PROXY -d 0/0 -
m multiport --dport 80,443,21 -j LOG --log-prefix " Squid Proxy: "

```

```
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s $SERVICE_PROXY -d 0/0 -
m multiport --dport 80,443,21 -j ACCEPT
```

```
# -----
# DNAT SFTP address to SERVICE_FTP for incoming connections from the Internet
# -----
```

```
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $SFTP --dport 22 -j DNAT --to
$SERVICE_SFTP:22
```

```
# -----
# Log and allow Partners to access SFTP sever through port 22
# -----
```

```
for x in $PARTNER_IPs; do
iptables -A FORWARD -i eth0 -o eth1 -p tcp -s $x -d $SERVICE_SFTP --dport 22 -j LOG --log-
level info --log-prefix " Partner SFTP: "
iptables -A FORWARD -i eth0 -o eth1 -p tcp -s $x -d $SERVICE_SFTP --dport 22 -j ACCEPT
done
```

```
# -----
# Log and allow Suppliers to access SFTP sever through port 22
# -----
```

```
for x in $SUPPLIER_IPs; do
iptables -A FORWARD -i eth0 -p tcp -s $x -d $SERVICE_SFTP --dport 22 -j LOG --log-level info -
-log-prefix " Supplier SFTP: "
iptables -A FORWARD -i eth0 -p tcp -s $x/32 -d $SERVICE_SFTP --dport 22 -j ACCEPT
done
```

```
# -----
# Allow internal users to download/upload files to Secure FTP server
# -----
```

```
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s $INTERNAL_NET -d
$SERVICE_SFTP --dport 22 -j LOG --log-level info --log-prefix "Internal SFTP: "
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s $INTERNAL_NET -d
$SERVICE_SFTP --dport 22 -j ACCEPT
```

```
# -----
# Allow GIAC's NTP server to talk to the upstream external NTP servers
# -----
```

```
for x in $EXTERNAL_NTP_SERVERS; do
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -p tcp -s $SERVICE_NTP -d $x --
dport 123 -j ACCEPT
```

```
# -----
# Pass all internal NTP queries
# -----
```

```
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s $INTERNAL_NET -d
$SERVICE_NTP --dport 123 -j LOG --log-level info --log-prefix " NTP: "
iptables -A FORWARD -i eth2 -o eth1 -m state --state NEW -p tcp -s $INTERNAL_NET -d
$SERVICE_NTP --dport 123 -j ACCEPT
done
```

```
# -----
# Wireless Business Engine Traffic Rules
# -----
```

```
# Port forward tcp/2085 to Infowave Wireless Business Engine (WBE) server
```



```
iptables -t nat -A PREROUTING -i eth0 -p tcp -d $eth0_IP --dport 2085 -j DNAT --to $WBE:2085
```

```
for x in $MOBILE_WORKERS; do
# Log and pass authorized natted Infowave traffic through to DMZ
iptables -A FORWARD -m state --state NEW -p tcp -s $x -d $WBE --dport 2085 -j LOG --log-level
info --log-prefix "Incoming Infowave: "
iptables -A FORWARD -m state --state NEW -p tcp -s $x -d $WBE --dport 2085 -j ACCEPT
done
```

```
# -----
# Pass Infowave traffic between WBE and Infowave Exchange Connector
# -----
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s $WBE -d
$INFOWAVE_XCONNECTOR --dport 2085 -j LOG --log-level info --log-prefix "WBE-XConnector:
"
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -p tcp -s $WBE -d
$INFOWAVE_XCONNECTOR --dport 2085 -j ACCEPT
```

```
# -----
# Allow VPN traffic to Windows Terminal Server only
# -----
iptables -A FORWARD -i eth3 -o eth2 -m state --state NEW -p tcp -d $WIN_TSERVER --dport
3389 -j LOG --log-level info --log-prefix "VPN-WTS: "
iptables -A FORWARD -i eth3 -o eth2 -m state --state NEW -p tcp -d $WIN_TSERVER --dport
3389 -j ACCEPT
```

```
# -----
# Catch all rule, all other packets are logged and dropped
# -----
iptables -A FORWARD -j LOG --log-level info --log-prefix " Default Drop (FWD): "
#iptables -A FORWARD -j DROP
iptables -A INPUT -j LOG --log-level info --log-prefix " Default Drop (IN): "
#iptables -A INPUT -j DROP
iptables -A OUTPUT -j LOG --log-level info --log-prefix " Default Drop (OUT): "
#iptables -A OUTPUT -j DROP
```

```
# -----
# Set default policies to drop everything
# -----
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
```

Appendix C: Chris Brenton's Sample Netfilter Files

```
# Flush all old rules on restart
iptables -F INPUT
iptables -F OUTPUT
iptables -F FORWARD

# Allow all state matches through
# -----FORWARD RULES-----
#
iptables -A FORWARD -m state --state ESTABLISH,RELATED -j ACCEPT

# -----NEW TEST LOGGING RULES-----
#
iptables -A FORWARD -p tcp --tcp-flags ALL SYN,FIN -j LOG --log-prefix " SYNFINSCAN "
iptables -A FORWARD -p tcp --tcp-flags ALL SYN,ACK -d 2.3.246.0/23 -j LOG --log-prefix "
SYNACK "
iptables -A FORWARD -p tcp --tcp-flags ALL FIN -j LOG --log-prefix " FINSCAN "
iptables -A FORWARD -p tcp --tcp-flags ALL NONE -j LOG --log-prefix " NULLSCAN "
iptables -A FORWARD -p tcp --tcp-flags ALL FIN,PSH,URG -j LOG --log-prefix " NMAPXMAS "
iptables -A FORWARD -p icmp -f -j LOG --log-prefix " ICMPFRAG "
iptables -A FORWARD -p tcp -s 0/0 --sport 31789 -d 0/0 --dport 31789 -j LOG --log-prefix "
HACKATAACK "
iptables -A FORWARD -p tcp -d 0/0 --dport 12345:12346 -j LOG --log-prefix " NETBUS "
iptables -A FORWARD -p tcp -d 2.3.246.0/23 --dport 1080 -j LOG --log-prefix " PROXYSCAN "
iptables -A FORWARD -p tcp -d 2.3.246.0/23 --dport 8080 -j LOG --log-prefix " PROXYSCAN "
iptables -A FORWARD -p tcp -d 0/0 --dport 110 -j LOG --log-prefix " POP3SCAN "
iptables -A FORWARD -p udp -s 0/0 -d 0/0 --dport 500 -j LOG --log-prefix " IPSEC "

# Block all private source IP packets
iptables -A FORWARD -i eth0 -p tcp -s 192.168.0.0/16 -d 2.3.246.0/23 -j DROP
iptables -A FORWARD -i eth0 -p tcp -s 172.16.0.0/12 -d 2.3.246.0/23 -j DROP
iptables -A FORWARD -i eth0 -p tcp -s 10.0.0.0/8 -d 2.3.246.0/23 -j DROP
iptables -A FORWARD -i eth0 -p tcp -s 0.0.0.0/32 -d 2.3.246.0/23 -j DROP
iptables -A FORWARD -i eth0 -p tcp -s 255.255.255.255/32 -d 2.3.246.0/23 -j DROP
iptables -A FORWARD -i eth0 -p tcp -s 127.0.0.0/8 -d 2.3.246.0/23 -j DROP

# Banned IP addresses
# Use a for loop to ban legal IP's
for ADDRESS in 216.242.155.144/24 194.186.145.0/24 129.170.39.182 217.10.38.0/24
203.166.96.235 129.170.246.0/24 129.170.76.0/23 61.177.70.237 200.151.29.0/24
213.233.75.0/24 210.172.82.122 195.152.174.144 148.81.24.0/24 217.97.92.128/25
61.163.229.154 203.253.193.236 207.21.247.4 200.128.0.0/16 200.158.0.0/16 213.23.97.14
24.207.211.131 210.97.5.252 211.73.137.60 194.143.41.132 208.42.136.125 208.41.144.178
216.43.203.230 63.236.0.180 63.70.170.62 204.212.245.6 200.177.162.131 210.96.87.194
193.226.102.0/24 212.124.136.0/25 207.35.158.26 212.244.177.0/25 63.49.116.0/24
200.56.242.0/24 216.114.181.231 207.228.64.4; do
iptables -A FORWARD -i eth0 -s $ADDRESS -d 0/0 -j LOG --log-level info --log-prefix " BADGUY
"
iptables -A FORWARD -i eth0 -s $ADDRESS -d 0/0 -j DROP
done

# Or you can block them in a more conventional way
```

```

iptables -A FORWARD -i eth0 -s 64.170.229.16/28 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 64.170.229.16/28 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 217.135.183.0/24 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 217.135.183.0/24 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 64.152.164.0/24 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 64.152.164.0/24 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 209.255.50.0/24 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 209.255.50.0/24 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 217.135.156.0/24 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 217.135.156.0/24 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 158.252.127.0/24 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 158.252.127.0/24 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 216.53.208.0/21 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 216.53.208.0/21 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 24.242.82.0/24 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 24.242.82.0/24 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 203.121.130.0/24 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 203.121.130.0/24 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 61.144.0.0/16 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 61.144.0.0/16 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 61.140.0.0/16 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 61.140.0.0/16 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 202.102.0.0/16 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 202.102.0.0/16 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 210.100.0.0/14 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 210.100.0.0/14 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 210.104.0.0/14 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 210.104.0.0/14 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 207.69.112.0/23 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 207.69.112.0/23 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 216.129.51.0/24 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 216.129.51.0/24 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 211.100.0.0/16 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 211.100.0.0/16 -d 0/0 -j DROP
iptables -A FORWARD -i eth0 -s 217.80.26.0/24 -d 0/0 -j LOG --log-level info --log-prefix "
BADGUY "
iptables -A FORWARD -i eth0 -s 217.80.26.0/24 -d 0/0 -j DROP

```

All high volume rules in both directions

```

iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 --sport 1024:65535 -d 2.3.247.6/32 --dport 80 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p udp -s 2.3.246.0/23 -d 2.3.246.10/32 --dport 514 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p udp -s 2.3.247.3/32 -d 0/0 --dport 53 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 --sport 1024:65535 -d 2.3.247.4/32 --dport 80 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p udp -s 0/0 -d 2.3.246.130/32 --dport 53 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p udp -s 0/0 -d 2.3.246.131/32 --dport 53 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 --sport 1024:65535 -d 2.3.247.7/32 --dport 80 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.3/32 -d 0/0 --dport 25 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p udp -s 2.3.246.0/23 -d 2.3.246.130/32 --dport 123 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p udp -s 2.3.246.0/23 -d 2.3.246.131/32 --dport 123 -j ACCEPT

```

Ignore NetBIOS probes due to high noise

```

iptables -A FORWARD -p udp -s 0/0 -d 2.3.246.0/23 --dport 135:139 -j DROP
iptables -A FORWARD -p tcp -s 0/0 -d 2.3.246.0/23 --dport 135:139 -j DROP
iptables -A FORWARD -p tcp -s 0/0 -d 2.3.246.0/23 --dport 445 -j DROP

```

Inbound services to one of the service networks

```

iptables -A FORWARD -m state --state NEW -p tcp -s 220.10.43.49/32 --sport 1024:65535 -d 2.3.247.10/32 --dport 80 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 --sport 1024:65535 -d 2.3.247.2/32 --dport 80 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.11/32 --dport 22 -j LOG --log-level info --log-prefix "SSH_fubar "
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.0/28 --dport 25 -j LOG --log-level info --log-prefix "MAIL_fubar "
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.0/28 --dport 443 -j LOG --log-level info --log-prefix "SSL_fubar "
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.11/32 --dport 22 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 148.75.63.82/32 -d 2.3.247.8/32 --dport 22 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.6/32 --dport 443 -j DROP
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.0/28 --dport 443 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.12 --dport 443 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p udp -s 0/0 -d 2.3.247.3/32 --dport 53 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p udp -s 63.100.47.51 -d 2.3.247.11 --dport 1812 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 --sport 1024:65535 -d 2.3.247.3/32 --dport 25 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 --sport 1024:65535 -d 2.3.247.10/32 --dport 80 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 --sport 1024:65535 -d 2.3.247.10/32 --dport 443 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.10/32 --dport 22 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.10/32 --dport 25 -j ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.247.2/32 --dport 8080 -j ACCEPT

```

```

# Outbound services from that service network
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 206.253.210.201/32
-j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.7/32 -d 206.253.210.201/32
-j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.12/32 -d 0/0 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 129.6.13.136/32 --
dport 80 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 130.85.6.7/32 --
dport 80 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 153.2.228.50/32 --
dport 80 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 153.2.224.50/32 --
dport 80 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 153.2.228.50/32 --
dport 443 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 153.2.224.50/32 --
dport 443 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.3/32 -d 0/0 --dport 53 -j
ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.8/32 -d 0/0 --dport 22 -j
ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.0/28 -d 0/0 --dport 79 -j
ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.0/28 -d 2.3.246.130/32 --
dport 53 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.0/28 -d 2.3.246.131/32 --
dport 53 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 209.140.49.4/32 --
dport 50 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 0/0 --dport 43 -j
ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.2/32 -d 0/0 --dport 43 -j
ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.9/32 -d 0/0 --dport 43 -j
ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 206.253.210.200/32
--dport 443 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.7/32 -d 206.253.210.200/32
--dport 443 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.3/32 -d 208.185.125.0/24 --
dport 80 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.3/32 -d 216.33.22.0/24 --
dport 80 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.3/32 -d 65.200.202.0/24 --
dport 80 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.9/32 -d 66.129.1.101/32 --
dport 22 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 66.129.1.101/32 --
dport 50 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.4/32 -d 129.170.248.235/32
--dport 80 -j ACCEPT
iptables -A FORWARD -i eth3 -m state --state NEW -p tcp -s 2.3.247.10/32 -d 0/0 --dport 25 -j
ACCEPT

```

```

# Let Service network machines do their thing
iptables -A FORWARD -i eth1 -m state --state NEW -p udp -s 2.3.246.130/32 -d 0/0 --dport 53 -j
ACCEPT
iptables -A FORWARD -i eth1 -m state --state NEW -p tcp -s 2.3.246.130/32 -d 0/0 --dport 53 -j
ACCEPT
iptables -A FORWARD -i eth1 -m state --state NEW -p udp -s 2.3.246.131/32 -d 0/0 --dport 53 -j
ACCEPT
iptables -A FORWARD -i eth1 -m state --state NEW -p tcp -s 2.3.246.131/32 -d 0/0 --dport 53 -j
ACCEPT
iptables -A FORWARD -i eth1 -m state --state NEW -p tcp -s 2.3.246.131/32 -d 0/0 --dport 25 -j
ACCEPT

# Let external users access services
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.10/32 --dport 22 -j LOG --
log-level info --log-prefix " SSH_Admin1 "
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.10/32 --dport 22 -j ACCEPT
# iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.132/32 --dport 80 -j
ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.131/32 --dport 25 -j LOG --
log-level info --log-prefix " MAIL "
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.131/32 --dport 25 -j
ACCEPT
iptables -A FORWARD -m state --state NEW -p udp -s 63.79.58.145/32 -d 2.3.246.10/32 --dport
514 -j ACCEPT

# Fix authentication delays
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 0/0 --dport 113 -j LOG --log-level info
--log-prefix " IDENT "
iptables -A FORWARD -m state --state NEW -p tcp -s 2.3.246.0/23 -d 0/0 --dport 113 -j ACCEPT
iptables -A FORWARD -p tcp -s 0/0 -d 2.3.246.0/23 --dport 113 -j REJECT --reject-with tcp-reset

# Allow NTP servers to talk to the off-site feeder servers
iptables -A FORWARD -i eth1 -m state --state NEW -p udp -s 2.3.246.130/32 -d
216.152.230.3/32 --dport 123 -j ACCEPT
iptables -A FORWARD -i eth1 -m state --state NEW -p udp -s 2.3.246.131/32 -d
216.152.230.3/32 --dport 123 -j ACCEPT
iptables -A FORWARD -i eth1 -m state --state NEW -p udp -s 2.3.246.130/32 -d
130.207.244.240/32 --dport 123 -j ACCEPT
iptables -A FORWARD -i eth1 -m state --state NEW -p udp -s 2.3.246.131/32 -d
130.207.244.240/32 --dport 123 -j ACCEPT
iptables -A FORWARD -i eth1 -m state --state NEW -p udp -s 2.3.246.130/32 -d
128.59.35.142/32 --dport 123 -j ACCEPT
iptables -A FORWARD -i eth1 -m state --state NEW -p udp -s 2.3.246.131/32 -d
128.59.35.142/32 --dport 123 -j ACCEPT

# Let internal users do anything (bad idea)
iptables -A FORWARD -i eth2 -m state --state NEW -s 2.3.246.1/26 -d 0/0 -j ACCEPT

# Setup honeypot ports
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.134/32 --dport 21 -j
ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.134/32 --dport 23 -j
ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.134/32 --dport 25 -j
ACCEPT

```

```

iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.134/32 --dport 80 -j
ACCEPT
iptables -A FORWARD -m state --state NEW -p tcp -s 0/0 -d 2.3.246.134/32 --dport 31337 -j
ACCEPT
iptables -A FORWARD -m state --state NEW -p udp -s 0/0 -d 2.3.246.134/32 --dport 31337 -j
ACCEPT

```

Log all non-matches

```

iptables -A FORWARD -p tcp -d 0/0 --dport 21 -j LOG --log-prefix " FTPSCAN "
iptables -A FORWARD -p tcp -d 0/0 --dport 22 -j LOG --log-prefix " SSHSCAN "
iptables -A FORWARD -p tcp -d 0/0 --dport 25 -j LOG --log-prefix " BAD_MAIL "
iptables -A FORWARD -p tcp -d 0/0 --dport 80 -j LOG --log-prefix " WEBSCAN "
iptables -A FORWARD -p tcp -d 0/0 --dport 110 -j LOG --log-prefix " RPCSCAN "
iptables -A FORWARD -s 2.3.247.0/28 -d 0/0 -j LOG --log-level info --log-prefix " BADfubar "
iptables -A FORWARD -s 2.3.247.0/28 -d 0/0 -j DROP
iptables -A FORWARD -s 0/0 -j LOG --log-level info --log-prefix " DROP_FORWARD "

```

-----INPUT RULES-----

#

Ignore internal bootp & NetBIOS traffic

```

iptables -A INPUT -i eth2 -s 2.3.246.0/26 -d 2.3.246.63/32 -j DROP
iptables -A INPUT -i eth1 -s 2.3.246.128/27 -d 2.3.246.159/32 -j DROP

```

```

iptables -A INPUT -m state --state ESTABLISH,RELATED -j ACCEPT

```

```

iptables -A INPUT -p udp -s 127.0.0.1/32 -d 127.0.0.1/32 --dport 123 -j ACCEPT

```

```

iptables -A INPUT -i eth3 -s 2.3.247.0/28 -d 2.3.247.15/32 -j DROP

```

```

iptables -A INPUT -s 0/0 -d 255.255.255.255 -j DROP

```

Allow firewall to be managed

```

iptables -A INPUT -m state --state NEW -p tcp -s 2.3.246.10 -d 2.3.246.1 --dport 22 -j ACCEPT

```

Log all non-matched packets

```

iptables -A INPUT -s 0/0 -j LOG --log-level info --log-prefix " DROP_INPUT "

```

-----OUTPUT RULES-----

#

Permit log entries to be submitted

```

iptables -A OUTPUT -p udp -s 2.3.246.1/32 -d 2.3.246.10 --dport 514 -j ACCEPT

```

Permit State matches

```

iptables -A OUTPUT -m state --state ESTABLISH,RELATED -j ACCEPT

```

Drop all outbound unreachable

```

iptables -A OUTPUT -p icmp -s 2.3.246.249/32 -d 0/0 -j DROP

```

Allow time sync

```

iptables -A OUTPUT -p udp -s 0/0 -d 2.3.246.130 --dport 123 -j ACCEPT

```

```

iptables -A OUTPUT -p udp -s 0/0 -d 2.3.246.131 --dport 123 -j ACCEPT

```

```
# Log all non-matched packets
iptables -A OUTPUT -s 0/0 -j LOG --log-level info --log-prefix " DROP_OUTPUT "

# Don't think these are used
#iptables -A OUTPUT -m state --state NEW -s 2.3.246.1 -d 0/0 -j ACCEPT
#iptables -A OUTPUT -m state --state NEW -s 2.3.246.249 -d 0/0 -j ACCEPT
#iptables -A OUTPUT -m state --state NEW -s 2.3.247.1 -d 0/0 -j ACCEPT
#iptables -A OUTPUT -m state --state NEW -s 2.3.246.129 -d 0/0 -j ACCEPT
iptables -A OUTPUT -p udp -s 127.0.0.1/32 -d 127.0.0.1/32 --dport 123 -j ACCEPT

# -----POLICY RULES-----
#
# Define default policy for all chains
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

iptables -L -n
```

© SANS Institute 2003, Author retains full rights.

Appendix D: Netfilter Log Entries for the Firewall Audit

-----Firewall Management Rule-----

Feb 5 13:21:16 Obfuscate kernel: IT Remote Management IN=eth0 OUT=
MAC=00:09:5b:09:fa:33:00:60:97:8b:ea:d2:08:00 SRC=172.16.45.45 DST=172.16.1.253 LEN=40
TOS=0x00 PREC=0x00 TTL=49 ID=34999 PROTO=TCP SPT=57884 DPT=22 WINDOW=2048
RES=0x00 SYN URGP=0

Feb 5 13:26:57 Obfuscate kernel: Default Drop (IN): IN=eth0 OUT=
MAC=00:09:5b:09:fa:33:00:60:97:8b:ea:d2:08:00 SRC=172.16.44.45 DST=172.16.1.253 LEN=40
TOS=0x00 PREC=0x00 TTL=47 ID=30220 PROTO=TCP SPT=60826 DPT=22 WINDOW=4096
RES=0x00 SYN URGP=0

[root@Obfuscate log]# Internal FW Management: IN=eth2 OUT=
MAC=00:07:e9:b0:2e:1d:00:60:97:8b:ea:d2:08:00 SRC=192.168.50.8 DST=192.168.50.254
LEN=40 TOS=0x00 PREC=0x00 TTL=40 ID=8733 PROTO=TCP SPT=33230 DPT=22
WINDOW=1024 RES=0x00 SYN URGP=0

Feb 5 13:48:57 Obfuscate kernel: Default Drop (IN): IN=eth2 OUT=
MAC=00:07:e9:b0:2e:1d:00:60:97:8b:ea:d2:08:00 SRC=192.168.50.100 DST=192.168.50.254
LEN=40 TOS=0x00 PREC=0x00 TTL=52 ID=16850 PROTO=TCP SPT=48491 DPT=22
WINDOW=1024 RES=0x00 SYN URGP=0

-----Service Net Management-----

Feb 5 14:23:01 Obfuscate kernel: Service Net Management: IN=eth2 OUT=eth1
SRC=192.168.50.8 DST=192.168.40.202 LEN=40 TOS=0x00 PREC=0x00 TTL=39 ID=50041
PROTO=TCP SPT=42612 DPT=22 WINDOW=1024 RES=0x00 SYN URGP=0

Feb 5 14:24:36 Obfuscate kernel: Default Drop (FWD): IN=eth2 OUT=eth1
SRC=192.168.50.100 DST=192.168.40.202 LEN=40 TOS=0x00 PREC=0x00 TTL=49 ID=14045
PROTO=TCP SPT=43181 DPT=22 WINDOW=3072 RES=0x00 SYN URGP=0

-----Internal Employee Access-----

Feb 4 13:33:46 Obfuscate kernel: Int Squid Traffic: IN=eth2 OUT=eth1 SRC=192.168.50.188
DST=192.168.40.205 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=28270 DF PROTO=TCP
SPT=32783 DPT=8008 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 4 13:36:26 Obfuscate kernel: Default Drop (FWD): IN=eth2 OUT=eth1
SRC=192.168.50.188 DST=192.168.40.205 LEN=48 TOS=0x00 PREC=0x00 TTL=127
ID=14667 DF PROTO=TCP SPT=1106 DPT=80 WINDOW=16384 RES=0x00 SYN URGP=0

Feb 4 13:40:11 Obfuscate kernel: Default Drop (FWD): IN=eth2 OUT=eth1
SRC=192.168.50.188 DST=192.168.40.205 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=23820
DF PROTO=TCP SPT=32786 DPT=21 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 4 13:41:16 Obfuscate kernel: Drop All (FWD): IN=eth2 OUT=eth1 SRC=192.168.50.188
DST=192.168.40.206 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=9451 DF PROTO=TCP
SPT=32785 DPT=8008 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 4 13:55:45 Obfuscate kernel: Drop All (FWD): IN=eth2 OUT=eth1 SRC=192.168.50.188
DST=192.168.40.203 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=36234 DF PROTO=TCP
SPT=32788 DPT=25 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 4 14:01:15 Obfuscate kernel: Default Drop (FWD): IN=eth2 OUT=eth0
SRC=192.168.50.188 DST=172.16.1.250 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=9143 DF
PROTO=TCP SPT=32790 DPT=80 WINDOW=5840 RES=0x00 SYN URGP=0

-----Squid Proxy traffic to Internet -----

Feb 5 16:20:07 Obfuscate kernel: Squid Proxy: IN=eth1 OUT=eth0 SRC=192.168.40.205
DST=172.16.1.251 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=12683 DF PROTO=TCP
SPT=32782 DPT=80 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 5 16:21:16 Obfuscate kernel: Squid Proxy: IN=eth1 OUT=eth0 SRC=192.168.40.205
DST=172.16.1.251 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=31957 DF PROTO=TCP
SPT=32783 DPT=443 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 5 16:21:30 Obfuscate kernel: Squid Proxy: IN=eth1 OUT=eth0 SRC=192.168.40.205
DST=172.16.1.251 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=45306 DF PROTO=TCP
SPT=32784 DPT=21 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 5 16:21:47 Obfuscate kernel: Default Drop (FWD): IN=eth1 OUT=eth0
SRC=192.168.40.205 DST=172.16.1.251 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=40559 DF
PROTO=TCP SPT=32785 DPT=23 WINDOW=5840 RES=0x00 SYN URGP=0

-----Incoming DNS Query-----

Feb 5 03:25:00 Obfuscate kernel: Ext DNS query : IN=eth0 OUT=eth1 SRC=172.16.1.251
DST=192.168.40.202 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=906 DF PROTO=TCP
SPT=32781 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 5 03:28:28 Obfuscate kernel: Ext DNS query : IN=eth0 OUT=eth1 SRC=172.16.1.251
DST=192.168.40.202 LEN=28 TOS=0x00 PREC=0x00 TTL=51 ID=12413 PROTO=UDP
SPT=50862 DPT=53 LEN=8

-----Zone Transfer-----

Feb 5 04:14:14 Obfuscate kernel: Default Drop (FWD): IN=eth0 OUT=eth1 SRC=172.16.1.251
DST=192.168.40.202 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=23597 DF PROTO=TCP
SPT=32775 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 5 04:19:49 Obfuscate kernel: Zone xfer : IN=eth0 OUT=eth1 SRC=172.16.1.200
DST=192.168.40.202 LEN=40 TOS=0x00 PREC=0x00 TTL=51 ID=41569 PROTO=TCP
SPT=63988 DPT=53 WINDOW=1024 RES=0x00 SYN URGP=0

Feb 5 04:19:52 Obfuscate kernel: Default Drop (OUT): IN= OUT=lo SRC=172.16.1.253
DST=172.16.1.253 LEN=68 TOS=0x00 PREC=0xC0 TTL=64 ID=25923 PROTO=ICMP TYPE=3
CODE=1 [SRC=172.16.1.253 DST=172.16.1.200 LEN=40 TOS=0x00 PREC=0x00 TTL=127
ID=498 PROTO=TCP SPT=53 DPT=63988 WINDOW=0 RES=0x00 ACK RST URGP=0]

-----DNS Recursion-----

Feb 5 14:57:44 Obfuscate kernel: DNS Recursion: IN=eth2 OUT=eth1 SRC=192.168.50.1
DST=192.168.40.202 LEN=28 TOS=0x00 PREC=0x00 TTL=56 ID=32620 PROTO=UDP
SPT=34938 DPT=53 LEN=8

Feb 5 14:58:39 Obfuscate kernel: DNS Recursion: IN=eth2 OUT=eth1 SRC=192.168.50.2
DST=192.168.40.202 LEN=28 TOS=0x00 PREC=0x00 TTL=37 ID=36278 PROTO=UDP
SPT=36389 DPT=53 LEN=8

Feb 5 14:59:19 Obfuscate kernel: Default Drop (FWD): IN=eth2 OUT=eth1 SRC=192.168.50.3
DST=192.168.40.202 LEN=28 TOS=0x00 PREC=0x00 TTL=37 ID=35053 PROTO=UDP
SPT=47830 DPT=53 LEN=8

-----DNS Traffic from DNS server-----

Feb 5 15:18:59 Obfuscate kernel: Outgoing DNS traffic: IN=eth1 OUT=eth0
SRC=192.168.40.202 DST=172.16.1.251 LEN=28 TOS=0x00 PREC=0x00 TTL=45 ID=52466
PROTO=UDP SPT=50113 DPT=53 LEN=8

Feb 5 15:21:54 Obfuscate kernel: Default Drop (FWD): IN=eth1 OUT=eth0
SRC=192.168.40.205 DST=172.16.1.251 LEN=28 TOS=0x00 PREC=0x00 TTL=46 ID=56878
PROTO=UDP
SPT=54777 DPT=53 LEN=8

-----Incoming HTTP and HTTPS traffic-----

Feb 5 17:17:57 Obfuscate kernel: Incoming HTTP, SSL: IN=eth0 OUT=eth1 SRC=172.16.1.251
DST=192.168.40.204 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=15389 DF PROTO=TCP
SPT=32786 DPT=80 WINDOW=5840 RES=0x00 SYN URG=0

Feb 5 17:19:41 Obfuscate kernel: Incoming HTTP, SSL: IN=eth0 OUT=eth1 SRC=172.16.1.251
DST=192.168.40.204 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=47139 DF PROTO=TCP
SPT=32787 DPT=443 WINDOW=5840 RES=0x00 SYN URG=0

Feb 5 17:20:45 Obfuscate kernel: Default Drop (IN): IN=eth0 OUT=
MAC=00:09:5b:09:fa:33:00:60:97:8b:ea:d2:08:00 SRC=172.16.1.251 DST=172.16.1.253 LEN=60
TOS=0x00 PREC=0x00 TTL=64 ID=11148 DF PROTO=TCP SPT=32788 DPT=21
WINDOW=5840 RES=0x00
SYN URG=0

-----Inverse Squid Proxy traffic to Web Server-----

Feb 5 19:56:47 Obfuscate kernel: HTTP, HTTPS to Web: IN=eth1 OUT=eth2
SRC=192.168.40.204 DST=192.168.50.4 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=59707 DF
PROTO=TCP SPT=32806 DPT=80 WINDOW=5840 RES=0x00 SYN URG=0

Feb 5 19:57:05 Obfuscate kernel: HTTP, HTTPS to Web: IN=eth1 OUT=eth2
SRC=192.168.40.204 DST=192.168.50.4 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=593 DF
PROTO=TCP SPT=32807 DPT=443 WINDOW=5840 RES=0x00 SYN URG=0

-----Incoming SMTP traffic-----

Feb 5 17:40:51 Obfuscate kernel: Incoming SMTP: IN=eth0 OUT=eth1 SRC=172.16.1.251
DST=192.168.40.203 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=39147 DF PROTO=TCP
SPT=32790 DPT=25 WINDOW=5840 RES=0x00 SYN URG=0

Feb 5 17:42:30 Obfuscate kernel: Default Drop (IN): IN=eth0 OUT=
MAC=00:09:5b:09:fa:33:00:60:97:8b:ea:d2:08:00 SRC=172.16.1.251 DST=172.16.1.253 LEN=60

TOS=0x00 PREC=0x00 TTL=64 ID=7708 DF PROTO=TCP SPT=32791 DPT=10056
WINDOW=5840 RES=0x00 SYN URGP=0

-----SMTP Relay traffic to Internet-----

Feb 5 15:35:12 Obfuscate kernel: SMTP Relay: IN=eth1 OUT=eth0 SRC=192.168.40.203
DST=172.16.1.251 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=60689 DF PROTO=TCP
SPT=32777 DPT=25 WINDOW=5840 RES=0x00 SYN URGP=0

Feb 5 15:47:04 Obfuscate kernel: Default Drop (FWD): IN=eth1 OUT=eth0
SRC=192.168.40.100 DST=172.16.1.251 LEN=40 TOS=0x00 PREC=0x00 TTL=46 ID=15989
PROTO=TCP
SPT=59266 DPT=25 WINDOW=4096 RES=0x00 SYN URGP=0

-----SMTP Relay to Exchange Server-----

Feb 5 19:40:11 Obfuscate kernel: SMTP-Xchange: IN=eth1 OUT=eth2 SRC=192.168.40.203
DST=192.168.50.3 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=13306 DF PROTO=TCP
SPT=32802 DPT=25 WINDOW=5840 RES=0x00 SYN URGP=0

-----Incoming Secure FTP-----

Feb 5 18:14:57 Obfuscate kernel: Partner SFTP: IN=eth0 OUT=eth1 SRC=172.16.33.39
DST=192.168.40.201 LEN=40 TOS=0x00 PREC=0x00 TTL=38 ID=60407 PROTO=TCP
SPT=41587 DPT=22 WINDOW=4096 RES=0x00 SYN URGP=0

Feb 5 18:16:23 Obfuscate kernel: Supplier SFTP: IN=eth0 OUT=eth1 SRC=172.16.245.35
DST=192.168.40.201 LEN=40 TOS=0x00 PREC=0x00 TTL=48 ID=54394 PROTO=TCP
SPT=56971 DPT=22 WINDOW=2048 RES=0x00 SYN URGP=0

Feb 5 18:17:05 Obfuscate kernel: Default Drop (FWD): IN=eth0 OUT=eth1
SRC=172.16.100.100 DST=192.168.40.201 LEN=40 TOS=0x00 PREC=0x00 TTL=58 ID=29472
PROTO=TCP SPT=55692 DPT=22 WINDOW=4096 RES=0x00 SYN URGP=0

-----Secure FTP from internal network-----

Feb 5 19:29:52 Obfuscate kernel: Internal SFTP: IN=eth2 OUT=eth1 SRC=192.168.50.7
DST=192.168.40.201 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=32707 DF PROTO=TCP
SPT=32800 DPT=22 WINDOW=5840 RES=0x00 SYN URGP=0

-----NTP Traffic from Internal network-----

Feb 5 19:13:02 Obfuscate kernel: NTP: IN=eth2 OUT=eth1 SRC=192.168.50.188
DST=192.168.40.201 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=624 DF PROTO=TCP
SPT=32796 DPT=123 WINDOW=5840 RES=0x00 SYN URGP=0

-----Incoming Infowave Traffic-----

Feb 5 18:47:02 Obfuscate kernel: Incoming Infowave: IN=eth0 OUT=eth1 SRC=172.16.45.123
DST=192.168.40.206 LEN=40 TOS=0x00 PREC=0x00 TTL=49 ID=9248 PROTO=TCP
SPT=39720 DPT=2085 WINDOW=3072 RES=0x00 SYN URGP=0

Feb 5 18:48:00 Obfuscate kernel: Default Drop (FWD): IN=eth0 OUT=eth1
SRC=172.16.100.100 DST=192.168.40.206 LEN=40 TOS=0x00 PREC=0x00 TTL=53 ID=30811
PROTO=TCP SPT=36974 DPT=2085 WINDOW=3072 RES=0x00 SYN URG=0

-----Infowave WBE to Exchange Connector Traffic-----

Feb 5 19:45:16 Obfuscate kernel: WBE-XConnector: IN=eth1 OUT=eth2 SRC=192.168.40.206
DST=192.168.50.7 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=58195 DF PROTO=TCP
SPT=32803 DPT=2085 WINDOW=5840 RES=0x00 SYN URG=0

-----VPN Traffic-----

Feb 5 21:11:40 Obfuscate kernel: VPN-WTS: IN=eth3 OUT=eth2 SRC=192.168.60.60
DST=192.168.50.9 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=23072 DF PROTO=TCP
SPT=32775 DPT=3389 WINDOW=5840 RES=0x00 SYN URG=0

Feb 5 21:15:48 Obfuscate kernel: Default Drop (FWD): IN=eth3 OUT=eth2 SRC=192.168.60.60
DST=192.168.50.5 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=43638 DF PROTO=TCP
SPT=32776 DPT=80 WINDOW=5840 RES=0x00 SYN URG=0

-----Address Spoofing-----

Feb 9 19:09:45 Obfuscate kernel: Private address spoof IN=eth0 OUT=eth1 SRC=10.10.10.10
DST=192.168.40.204 LEN=40 TOS=0x00 PREC=0x00 TTL=54 ID=1797 PROTO=TCP
SPT=38416 DPT=80 WINDOW=4096 RES=0x00 SYN URG=0

Feb 9 19:14:40 Obfuscate kernel: Private address spoof IN=eth0 OUT=eth1 SRC=192.168.1.1
DST=192.168.40.204 LEN=40 TOS=0x00 PREC=0x00 TTL=39 ID=1002 PROTO=TCP
SPT=33417 DPT=80 WINDOW=1024 RES=0x00 SYN URG=0

-----Banned IP Addresses-----

Feb 9 19:37:17 Obfuscate kernel: BADGUY IN=eth0 OUT=eth1 SRC=172.16.205.56
DST=192.168.40.204 LEN=40 TOS=0x00 PREC=0x00 TTL=37 ID=35136 PROTO=TCP
SPT=45299 DPT=80 WINDOW=3072 RES=0x00 SYN URG=0

Feb 9 19:31:57 Obfuscate kernel: BADGUY IN=eth0 OUT=eth1 SRC=172.16.200.201
DST=192.168.40.204 LEN=40 TOS=0x00 PREC=0x00 TTL=46 ID=34152 PROTO=TCP
SPT=53757 DPT=80 WINDOW=4096 RES=0x00 SYN URG=0

© SANS Institute 2003, As part of GIAC practical repository.