Global Information Assurance Certification Paper

GIAC Enterprise
Security Architecture and Policy

SANS GIAC Certification
GCFW Practical Assignment
Version 2.0

Grace Foo
November 20, 2003

Table of Contents

Abstract

This paper is to fulfill the practical assignment requirement (version 2.0) for the GCFW certification. It consists of four parts/assignments:

1. the network security architecture for a fictitious Company, GIAC Enterprises. This includes a description of the business/access requirements of its users.
2. the security policy for border router, primary firewall and VPN of the architecture defined. A tutorial for the firewall policy is included.
3. verification of the security policy defined for the primary firewall
4. security evaluation of a previous network design by designing attacks

Assignment 1 – Security Architecture

1.1 Introduction

GIAC Enterprises (GE) is an e-business dealing in the online sale of fortune cookie sayings ("cookies"). As a modern e-business, efficient, reliable, highly available (24x7) and secure network/ internet communications lie at the heart of its business operations. The network architecture must meet the access requirements of its users listed here:

- Customers (who bulk purchase GE's online fortunes)
- Suppliers (who supply GE with fortune cookie sayings)
- Partners (international companies that translate and resell the fortunes)
- GIAC Enterprise's employees on GE's internal networks
- GIAC Enterprise's mobile sales force and teleworkers
- The general public

Additionally, a secure network architecture is needed to meet data confidentiality, integrity and availability requirements. The GE network architecture includes the following components:

- Filtering router
- External and internal firewalls
- VPN
- IP addressing scheme (non-routable addresses internally, routable externally)

In section 1.2, the access requirements for the users above are described. The security architecture design is detailed in section 1.3, while section 1.4 lists the choices of components used.

1.2 Access Requirements

As a modern, global e-business, the GIAC Enterprises network and services has to be available 24 hours a day. The network security architecture must meet the business requirements of its users. These user requirements are described next, and we also look at the infrastructure that would be needed to support them.

Users

- **Customers**

These are companies or individuals that bulk purchase fortune cookie sayings online through the internet. Access to the application on the GE web server to order and download cookies requires login ID and password, and transactions must be secure to ensure confidentiality, integrity. They may also communicate with GE through email.

- **Suppliers**

These companies make and sell the fortune cookie sayings to GIAC Enterprises. On receiving an order from GE, say through email, they upload their fortune cookie sayings through the cookie upload application running on the GE web server. Access requires login id and password, and transactions must be secure.

- **Partners**

These are international companies that translate and resell fortunes. As with customers and suppliers, they transact with GE through the internet. Access to the GE web application to transact and download the fortune cookie sayings, requires login ID and password. Transactions must be secure.

- **Employees on internal network**

They perform various corporate/administrative functions using applications running on servers in the GE internal network. Access to applications, files and databases require the proper authorization. They also have access to email and print services as well as internet/web and ftp.

System administrators of GE network/servers must have proper root login access to the servers/systems they administer. They use secure shell (ssh) to access the servers

- **Mobile sales force and teleworkers**

These GE employees have same requirements as employees on internal network. They have SecureClient installed on their notebooks to connect to the GE VPN.

- **General public**

They browse GE's web site for information and they may communicate with GE through email.

Therefore, we have these preliminary service requirements:

Description	Protocol / Application	Port
Web, secure web access	HTTP HTTPS	TCP/80 TCP/443
File transfers	FTP	TCP/20, 21
Mail	Sendmail	TCP/25
DNS	Bind	TCP/53, UDP/53
Database	PostgreSQL	TCP/4871
VPN	IPSEC	UDP/500, IP/50
Secure access	SSH	TCP/22

Infrastructure

The following servers/components are needed:

- **Reverse web proxy server (ge_web_ext) / web server (ge_web_int)**

A reverse web proxy is set up for the GE web site. The customers, suppliers, partners connect via SSL (HTTPS) for secure transactions with GE. The reverse proxy accesses the web server in the internal service network, which in turn accesses the database server. Thus, there is no direct connection to the web and database servers. The reverse web proxy can also filter out malformed traffic, viruses etc.

The general public use HTTP protocol to connect to the reverse web proxy.

- **Mail relay (ge_mail_ext) / mail server (ge_mail_int)**

Mail to GE is sent to the SMTP relay on the external network which relays it to the mail server on the internal service network. There is no direct connection to the mail server. The mail server is also protected by a virus scanner.

- **External DNS (ge_dns_ext) / internal DNS (ge_dns_int)**

Split DNS is used for hostname resolution. The internal DNS has entries for internal hosts and external DNS has entries for external hosts.

- **Network time server (ge_ntp_int)**

A time server is used to synchronize time among the servers and other network devices, using network time protocol (NTP).

- **VPN**

Provide remote employees a secure tunnel to connect to the internal network.

- **Database (ge_dbs_adm)**

The database houses customers', suppliers', partners' authentication and transaction data.

- **System administrator's workstation (ge_SA_adm)**

Workstations used by GE system administrators of GE servers and network.

- **Central log server (ge_log_adm)**

Central repository/server for system logs from servers and other network devices.

- **Internal servers/workstations (ge_wks_usr)**

These workstations, file/print servers are used by GE employees on for various administrative tasks. A single workstation is used to represent this resource.

- **Web proxy server (ge_wbx_usr)**

Users of GE internal networks access the internet through a web proxy.

From above, it is seen that we have segmented GE network into logical parts: external service, internal service, admin and user, with components shown, in the following notation:

External service: **ge_ext** (ge_web_ext, ge_mail_ext, ge_dns_ext)
Internal service: **ge_int** (ge_web_int, ge_mail_int, ge_dns_int, ge_ntp_int)
Admin: **ge_adm** (ge_SA_adm, ge_log_adm, ge_dbs_adm)
User: **ge_usr** (ge_wks_usr, ge_wbx_usr)

The internal service, admin and user subnets together form GE's internal networks. Some other notations we use:

GP: general public, anyone on web
CSP: GE customers, suppliers, partners
GIU: GE internal users
GRU: GE remote users
GE network: **ge_net** (ge_exts, ge_ints, ge_usr, ge_adm)
Security devices: **ge_sec** (router, primary firewall, secondary firewall)

Services and Network traffic

Summary of the service requirements:

Service	Port	Direction	Description
HTTP	TCP/80	ge_wbx_usr to internet Internet to ge_web_ext ge_web_ext to ge_web_int	GIU accessing web GP, CSP accessing GE web Reverse web proxying

HTTPS	TCP/4	ge_wbx_usr to internet Internet to ge_web_ext ge_web_ext to ge_web	GIU accessing web GP, CSP accessing GE web Reverse web proxying
FTP	TCP/20,21	ge_wbx_usr to internet	GIU accessing internet
Sendmail	TCP/25	ge_mail_ext to internet Internet to ge_web_ext ge_mail_ext to ge_mail_int ge_mail_int to ge_web_ext	Outbound email Inbound email mail relay mail relay
Bind	TCP/53, UDP/53	ge_dns_ext to internet internet to ge_dns_ext ge_dns_int to ge_dns_ext ge_int, ge_usr, ge_adm to ge_dns_int	External DNS services Internal DNS services
PostgreSQL	TCP/4871	ge_web_int to ge_dbs_adm	Web server to DB server
IKE, ESP	UDP/500, IP/50	GRU to ge_int, ge_adm, ge_usr	VPN connection between remote employees and GE internal networks
SSH	TCP/22	ge_SA_adm to ge_net, ge_sec	System administration of any server on GE net including security devices
NTP	UDP/123	ge_net, ge_sec to ge_ntp_int ge_ntp_int to stratum 2 NTP servers	GE servers/devices synching time with NTP server NTP server synching with stratum 2 NTP servers
Syslog	UDP/514	ge_net, ge_sec to ge_log_adm	GE server/device logs to central log server

Network Traffic patterns:

	Incoming zone	**Outgoing zone**	**Source**	**Destination**	**Service/ Port**
1	Internet	External service	any	ge_web_ext	HTTP TCP/80 HTTPS TCP/443
2	Internet	External service	any	ge_mail_ext	SMTP TCP/25

3	Internet	External service	any	ge_dns_ext	Bind TCP/53, UDP/53
4	External service	Internet	ge_mail_ext	any	SMTP TCP/25
5	External service	Internet	ge_dns_ext	any	Bind TCP/53, UDP/53
6	External service	Internal service	ge_mail_ext	ge_mail_int	SMTP TCP/25
7	External service	Internal service	ge_web_ext	ge_web_int	HTTP TCP/80 HTTPS TCP/443
8	Internal service	External service	ge_mail_int	ge_mail_ext	SMTP TCP/25
9	Internal service	External service	ge_dns_int	ge_dns_ext	Bind TCP/53, UDP/53
10	Internal service	Admin	ge_web_int	ge_dbs_adm	PostgreSQL TCP/4871
11	Admin	Internal service	ge_dbs_adm	ge_web_int	PostgreSQL TCP/4871
12	Admin, User	Internal service	ge_adm, ge_usr	ge_dns_int	Bind TCP/53, UDP/53
13	User	Internet	ge_wbx_usr	any	HTTP TCP/80 HTTPS TCP/443 FTP TCP/20,21
14	Admin	All zones	ge_SA_adm	ge_sec ge_net	SSH TCP/22
15	All zones	Internal service	ge_sec ge_net	ge_ntp_int	NTP UDP/123
16	All zones	Admin	ge_sec ge_net	ge_log_adm	Syslog UDP/514
17	GRU	VPN gateway	VPN clients	Secondary firewall	IKE, UDP/500, ESP, IP/50

1.3 Security Architecture Design

In this section, we get into the details of the GE network design that was outlined in the previous section.

General design principles

- provide security at each layer of TCP/IP stack
- use open source / standards technologies where possible
- everything explicitly denied, unless explicitly allowed
- design for maintainability, flexibility
- be a good Internet neighbor

In designing the GE network, a balance is taken between meeting the business and security needs of the company and the cost. Included in cost considerations are not only upfront costs of the components/devices/servers but also maintenance and recurrent costs. Ease of maintenance, expandability to meet the future needs are factors also considered.

Security zones

We had previously divided GIAC Enterprises network into segments based roughly on access requirements. The segments also logically fall into dedicated subnets/ security zones based on their security needs. This seems a logical design practice that would also help in the maintenance of the network. Security zoning is achieved through sub-netting with security devices at the subnet entry points. In addition to the security advantage, subnets also provide a performance advantage by limiting the size of the broadcast domain [1]. The security zones defined are:

- External service subnet (ge_ext)

This publicly accessible subnet contains servers for web, mail and DNS services. Servers in this subnet provide services to the public and GE customers, suppliers and partners. They never initiate connections to the Internet but may connect to servers in the internal service network as part of their function.

- Internal service subnet (ge_int)

This network contains servers for web, mail and DNS services. Only hosts from the external service, admin and user networks may connect to these servers. There is no direct access to the internal service network from the Internet.

- Admin subnet (ge_adm)

System administrator workstations, database and central log servers are located in this subnet. Servers and security devices are managed through the administrator workstation.

- User subnet (ge_usr)

  This network has workstations and file/print servers for use by GE employees.

Traffic in these zones is controlled by the following network security devices:

- Border Router

  Connects GE network to the Internet. It provides routing and basic static packet filtering.

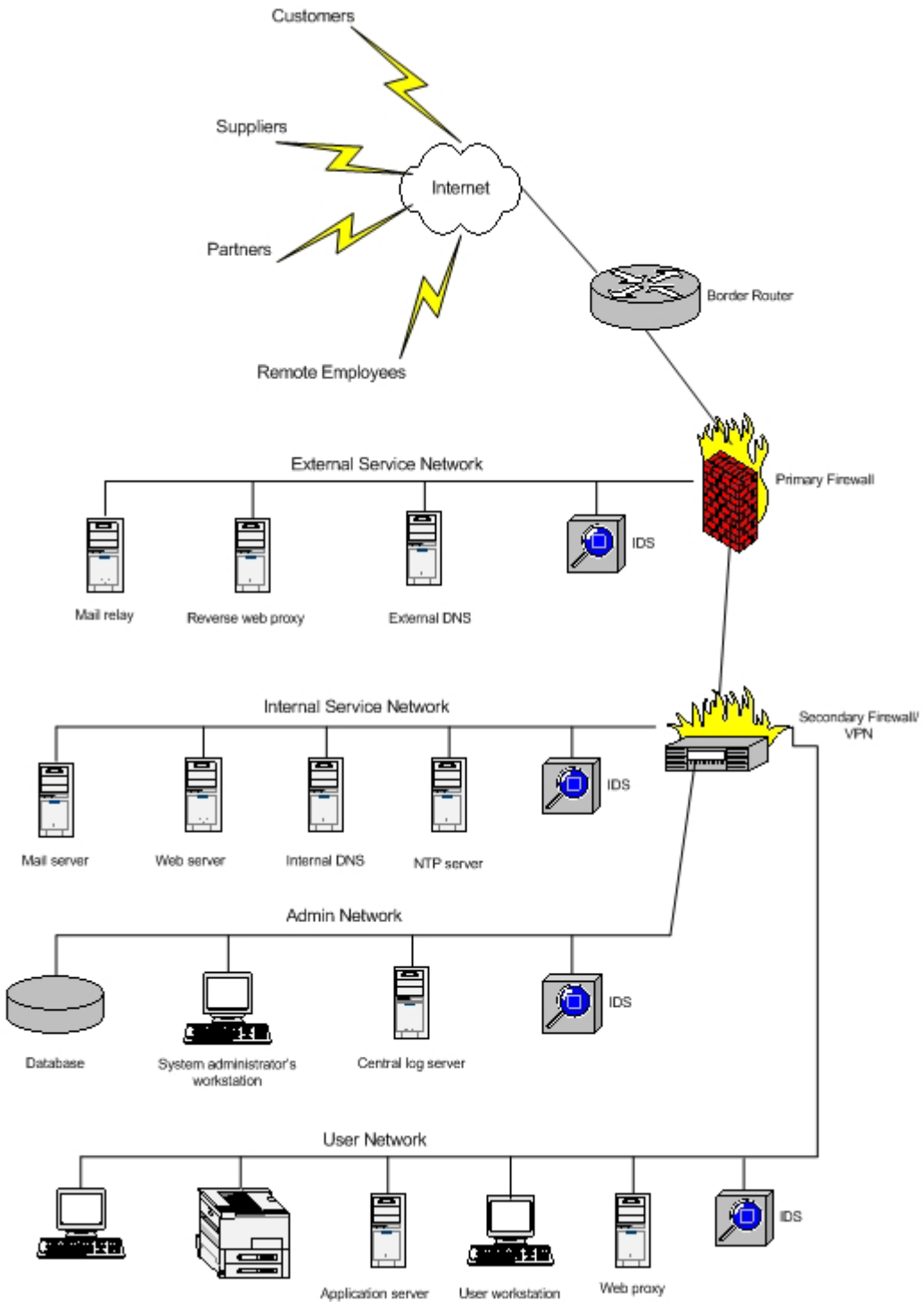
- Primary Firewall

  Located behind the router, it enforces most of the access controls in the subnets behind the firewall.

- Secondary Firewall/VPN

  Located behind the primary firewall, it augments and provides redundancy to the primary firewall. It also provides VPN support.

Network diagram of GIAC Enterprises network.



IP addressing scheme

The address space used for the router, firewall and external service network is the reserved space (100.1.1.0/24) chosen from the IANA reserved list [4] (http://www.iana.org/assignments/ipv4-address-space). Behind the firewall, we use

non-routable addresses (RFC1918 private subnet 192.168.x.x) for the internal service, general and user networks. The addresses used by the various components are shown in the following tables.

Network segments:

Segment	Address Space
Router	100.1.1.0/28
External subnet (ge_ext)	100.1.1.16/28
Internal subnet (ge_int)	192.168.1.0/24
Admin subnet (ge_adm)	192.168.2.0/24
User subnet (ge_usr)	192.168.3.0/24

Primary Firewall interfaces:

Interface	Connection	Address
eth0	Router	100.1.1.1/28
eth1	External network	100.1.1.17/28
eth2	Secondary FW/VPN	192.168.0.1/24

GE host (in External, Internal, User and Administrator network) addresses:

Component	Name	Address
Router	ge_router	100.1.1.2
Reverse web proxy	ge_web_ext	100.1.1.18
SMTP mail relay	ge_mail_ext	100.1.1.19
External DNS	ge_dns_ext	100.1.1.20
External IDS	ge_ids_ext	100.1.1.21
Web server	ge_web_int	192.168.1.2
Mail server	ge_mail_int	192.168.1.3
Internal DNS	ge_dns_int	192.168.1.4
NTP server	ge_ntp_int	192.168.1.5
Internal IDS	ge_ids_int	192.168.1.6
Central log server	ge_log_adm	192.168.2.2
System administrator's workstation	ge_SA_adm	192.168.2.3
Database server	ge_db_uadm	192.168.2.4
Admin IDS	ge_ids_adm	192.168.2.5
User workstation	ge_wks_usr	192.168.3.2

Web proxy server	ge_wbx_usr	192.168.3.3
User IDS	ge_ids_usr	192.168.3.4

NAT:

Interface	Name	External Address	Internal Address
Web proxy server	ge_wbx_usr	100.1.1.24	192.168.3.3
NTP server	ge_ntp_int	100.1.1.25	192.168.1.5

Defense in Depth

In designing the network for GE, we use the concept of defense in depth [1, 2]. Defense in depth focuses on a layered approach in design, which helps to ensure that there is no single point of security failure. Another general design guideline is that we use a mix of technologies and vendors [2]. We leverage on the strengths of different vendors/technologies to minimize against vulnerabilities in any one product.

As described above, GE network is divided into security zones with traffic in the zones controlled by router and firewalls. The IP addressing with non routable addresses for the internal networks also follows good practice. Here are more details of the security devices and features of the network design:

Border Router
The border router is at the outermost interface between GE network and the internet, and is GE network's first line of defense. Apart from routing function, the router efficiently does static packet filtering and blocks certain absolutes that could be used as probes, in spoofing, DoS or SMURF attacks. It does;

- Ingress filtering to block inbound traffic with source IP of internal network, loop back and private addresses, unallocated addresses (IANA RFC 1918), or with a destination of the internal broadcast address
- Egress filtering to allow outbound traffic with source IP of GE address space but block and log all other traffic
- Ingress/egress filtering to block in and out bound traffic for critical services
- Blocks traffic with specific options such as source routed packets, ICMP echo replies and unreachables

Primary Firewall
This device forms the second layer of defense in depth, and controls the traffic that enters through the router.

The packet filtering function of the router is limited, it cannot inspect packet payload or handle complex protocols such as FTP. These limitations may be exploited by maliciously crafted packets. The primary firewall is able to do stateful filtering. The packet state is captured in a table. Packets are matched against this connection table for earlier activity. If there is no match, the packets then pass through for normal rule processing. But stateful filtering is by no means foolproof.

Secondary Firewall/VPN
GE network has a second firewall, to add redundancy and avoid a single point of security failure (of having a single firewall) in the network design. It has Stateful Inspection and VPN functionality. The secondary firewall is a different type from the primary firewall to leverage on the strengths of different technologies. This is added security as there is less likelihood of different firewalls having the same vulnerability.

The VPN gateway provides remote employees with a secure connection to the GE internal subnets.

The firewall also provides network address translation (NAT). Hide NAT will be used for outbound sessions from the internal user and admin subnets to the internet.

Separation of services
Services such as web, email, database and DNS run on separate/individual servers. Hosting different services on different servers lessens the chance that multiple services will be affected when a server is compromised [1].

Some of these services are also separated into internal and external components, which make it possible to locate/implement them in the security zones we have defined.

Other services such as NTP, IDS, central logger, are also hosted on separate servers.

Proxies
Several dedicated proxies are used in the network: a reverse web proxy, a web proxy, and a SMTP relay. Unlike firewalls which work at the lower network layers, proxies are able to look at application layer data and screen based on data content. Proxies are positioned between the requester and the actual server, so there is no direct connection to the server. However, it must accept inbound connections through one or more listening ports. By leveraging on defense in depth and placing the proxy behind the router and firewalls, this risk can be mitigated.

Web proxies
Inbound connections to the GE web server on the internal network are directed to the reverse web proxy on the external network. Thus, hosts are prevented from connecting directly to the internal web server. The reverse proxy can also filter out malformed traffic and viruses.

Customers, suppliers, partners use HTTPS to connect securely to the reverse web proxy for transactions with GE. HTTP is used to access general information.

The web proxy on the user subnet services outbound connections. There are no direct connections to the Internet.

Email
The Sendmail server (mail proxy) on the external service subnet acts as a SMTP relay to the mail server which lies in the internal service subnet. There are no direct connections to the internal mail server. All inbound and outbound email pass through the sendmail relay. Sendmail is configured to forward mail to the internal mail server. Outbound mail headers are stripped of information about the internal mail server. A virus scanner scans the mail/attachments before passing to the internal mail server.

DNS
Split DNS is used to separate external and internal name resolution on the GE network. The external DNS is available to external Internet users, while the internal DNS server is used within GE. The external DNS resolves hostnames of GE public systems on external network. No information about GE internal network infrastructure is kept on the external DNS server, thus minimizing effect on internal resources if compromised. We also limit who can do recursive lookups on the external DNS server to reduce risk of DNS attacks. Zone transfers are limited to appropriate secondary DNS servers and logged. The internal DNS is configured to forward queries about external hosts to the external DNS server.

Database server
This is located in a secured internal admin subnet. The database server is only accessible from applications located on the GE web server.

IDS
Network based IDS (Intrusion Detection System) is used to augment security provided by the perimeter devices. They are placed on GE external/internal services, user and admin subnets. The IDS inspect packet content and are able to detect pattern signatures of known attacks, for example, DNS zone transfer requests from unauthorized hosts, Unicode attacks, buffer overflow attacks. When attack patterns are detected, the IDS sends reports/alerts to the system administrator.

Logging/ monitoring
A central log server is set up in the admin subnet. Security devices and servers are configured to send log files to the central log server (as well as maintaining a set locally). A NTP server is set up in the internal service subnet. It synchronizes with two different remote servers. The security devices and servers synchronize times with the NTP server. Synchronizing times across all the devices in the network will help ensure accuracy in timing of events in the log entries.

In the event that an individual server is compromised, having a central log server ensures that there is a reliable backup copy of the server logs for analysis. The central log repository also eases log analysis and management. Tools such as logrotate, logwatch, and swatch [12] are used to automate the management, analysis and monitoring of the logs. Periodic (weekly) human review of the logs is also scheduled. The logs are backuped daily. The log and NTP servers are hardened and dedicated to their respective functions.

Hardening servers and applications

Perimeter devices, servers and workstations/desktops are installed with latest OS (Operating System) version and patches. The OS are hardened, using some of the many tools/scripts available, e.g., at SANS, Titan and Bastille Linux websites [5-7]. As a result, only ports/services that are needed are opened. Anti-virus software is installed on the workstations and notebooks. Additionally, remote users have personal firewalls such as Zone Alarm [8] installed on their computers. The hosts are checked using auditing tools from Center for Internet Security (CIS) [9]. For the servers, a baseline audit is performed.

In keeping with good security management, the servers and devices are kept updated with the latest patches. The virus signatures are also kept up to date. Audits and vulnerability scans are conducted regularly using Nessus [10] and other tools.

Ensure applications (Sendmail, Bind, PostgreSQL, etc.) are also secured. For example, some tips for Sendmail [11]:

- Disallow VRFY and EXPN options
- Hide version numbers
- Limit message size
- Use smrsh as MDA

Many guides and scripts are available at the SANS, CIS and other websites for hardening the applications used.

Security Maintenance

Once the security infrastructure is set up, it should be sufficiently maintained. To summarize the requirements:

- Patching
  Maintain current OS releases, up to date patches
- Backups
  Regular backups of system files, system/monitoring logs, rule bases
- Log analysis/review
  Besides the automated analysis tools, periodic human review is needed
- Virus and IDS signatures
  Keep them up to date
- Regular audits of systems and processes

- Proper change management/validation of router/firewall rules

1.4 Component Choices

Cost, maintainability and scalability of the security infrastructure are important considerations in our network design/ component choices. Open Source technologies are used a lot. A number of low cost Intel based servers running Red Hat Linux are deployed in firewall, mail relay and proxies. Licensing and hardware costs are relatively low, and hardware may be upgraded at low cost. Many good Open Source software/tools are available. Expertise in Linux is also becoming more abundant as it becomes more widely adopted.

As with all systems, requirements are likely to change over time. Because of the layered design, individual components may be changed to some extent without impacting the overall design. The infrastructure was targeted at a small-medium sized company. As the company grows, performance of the devices needs to be reviewed and upgraded if necessary.

Security components:

- **Border router**

A CISCO 1721 router at IOS 12.2 with latest patches is used [13]. This is an entry level router for small/medium enterprises and should be sufficient to meet the network traffic needs of GIAC Enterprises (with some room to grow). The serial interface connects to the leased line to the internet while the ethernet port connects to GE network.

- **Primary firewall**

Iptables version 1.2.8 running on a DELL PowerEdge 2650 server installed with Red Hat 9 Linux which is Bastille Linux 2.1.1 hardened. This is a low cost (hardware/ software) solution that meets the technical requirements for a stateful firewall which is highly flexible and maintainable [14, 15]. Iptables also has robust logging ability.

- **Secondary firewall / VPN**

Our choice is Sun iForce VPN/Firewall appliance, a low-cost, high-performance (3+ Gbps FW-1, 1+ Gbps VPN-1), easily deployable solution for mid-sized businesses [16]. This appliance uses Check Point Express which includes VPN-1 Express gateway, FireWall-1, SmartDefense and SmartCenter [17]. SmartDashboard, is the friendly GUI tool which administrators can use to manage the firewall rules, VPN, NAT, etc. SmartDefense leverages on Stateful Inspection and Artificial Intelligence technologies for application level protection.

Off-site/remote GE employees install VPN-1 SecureClient on their notebooks/ workstations. VPN-1 SecureClient encrypts and authenticates data to allow remote employees to securely connect to GE network. It comes with a personal firewall and advanced management features.

- **Reverse web proxy**

Sun Java System Web Proxy Server (formerly Sun ONE web proxy) [18, 19] is used. It comes preinstalled on a hardened and patched Sun Fire V120 server running Solaris 9 [20].

- **Web proxy**

Squid Web Proxy version 2.5 release 4 [21, 22] is used. Squid is an open-source, full-featured web proxy available for unix systems. We install it on a hardened and patched DELL PowerEdge 2650 server running Red Hat 9 Linux.

- **IDS**

We use Snort 2.0.4, an open-source network intrusion detection system [23]. It is installed on hardened and patched DELL PowerEdge 1750 servers running Red Hat Linux 9.

- **Virus scanner**

Trend Micro InterScan VirusWall version 3.8 is used [24]. It is installed on the web server, mail proxies, mail relay and mail server.

Other components:

- **Web server**

Sun Java System Web Proxy Server which is preinstalled on a hardened and patched Sun Fire V120 server running Solaris 9.

- **External SMTP relay**

Sendmail version 8.12.3 [25] is used. It is installed on a hardened and patched Sun Fire V100 server running Solaris 9.

- **Database**

We use postgreSQL version 7.4 [26] installed on a hardened and patched DELL PowerEdge 2650 server running Red Hat 9 Linux.

- **NTP server**

We use ntpd which comes already installed on a hardened and patched DELL PowerEdge 1750 server running Red Hat 9 Linux.

Assignment 2 – Security Policy and Tutorial

In this assignment, we develop the security policy for some of the components in the security network design described in Assignment 1. These are the border router, primary firewall and VPN. The security policy takes into account the business and security requirements defined in the last assignment.

This is followed by a tutorial on how to implement the primary firewall policy.

2.1 Border Router

The CISCO 1721 router running IOS 12.2 performs static packet filtering in addition to routing. Static filters are good (efficient) at blocking absolutes through ingress and egress filtering. Extended access lists, which have the ability to compare more IP packet attributes, is used. The access list number must be in the range 100-199 for these lists.

Hardening the Router
The router must first of all be secured. To secure the router, some rules/configurations are required:

- Restrict access/logins to the router
  Access is allowed only from system administrator's workstation and console.

```
Access-list 10 permit tcp 192.168.2.3 0.0.0.255
line con 0
login local
line vty 0 4
login
transport input ssh
username <user> password <password>
```

- Encrypt passwords

```
service password-encryption
enable secret 5 <password>
```

- Disable SNMP, finger, TCP and UDP small services

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

- Disable CDP (Cisco Discovery Protocol), configuration autoloading, bootp, http server services

```
no cdp run
no service config
no ip bootp server
no ip http server
```

- Disable source routing

```
no ip source route
```

- Prevent layer 3 to layer 2 broadcast mapping and smurf amplification

```
no ip directed-broadcast
```

- Disable ICMP unreachable messages, redirects, mask replies

```
no ip unreachables
no ip redirects
no ip mask-reply
```

- Add warning banner

```
banner /
WARNING: Authorised Access Only
/
```

- Log significant events and console messages to secure remote server

```
logging <IP address of syslog server>
```

Ingress Filtering

In ingress filtering, certain types of inbound traffic are blocked:

- source IP of private addresses (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16)
- source IP of loopback address (127.0.0.0)
- source IP of internal network (100.1.1.0/24)
- destination IP of internal broadcast address (100.1.1.255)
- unallocated source IP. These reserved addresses may be obtained from the Internet Assigned Numbers Authority (IANA) list of IP v4 address space [4] at http://www.iana.org/assignments/ipv4-address-space.

Set up access lists:

```
Interface serial 0
  access-group 101 in
```

The Ingress filtering rules are:

```
access-list 101 deny ip 10.0.0.0 0.255.255.255 any log-input
access-list 101 deny ip 172.16.0.0 0.15.255.255 any log-input
access-list 101 deny ip 192.168.0.0 0.0.255.255 any log-input
access-list 101 deny ip 127.0.0.0 0.255.255.255.255 any log-input
access-list 101 deny ip 100.1.1.0 0.0.0.255 any log-input
access-list 101 deny ip 100.1.1.255 0.0.0.255 any log-input
access-list 101 deny ip 0.0.0.0 0.255.255.255 any log-input
access-list 101 deny ip 1.0.0.0 0.255.255.255 any log-input
```

```
access-list 101 deny ip 2.0.0.0 0.255.255.255 any log-input
access-list 101 deny ip 5.0.0.0 0.255.255.255 any log-input
….
access-list 101 deny ip 253.0.0.0 0.255.255.255 any log-input
access-list 101 deny ip 254.0.0.0 0.255.255.255 any log-input
access-list 101 deny ip 255.0.0.0 0.255.255.255 any log-input
```

Unused legal addresses are logged as this may give warning of a DDoS spoofing attack. The list is processed top down.

Critical services are blocked at outside of network perimeter inbound (to keep attackers out). The services are ssh, tftp, rpcbind, exec, biff, login, who, shell, syslog, nfs, X11 windows. NetBios traffic is also blocked.

```
access-list 101 deny tcp any any range 135 139
access-list 101 deny udp any any range 135 139
access-list 101 deny tcp any any 445
access-list 101 deny tcp any any eq 22 log-input
access-list 101 deny udp any any 69 log-input
access-list 101 deny tcp any any 111 log-input
access-list 101 deny udp any any 111 log-input
access-list 101 deny tcp any any range 512 514 log-input
access-list 101 deny udp any any range 512 514 log-input
access-list 101 deny tcp any any 2049 log-input
access-list 101 deny tcp any any range 6000 6255 log-input
acess-list 101 permit any any
```

As this router is the first line of defense, other addresses are allowed to the next level. There is an implicit deny statement, hence the last (permit) statement is necessary.

Egress Filtering

Egress filtering blocks outbound traffic.
Set up access lists:

```
Interface eth0
  access-group 102 in
```

The critical services are blocked outbound (to stop attacks outside if compromised):

```
access-list 102 deny tcp any any eq 22 log-input
access-list 102 deny udp any any 69 log-input
access-list 102 deny tcp any any 111 log-input
access-list 102 deny udp any any 111 log-input
access-list 102 deny tcp any any range 512 514 log-input
access-list 102 deny udp any any range 512 514 log-input
access-list 102 deny tcp any any 2049 log-input
access-list 102 deny tcp any any range 6000 6255 log-input
access-list 102 deny icmp any any echo-reply
access-list 102 deny icmp any any host-unreacheable
access-list 102 deny icmp any any time exceeded
```

The last rules block outbound ICMP echo-reply and unreachable packets, to stop probes from mapping our network.

Block other outbound traffic except for traffic with source IP of GIAC address. This prevents spoofed packets coming from GE network.

```
access-list 102 permit ip 100.1.1.0 0.0.0.255 any log-input
access-list 102 deny any log-input
```

2.2 Primary Firewall

The primary firewall used is iptables version 1.2.8 which is installed on a DELL PE2650 running Red Hat 9 Linux. This box is hardened with Bastille Linux 2.1.1.

Besides hardening, other configurations/setups are needed on the firewall box:

- system logs are copied to central log server on admin network
- time synchronization with NTP server on internal service network
- regular schedule of up2date patching
- backups of system files before and after system changes
- change management of firewall rules

The table summarizes the network traffic through the primary firewall.

	Incoming zone	Outgoing zone	Source	Destination	Service/ Port
1	Internet	External service	Any	ge_web_ext	HTTP TCP/80 HTTPS TCP/443
2	Internet	External service	Any	ge_mail_ext	SMTP TCP/25
3	Internet	External service	Any	ge_dns_ext	Bind TCP/53, UDP/53
4	External service	Internet	ge_mail_ext	any	SMTP TCP/25
5	External service	Internet	ge_dns_ext	any	Bind TCP/53, UDP/53
6	External service	Internal service	ge_mail_ext	ge_mail_int	SMTP TCP/25
7	External service	Internal service	ge_web_ext	ge_web_int	HTTP TCP/80 HTTPS TCP/443

8	Internal service	External service	ge_mail_int	ge_mail_ext	SMTP TCP/25
9	Internal service	External service	ge_dns_int	ge_dns_ext	Bind TCP/53, UDP/53
10	User	Internet	ge_wbx_usr	any	HTTP TCP/80 HTTPS TCP/443 FTP TCP/20,21
11	Admin	External service, Security devices	ge_SA_adm	ge_sec ge_ext	SSH TCP/22
12	External service, Security devices	Internal service	ge_sec ge_ext	ge_ntp_int	NTP UDP/123
13	External service, Security devices	Admin	ge_sec ge_ext	ge_log_adm	Syslog UDP/514

The iptables firewall rules to implement the above policy is described in the tutorial in section 2.4. The tutorial also covers a general overview of iptables, format and syntax.

2.3 VPN

A VPN (Virtual Private Network) system is needed so that remote employees (mobile sales and teleworkers) can connect securely to GE network. Our design uses the VPN integrated in the Checkpoint Firewall-1/VPN-1 secondary firewall. VPN technology uses IPSec (IP Security) protocol, in which security features are added to IP. IPSec allows a secure channel to be created between two communicating IPSec devices, which may be hosts or security gateways. The IPSec protocol is blocked by default by most firewalls and some ports must be unblocked if it is to be used.

We are implementing a remote access VPN. The IPSec devices are the VPN security gateway, which is the secondary firewall, and VPN hosts or clients on the remote employee's computer/notebook. The devices negotiate a Security Association (SA) that defines the security measures to be applied. Each device maintains databases of its security policies in the Security Policy Database (SPD) and parameters for active SAs in the Security Association Database (SAD).

In the following, we define the policies and databases for the GE VPN.

IKE Policy
IKE (internet key exchange) protocol is used in SA negotiation to manage key exchange between the IPSec devices. IKE uses UDP port 500. It is based on the ISAKMP (Internet Security Association and Key Management Protocol) framework. A secure channel or ISAKMP SA is first established using a pre-shared secret key xxxxxx. The other agreed upon parameters:

Parameter	Value
Pre-shared secret	xxxxxx
Mode	Main
Encryption Algorithm	3DES
Hashing Algorithm	MD5
Duration	720 minutes

IPSec Policy
After the ISAKMP SA is established, the IPSec devices negotiate and create the IPSec SA, after which secure data exchange may occur. Parameters used to negotiate the IPSec SA:

Parameter	Value
Protocol	ESP
Mode	Tunnel
Encryption Algorithm	3DES 168-bit
Hashing Algorithm	HMAC MD5
Duration	30 minutes

Tunnel mode is chosen because the VPN is of type gateway to host. ESP (Encapsulating Security Payload) is chosen to encrypt entire packets.

Virtual addresses from the subnet 192.168.4.0/24 will be created to represent the VPN hosts.

2.4 Tutorial for Primary Firewall

Setting up Iptables
The iptables control script is installed at /etc/init.d/iptables. We set it for auto startup on reboot by putting a link in /etc/rc3.d:

```
lrwxrwxrwx    1 root     root         18 Jun 13 14:55 S08iptables ->
../init.d/iptables
```

Options for the script:

```
/etc/init.d/iptables start|stop|restart|condrestart||status|panic|save
```

The default configuration file /etc/sysconfig/iptables-config is used.

Setting up Central Logging
The iptables messages are logged in /var/log/messages. We configure system logging for a copy of the messages and syslogs to be sent to the central log server on the admin subnet. This is done by having additional entries in the syslog configuration file /etc/syslog.conf:

```
*.info;mail.none;authpriv.none;cron.none        @192.168.2.2
syslog.*                                        @192.168.2.2
```

On the remote log server, set the –r option for syslogd in /etc/sysconfig/syslog to allow logging from remote systems:

```
SYSLOGD_OPTIONS="-r -m 0"
```

Setting up Time Synchronization
Synchronize time with the NTP server on the internal service subnet. To do this, change the server entry in the configuration file /etc/ntp.conf to point to the NTP server:

```
server  192.168.1.5
```

Setting Change Management
Institute change management of firewall rules, such as keeping a change log and going through proper authorization or approval process/procedure for rule changes.

Iptables Overview
Before defining the firewall rules for GE, we give an overview and general description of iptables format and syntax, with examples. Relevant parts are summarized from reference [3]. For further details, one may try the many iptables references on the internet or check the iptables man page for a detailed summary.

Iptables are used to create the packet filtering rules composing the firewall policy. It is implemented as a set of program modules, which are loaded dynamically and automatically upon first use. The iptables rules are organized around three rule tables (filter, nat, mangle) which have different packet processing functionality. The rules are stored in kernel tables, in the order in which they are defined, in built-in constructs called chains which target different types of packets:

- filter
  This default table contains these built-in chains:
  - INPUT - specifies changes to incoming packets
  - OUTPUT - specifies changes to local outgoing packets
  - FORWARD - specifies changes to non-local outgoing packets

  User-defined chains are also possible. The possible actions or target dispositions are ACCEPT or DROP. The IP header field matches operations for protocol, source and destination address, input and output interfaces, and fragment handling.

- nat
  The second type of table contains rules for source and destination address and port translation. The built-in chains are:
  - PREROUTING – specifies destination changes to incoming packets before routing (DNAT/REDIRECT)
  - OUTPUT - specifies destination changes to local outgoing packets before routing (DNAT/REDIRECT)
  - POSROUTING - specifies source changes to outgoing packets after routing (SNAT/MASQUERADE)

  The following forms of NAT are supported:
  - SNAT - source NAT
  - DNAT – destination NAT
  - MASQUERADE – special source NAT for connections assigned temporary/ dynamic IP
  - REDIRECT - special destination NAT that redirects packet to local host, regardless of IP header destination address

- mangle
  The third table type contains rules for setting specialized packet-routing flags which are examined later by rules in the filter table. The built-in chains are:
  - PREROUTING - specifies changes to incoming packets before routing
  - OUTPUT - specifies changes to local outgoing packets

We will be using the nat and filter tables to define our firewall rules. The filter table has two kinds of extensions, target and match. Examples of some of the match extensions that we use are shown in the iptables syntax overview covered next.

Syntax

iptables command

For a useful command summary, try *iptables –h* or *iptables --help*. The tables below list some of the many options available that allow very specific packets to be identified. The packets may be matched by combinations of packet message protocol type, source/destination IP addresses, source/destination service ports, network interface, etc. Some notation:

| alternative syntax
< > user specified string, numeric
[ ] optional parameter

- Filter table operations on entire chains

Command	Description
-N | --new <chain>	Create new user-defined chain
-F | --flush [<chain>]	flush (all) chain(s)
-X | --delete-chain [<chain>]	Delete (all) chain(s)
-Z | --zero	reset counters on (all) chain(s)
-h | <command> -h	list iptables (all) command(s)
-P | --policy <chain> <policy>	Define ACCEPT or DROP policy on built-in chain
-L | --list [<chain>]	list rules in (all) chain(s)
-L –n | --numeric	list IP addresses and port numbers numerically
-L -v | --verbose	list additional information
-L -x | --exact	list exact counter values
-L --line-numbers	list rule's position within chain

These basic filter table operations for chain initialization and information gathering are quite self-explanatory. We use a few (as you'll see later).

Examples:

```
# rule applies to the default filter table, unless specified otherwise
with –t option
iptables --flush
iptables -t nat --flush
iptables -t mangle --flush

# default drop policy is defined
iptables --policy INPUT   DROP
iptables --policy OUTPUT  DROP
```

```
iptables --policy FORWARD DROP
```

- Filter table operations on a rule

Command	Description
-A | --append <chain>	Append rule to chain end
-I | --insert <chain>	Insert rule to chain head
-D | --delete <chain> <rule number>	Delete rule at position rule number within chain

Examples:

```
# Remove any pre-existing user-defined chains
iptables --delete-chain
iptables -t nat --delete-chain
iptables -t mangle --delete-chain
```

- Filter table basic match operations

Command	Description
-I | --in-interface [!] [<interface>]	rule to be applied to incoming packets INPUT, FORWARD or user-defined chains' (all) interfaces
-o | --out-interface [!] [<interface>]	rule to be applied to outgoing packets OUTPUT, FORWARD or user-defined chains' (all) interfaces
-p | --protocol [!] [<protocol>]	IP protocol that rule applies to. Built-in protocols are tcp, udp, icmp and all.
-s | --source | --src [!] <address>[/<mask>]	specifies host or network source address in IP header
-d | --destination | - -dst [!] <address>[/<mask>]	specifies host or network destination address in IP header
-j | --jump <target>	Specifies target if packet matches rule. Default targets are ACCEPT, DROP, or user-defined.

Examples:

```
# Drop all incoming tcp packets with source IP address 137.120.1.1
coming from interface eth0
iptables -A INPUT –i eth0 -p tcp -s 137.120.1.1 -j DROP
```

- Filter table TCP match operations

-p tcp	Description
--source-port | --sport [!] <port>[:<port>]	Specifies source ports
--destination-port | --dport [!] <port>[:<port>]	Specifies destination ports
--tcp-flags [!] <mask>[,<mask>] <set>[,<set>]	Tests bits in mask list for matched set bits
[!] –syn	SYN flag must be set as initial connection request
--tcp-option [!] <number>	Maximum packet size that the sending host is willing to accept

Examples:

```
# Drop all incoming tcp packets with source IP address 137.120.1.1 and
port 25 coming from eth0 interface
iptables –A INPUT –i eth0 -p tcp –s 137.120.1.1 --sport 25 -j DROP

# Drop incoming tcp packets that have SYN and FIN both set
iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
```

- Filter table ICMP match operation

-p icmp	Description
--icmp-type [!] <type>	Specifies ICMP type name or number

The major supported ICMP type names/numbers:

- echo-reply (0)
- destination-unreachable (3)
  network-unreachable, host-unreachable, protocol-unreachable, port-unreachable, fragmentation-needed, network-unknown, host-unknown, network-prohibited, host-prohibited
- source-quench (4)
- redirect (5)
- echo-request (8)
- time-exceeded (10)
- parameter-problem (11)

Examples:

```
# Drop echo requests trying to cross firewall through interface eth0
iptables -A FORWARD –i eth0 -p icmp ––icmp-type 8 -s 0/0 –d 0/0 -j
DROP
```

- filter table target extensions

-j LOG	Description
--log-level <syslog level>	The log levels are emerg(0), alert(1), crit(2), err(3), warn(4), notice(5), info(6), debug(7).
--log-prefix <"descriptive string">	String will be printed at start of log message for the rule
--log-ip-options	Include IP header options in log output
--log-tcp-sequence	Include TCP packet sequence number in log output
--log-tcp- options	Include TCP header options in log output

Examples:

```
# Log incoming tcp packets that have SYN and FIN both set
iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j LOG
```

- Multiport filter table match extensions

-m | --match multiport	Description
--source-port <port>[,<port>]	Specifies source ports
--destination-port <port>[,<port>]	Specifies destination ports
--port <port>[,<port>]	Source and destination ports equal and match a port in list

This table extension allows multiple ports to be specified in an iptables command. Since the performance of the firewall depends on the number of rules, this feature enhances the efficiency of the firewall.

Examples:

```
# Accept tcp chains trying to cross firewall to destination IP address
137.120.1.1 and destination ports 80 and 443.
iptables -A FORWARD -p tcp –d 137.120.1.1 –m multiport –destination-
port 80,443 -j ACCEPT
```

- State filter table match extensions

-m | --match state	Description
--state < state >[,< state >]	Connection state is in the list. Valid state values are NEW, ESTABLISHED, RELATED, or INVALID

This table extension allows the firewall to do stateful packet filtering.

Examples:

```
# The first rule allows the mail server at 137.120.1.1 to establish an
SMTP connection with any internet host.  The second rule matches all
ESTABLISHED and RELATED traffic associated with the state table entry
set up in the first rule.
iptables -A FORWARD -m state --state NEW -p tcp -d 137.120.1.1 -d 0/0
--dport 25 -j ACCEPT
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -p tcp -j
ACCEPT
```

Processing of Packets

Incoming packet header fields are compared against each rule in the interface's INPUT chain until a match is found. Locally generated outgoing packet headers are compared against each rule in the interface's OUTPUT chain until a match is found. Non local outgoing packet headers are compared against each rule in the interface's FORWARD chain until a match is found. When a match is found, the comparison stops and the rule's packet disposition is applied: ACCEPT, DROP or REJECT. If there is no match, the default policy is applied.

Rules are inspected top to bottom, the first rule that matches applies. Thus, the order in which the rules are defined is very important. As rules of thumb, for accuracy and efficiency, rules on a chain are ordered general to specific and rules for heavily used services are placed as early as possible. The multiport extension module is used to specify lists of ports, where possible, to reduce the number of rules traversed. Stateful filtering, implemented through the state module (NEW, ESTABLISHED, RELATED states), allows more effective control for complex protocols. Iptables also has a strong logging feature, log prefixes may be used to easily identify the packets logged.

GE Primary Firewall Rules

Finally, we define the iptables rules for the primary firewall, describing as much as possible the purpose and ordering of the rules, tips, tricks and potential problems.

We define a deny everything by default policy for the primary firewall. Services are individually enabled as exceptions to the policy.

A shell script is used to set up the iptables rules. Using a script makes it easier to track and maintain the rules. Symbolic constants are defined for recurring names, e.g., hostnames, service ports, interfaces, network addresses, for clarity and ease of rule maintenance.

Iptables Initialization

- Load modules, define symbolic constants

```
#!/bin/bash

# load connection tracking modules
modprobe ip_conntrack_*
modprobe ip_state

# interfaces
IF_ROUTER="eth0"                    # Router-connected interface
IF_EXTERNAL="eth1"                  # External network interface
IF_SECFW="eth2"                     # Secondary firewall interface
IF_LOOPBACK="lo"                    # loopback interface

# router and firewall addresses
IP_ROUTER="100.1.1.2"               # router
IP_FW="100.1.1.1"                   # firewall at eth0

# external service subnet addresses
IP_EXT="100.1.1.16/28"              # external service address space
IP_WEB_EXT="100.1.1.18"             # reverse web proxy
IP_MAIL_EXT="100.1.1.19"            # SMTP mail relay
IP_DNS_EXT="100.1.1.20"             # external DNS

# internal service subnet addresses
IP_INT="192.168.1.0/24"             # internal service address space
IP_WEB_INT="192.168.1.2"            # internal web server
IP_MAIL_INT="192.168.1.3"           # internal mail server
IP_DNS_INT="192.168.1.4"            # internal DNS
IP_NTP_INT="192.168.1.5"            # NTP server

# admin subnet addresses
IP_ADM="192.168.2.0/24"             # admin subnet address space
IP_LOG_ADM="192.168.2.2"            # central log server
IP_SA_ADM="192.168.2.3"             # system admin workstation
IP_DB_ADM="192.168.2.4"             # database server

# miscellaneous addresses
IP_WBX_USR="192.168.3.3"            # web proxy server
IP_WBX_GE="100.1.1.22"              # external IP of web proxy server
IP_NTP_GE="100.1.1.23"              # external IP of ntp server
IP_NTP1="aa.bb.cc.dd"               # stratum 2 NTP server 1
IP_NTP2="ee.ff.gg.hh"               # stratum 2 NTP server 2
IP_ANY="0.0.0.0/0"                  # any IP address range
```

```
LOOPBACK="127.0.0.0/8"                  # reserved loopback address range

# ports
UNPRIVPORTs="1024:65535"                # unprivileged port range
HTTP_PORT="80"                          # HTTP
HTTPS_PORT="443"                        # HTTPS
DNS_PORT="53"                           # DNS
SMTP_PORT="25"                          # SMTP
SSH_PORT="22"                            # SSH
FTP_PORT="21"                            # FTP
NTP_PORT="123"                           # NTP
SLOG_PORT="514"                          # SYSLOG
```

- Initialize kernel parameters

```
# Enable broadcast echo Protection
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

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

# Disable Source Routed Packets
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
    echo 0 > $f
done
```

- Initialize the firewall, define default policy

Note, the default drop policy rule is not position dependent, as they are not rules per se [3]. They are applied after a packet has been compared to each rule without a match. But the default policy is defined at the beginning of the script for the reason that if they were defined at the end of the script, and if the script contained an error causing it to exit prematurely, a default accept-everything policy would be in effect.

```
# Remove any existing rules from all chains
iptables --flush
iptables -t nat --flush
iptables -t mangle --flush

# Set the default policy to drop
iptables --policy INPUT   DROP
iptables --policy OUTPUT  DROP
iptables --policy FORWARD DROP

iptables -t nat --policy PREROUTING  DROP
iptables -t nat --policy OUTPUT DROP
iptables -t nat --policy POSTROUTING DROP
iptables -t mangle --policy PREROUTING DROP
iptables -t mangle --policy OUTPUT DROP
```

```
# Allow traffic on the loopback interface
iptables -A INPUT  -i $IF_LOOPBACK -j ACCEPT
iptables -A OUTPUT -o $IF_LOOPBACK -j ACCEPT
```

- By-pass rule checking for prior connections

```
iptables –A INPUT –m state –state ESTABLISHED,RELATED –j ACCEPT
iptables –A OUTPUT –m state –state ESTABLISHED,RELATED –j ACCEPT
iptables –A FORWARD –m state –state ESTABLISHED,RELATED –j ACCEPT
```

- For efficiency, we order the rules from general to specific. Here are some general rules to block TCP stealth scans / probes that may be linked to spoofing activities. These scans are logged.

```
# TCP Stealth Scans
# All bits cleared
iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
iptables -A INPUT -p tcp --tcp-flags ALL NONE -j LOG \
--log-prefix ”iptables TCP flags – ALL NONE-”

# SYN and FIN both set
iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j LOG \
--log-prefix ”iptables TCP flags – SYN,FIN-”

# SYN and RST both set
iptables -A INPUT -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
iptables -A INPUT -p tcp --tcp-flags SYN,RST SYN,RST -j LOG \
--log-prefix ”iptables TCP flags – SYN,RST-”

# FIN and RST both set
iptables -A INPUT -p tcp --tcp-flags FIN,RST FIN,RST -j DROP
iptables -A INPUT -p tcp --tcp-flags FIN,RST FIN,RST -j DROP \
--log-prefix ”iptables TCP flags - FIN,RST-”

# FIN bit set, no ACK
iptables -A INPUT -p tcp --tcp-flags ACK,FIN FIN -j DROP
iptables -A INPUT -p tcp --tcp-flags ACK,FIN FIN -j DROP \
--log-prefix ”iptables TCP flags – FIN no ACK-”

# PSH bit set, no ACK
iptables -A INPUT -p tcp --tcp-flags ACK,PSH PSH -j DROP
iptables -A INPUT -p tcp --tcp-flags ACK,PSH PSH -j DROP \
--log-prefix ”iptables TCP flags – PSH no ACK-”

# URG bit set, no ACK
iptables -A INPUT -p tcp --tcp-flags ACK,URG URG -j DROP
iptables -A INPUT -p tcp --tcp-flags ACK,URG URG -j DROP \
--log-prefix ”iptables TCP flags – URG no ACK-”
```

Enabling required services

Since rules are executed on first match, we should put the rules for heavily used services as early as possible. We expect the most heavy traffic to come from web HTTP, HTTPS services. However, to maintain the logical flow, we do not show the ordering here.

To reduce the number of rules and the time taken to traverse the rule set, the multiport option was used to specify port lists where possible.

- Inbound from Internet

Internet to servers in external service network

```
# HTTP, HTTPS
iptables -A FORWARD –m state –-state NEW -p tcp –i $IF_ROUTER –-sport
$UNPRIVPORTS –o $IF_EXTERNAL -d $IP_WEB_EXT –m multiport
–-destination-port $HTTP_PORT,$HTTPS_PORT -j ACCEPT

# DNS
iptables -A FORWARD –m state –state NEW -p udp –i $IF_ROUTER –-sport
$UNPRIVPORTS –o $IF_EXTERNAL -d $IP_DNS_EXT –-dport $DNS_PORT -j
ACCEPT
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_ROUTER –-sport
$UNPRIVPORTS –o $IF_EXTERNAL  -d $IP_DNS_EXT –-dport $DNS_PORT -j
ACCEPT

# SMTP
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_ROUTER –-sport
$UNPRIVPORTS –o $IF_EXTERNAL -d $IP_MAIL_EXT –-dport $SMTP_PORT -j
ACCEPT
```

- Outbound to Internet

External services to Internet

```
# DNS
iptables -A FORWARD –m state –state NEW -p udp –i $IF_EXTERNAL –s
$IP_DNS_EXT –-sport $UNPRIVPORTS –o $IF_ROUTER –-dport $DNS_PORT -j
ACCEPT
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_EXTERNAL –s
$IP_DNS_EXT –-sport $UNPRIVPORTS –o $IF_ROUTER –-dport $DNS_PORT -j
ACCEPT

# SMTP
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_EXTERNAL –s
$IP_SMTP_EXT –-sport $UNPRIVPORTS –o $IF_ROUTER –-dport $SMTP_PORT -j
ACCEPT
```

Web proxy to internet

```
# HTTP, HTTPS, FTP
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_SECFW –s
$IP_WBX_USR --sport $UNPRIVPORTS –o $IF_ROUTER –m multiport
--destination-port $HTTP_PORT,$HTTPS_PORT,$FTP_PORT -j ACCEPT
```

- External services to internal services

```
# HTTP, HTTPS
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_EXTERNAL –s
$IP_WEB_EXT --sport $UNPRIVPORTS –o $IF_SECFW -d $IP_WEB_INT –m
multiport --destination-port $HTTP_PORT,$HTTPS_PORT -j ACCEPT


# SMTP
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_EXTERNAL –s
$IP_SMTP_EXT --sport $UNPRIVPORTS –o $IF_SECFW -d $IP_SMTP_INT --dport
$SMTP_PORT -j ACCEPT
```

- Internal services to external services

```
# DNS
iptables -A FORWARD –m state –state NEW -p udp –i $IF_SECFW –s
$IP_DNS_INT --sport $DNS_PORT –o $IF_EXTERNAL -d $IP_DNS_EXT --dport
$DNS_PORT -j ACCEPT

# SMTP
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_SECFW –s
$IP_SMTP_INT --sport $UNPRIVPORTS –o $IF_EXTERNAL -d $IP_SMTP_EXT
--dport $SMTP_PORT -j ACCEPT
```

- Firewall to external DNS

```
iptables -A OUTPUT –m state –state NEW -p udp –o $IF_EXTERNAL –d
$IP_DNS_EXT --dport $DNS_PORT -j ACCEPT
```

- System administration of external servers and security devices

SSH

```
# external service
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_SECFW –s
$IP_SA_ADM –o $IF_EXTERNAL –d $IP_EXT --dport $SSH_PORT -j ACCEPT

# router
iptables -A FORWARD –m state –state NEW -p tcp –i $IF_SECFW –s
$IP_SA_ADM –o $IF_ROUTER –d $IP_ROUTER --dport $SSH_PORT -j ACCEPT

# firewall
```

```
iptables -A INPUT –m state –state NEW -p tcp –i $IF_SECFW –s
$IP_SA_ADM --dport $SSH_PORT -j ACCEPT
```

PING ICMP

```
# external service
iptables -A FORWARD -p icmp –icmp-type echo-request –i $IF_SECFW –s
$IP_SA_ADM -o $IF_EXTERNAL –d $IP_EXT -j ACCEPT
iptables -A FORWARD -p icmp –icmp-type echo-reply –i $IF_EXTERNAL –s
$IP_EXT –o $IF_SECFW –d $IP_SA_ADM -j ACCEPT

# firewall
iptables -A INPUT -p icmp –icmp-type echo-request –i $IF_SECFW –s
$IP_SA_ADM -j ACCEPT
iptables -A OUTPUT -p icmp –icmp-type echo-reply –o $IF_SECFW –d
$IP_SA_ADM -j ACCEPT

# router
iptables -A FORWARD -p icmp –icmp-type echo-request –i $IF_SECFW –s
$IP_SA_ADM -o $IF_ROUTER –d $IP_ROUTER -j ACCEPT
iptables -A FORWARD -p icmp –icmp-type echo-reply –i $IF_ROUTER –s
$IP_ROUTER –o $IF_SECFW –d $IP_SA_ADM -j ACCEPT
```

- Central logging for external service servers/security devices

```
# external service
iptables -A FORWARD –m state –state NEW -p udp –i $IF_EXTERNAL –s
$IP_EXT –o $IF_SECFW –d $IP_LOG_ADM --dport $SLOG_PORT -j ACCEPT

# router
iptables -A FORWARD –m state –state NEW -p udp –i $IF_ROUTER –s
$IP_ROUTER –o $IF_SECFW –d $IP_LOG_ADM --dport $SLOG_PORT -j ACCEPT

# firewall
iptables -A OUTPUT –m state –state NEW -p udp –o $IF_SECFW –d
$IP_LOG_ADM --dport $SLOG_PORT -j ACCEPT
```

- NTP synchronization for external service servers/security devices

```
# external service
iptables -A FORWARD –m state –state NEW -p udp –i $IF_EXTERNAL –s
$IP_EXT –o $IF_SECFW –d $IP_NTP_INT --dport $NTP_PORT -j ACCEPT

# router
iptables -A FORWARD –m state –state NEW -p udp –i $IF_ROUTER –s
$IP_ROUTER –o $IF_SECFW –d $IP_NTP_INT --dport $NTP_PORT -j ACCEPT

# firewall
iptables -A OUTPUT –m state –state NEW -p udp –o $IF_SECFW –d
$IP_NTP_INT --dport $NTP_PORT -j ACCEPT
```

- Synchronization of NTP server with stratum 2 NTP servers

```
iptables -A FORWARD –m state –state NEW -p udp –i $IF_SECFW –s
$IP_NTP_INT –o $IF_ROUTER –d $IP_NTP1 --dport $NTP_PORT -j ACCEPT

iptables -A FORWARD –m state –state NEW -p udp –i $IF_SECFW –s
$IP_NTP_INT –o $IF_ROUTER –d $IP_NTP2 --dport $NTP_PORT -j ACCEPT
```

- Source NAT

for outbound connections

```
iptables –t nat -A POSTROUTING –o $IF_ROUTER –j SNAT --to-source
$IP_FW
```

- Destination NAT

for inbound connections to web proxy and NTP server

```
iptables –t nat -A PREROUTING –p tcp –i $IF_ROUTER –d $IP_WBX_GE
--dport $HTTP_PORT –j DNAT --to-destination $IP_WBX_USR

iptables –t nat -A PREROUTING –p tcp –i $IF_ROUTER –d $IP_WBX_GE
--dport $HTTPS_PORT –j DNAT --to-destination $IP_WBX_USR

iptables –t nat -A PREROUTING –p udp –i $IF_ROUTER –d $IP_NTP_GE
--dport $NTP_PORT –j DNAT --to-destination $IP_NTP_INT
```

- Log whatever is left. These packets will be dropped by the default policy.

```
iptables -A INPUT -j LOG --log-prefix ”iptables INPUT dropped”
iptables -A OUTPUT -j LOG --log-prefix ”iptables OUTPUT dropped”
iptables -A FORWARD -j LOG --log-prefix ”iptables FORWARD dropped”
```

Note, in the above, rules are ordered logically and not according to what is most frequently used as recommended. To verify what rules are most frequently used, logging rules could be added. Thus, by logging and examining the logs, the firewall may be further tuned for performance.

Assignment 3 – Verify the Firewall Policy

3.1 Introduction

The primary firewall policy is verified as described in this section. Prior to the verification work, a plan is made which includes:

- scope/aim
- schedule
- cost/effort

- risks
- technical approach/tools

During the verification, the commands/tools and results are recorded for post-verification analysis and evaluation.

3.2 Verification Plan

Scope/Aim
To verify the primary firewall security policy. This includes:
- verifying the primary firewall hardening measures
- verifying the primary firewall rule set
- reviewing the logging process and procedures

Schedule
To minimize disruption to business operations, the validation tests are scheduled on a weekend (Saturday or Sunday) evening.

Cost/effort
Existing equipment and free software will be used, so there will be no associated hardware/software costs. Planning, preparation/scripting time by a team of 2 IT staff is estimated to be 6 hours. This preparation time includes formulation of the technical plan, installation/configuration/testing of tools, preparation of scripts. The verfication tests will take an estimate of 10 hours. Logs and result analysis and reporting will take another 6 hours. The pre- and post-verification work can be done during office hours. As the verification tests will be carried out after office hours, over-time pay or leave in lieu of will be incurred.

Risks
The risks involved are low as the verification tests are non intrusive. During the testing, GE network will be slowed down. GE Customers, suppliers, partners will be notified of the scheduled testing. As the testing is at a low use period, it is anticipated that there will be minimal impact on business.

Approach/Tasks
- verifying the primary firewall hardening measures using nessus
- verifying the primary firewall rule set using nmap
  - Scan interfaces of the firewall to audit the traffic to/from the firewall. This validates the iptables INPUT and OUTPUT rules.
  - Scan networks behind the firewall to audit the traffic through the firewall. This validates the iptables FORWARD rules.
- reviewing the logging process and procedures
  - Review the logs for the scans and attacks launched during the test and ensure that the firewall logs captured the activity.

Tools
We will use several notebooks for the tests. These have the scanning tools installed.

Nessus [10]

A free and powerful security vulnerability scanner. Version 2.0.9 is used.
The standalone auto-install package is downloaded and installed:
`sh nessus-install.sh`

The nessus server daemon is nessusd. A nessusd user, admin1, is added.
Nessus server is started: `/opt/nessus/sbin/nessusd -D`
and runs at port 1241.

The nessus client daemon, nessus, is the user interface for configuring the scan. We can configure plugins and preferences, select scan options and targets from the GUI.

In our test, we will be using nessus for validating the firewall OS and not for port scanning.

Nmap [27]

Another free software, it may be used for port scanning, ping sweeps and OS identification. Version 3.0 is used.

The nmap man page gives a detailed description of the command:
`man nmap`

The –h option gives a useful command summary:
`nmap –h`

```
Nmap V. 3.00 Usage: nmap [Scan Type(s)] [Options] <host or net list>
Some Common Scan Types ('*' options require root privileges)
* -sS TCP SYN stealth port scan (default if privileged (root))
  -sT TCP connect() port scan (default for unprivileged users)
* -sU UDP port scan
  -sP ping scan (Find any reachable machines)
* -sF,-sX,-sN Stealth FIN, Xmas, or Null scan (experts only)
  -sR/-I RPC/Identd scan (use with other scan types)
Some Common Options (none are required, most can be combined):
* -O Use TCP/IP fingerprinting to guess remote operating system
  -p <range> ports to scan.  Example range: '1-1024,1080,6666,31337'
  -F Only scans ports listed in nmap-services
  -v Verbose. Its use is recommended.  Use twice for greater effect.
  -P0 Don't ping hosts (needed to scan www.microsoft.com and others)
* -Ddecoy_host1,decoy2[,...] Hide scan using many decoys
```

```
  -T <Paranoid|Sneaky|Polite|Normal|Aggressive|Insane> General timing
policy
  -n/-R Never do DNS resolution/Always resolve [default: sometimes
resolve]
  -oN/-oX/-oG <logfile> Output normal/XML/grepable scan logs to
<logfile>
  -iL <inputfile> Get targets from file; Use '-' for stdin
* -S <your_IP>/-e <devicename> Specify source address or network
interface
  --interactive Go into interactive mode (then press h for help)
Example: nmap -v -sS -O www.my.com 192.168.0.0/16 '192.88-90.*.*'
SEE THE MAN PAGE FOR MANY MORE OPTIONS, DESCRIPTIONS, AND EXAMPLES
```

The options that we will be using in our tests are: P0 (no ping), sT (TCP port scan), sU (UDP port scan).

3.3 Verification Tests

Primary firewall hardening measures
Nessus is used. Configured for no denial of service attack, no port scans, target host, the firewall.
Rules were added to the firewall to allow connection from the nessus scanner so that scanning could proceed.

No Vulnerabilities were found. This verifies the firewall hardening measures.

Primary firewall rule set
Verify TCP, UDP and ICMP traffic.

Verify traffic to/from/through the firewall
To verify the traffic, we scan the firewall interfaces and networks behind firewall.

For TCP traffic, we launch nmap TCP scan in all the segments: Internet, external services, internal services, users, admin. Nmap will find the open TCP ports on the scanned host.

Nmap UDP scans take too long and we restricted the port range for scanning.

a. TCP traffic
a1. From Internet
These scans ran with the notebook placed external to the router and firewall. Target hosts are firewall internet/router interface and hosts in external service network.

- firewall router interface

```
nmap -P0 -sT -p 1-65535 -e eth0 100.1.1.1

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 100.1.1.1:
(The 65535 ports scanned but not shown below are in state: filtered)

Nmap run completed -- 1 IP address (1 host up) scanned in 87 seconds
```

The firewall does not have any TCP ports opened.

- Reverse web proxy

```
nmap -P0 -sT -p 1-65535 -e eth0 100.1.1.18

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 100.1.1.18:
(The 65533 ports scanned but not shown below are in state: filtered)
Port       State       Service
80/tcp     open        http
443/tcp    open        https

Nmap run completed -- 1 IP address (1 host up) scanned in 85 seconds
```

Only TCP ports 80 and 443 are opened on the reverse web proxy.

- SMTP mail relay

```
nmap -P0 -sT -p 1-65535 -e eth0 100.1.1.19

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 100.1.1.19:
(The 65534 ports scanned but not shown below are in state: filtered)
Port       State       Service
25/tcp     open        smtp

Nmap run completed -- 1 IP address (1 host up) scanned in 86 seconds
```

Only TCP port 25 was opened on the SMTP relay.

- External DNS

```
nmap -P0 -sT -p 1-65535 -e eth0 100.1.1.20

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 100.1.1.20:
(The 65534 ports scanned but not shown below are in state: filtered)
Port       State       Service
53/tcp     open        domain
```

```
Nmap run completed -- 1 IP address (1 host up) scanned in 82 seconds
```

Only TCP port 53 was opened on the external DNS.

a2. From External service network

The scans ran with the notebook placed between external service network and firewall. Target hosts are firewall external service network interface and hosts in internal service, admin and user networks.

- firewall external service network interface

```
nmap -P0 -sT -p 1-65535 -e eth1 100.1.1.1

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 100.1.1.1:
(The 65535 ports scanned but not shown below are in state: filtered)

Nmap run completed -- 1 IP address (1 host up) scanned in 4 seconds
```

The firewall does not have any TCP ports opened.

- Web server

```
nmap -P0 -sT -p 1-65535 -e eth0 192.168.1.2

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.1.2:
(The 65533 ports scanned but not shown below are in state: filtered)
Port       State       Service
80/tcp     open        http
443/tcp    open        https

Nmap run completed -- 1 IP address (1 host up) scanned in 3 seconds
```

Only TCP ports 80 and 443 are opened on the web server.

- Mail server

```
nmap -P0 -sT -p 1-65535 -e eth0 192.168.1.3

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.1.3:
(The 65534 ports scanned but not shown below are in state: filtered)
Port       State       Service
25/tcp     open        smtp

Nmap run completed -- 1 IP address (1 host up) scanned in 3 seconds
```

Only TCP port 25 was opened on the mail server.

- Internal DNS

```
nmap –P0 –sT –p 1-65535 –e eth0 192.168.1.4

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.1.4:
(The 65535 ports scanned but not shown below are in state: filtered)

Nmap run completed -- 1 IP address (1 host up) scanned in 3 seconds
```

No TCP ports are opened on the internal DNS.

- NTP server

```
nmap –P0 –sT –p 1-65535 –e eth0 192.168.1.5

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.1.5:
(The 65535 ports scanned but not shown below are in state: filtered)

Nmap run completed -- 1 IP address (1 host up) scanned in 3 seconds
```

No TCP ports are opened on the NTP server.

- Central log server

```
nmap –P0 –sT –p 1-65535 –e eth0 192.168.2.2

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.2.2:
(The 65535 ports scanned but not shown below are in state: filtered)

Nmap run completed -- 1 IP address (1 host up) scanned in 4 seconds
```

No TCP ports are opened on the central log server.

- System administrator's workstation

```
nmap –P0 –sT –p 1-65535 –e eth0 192.168.2.3

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.2.3:
(The 65535 ports scanned but not shown below are in state: filtered)

Nmap run completed -- 1 IP address (1 host up) scanned in 2 seconds
```

No TCP ports are opened on the system administrator's workstation.

- Database server

```
nmap –P0 –sT –p 1-65535 –e eth0 192.168.2.4

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.2.4:
(The 65535 ports scanned but not shown below are in state: filtered)

Nmap run completed -- 1 IP address (1 host up) scanned in 3 seconds
```

No TCP port was opened on the database server.

- User workstation

```
nmap –P0 –sT –p 1-65535 –e eth0 192.168.3.2

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.3.2:
(The 65535 ports scanned but not shown below are in state: filtered)

Nmap run completed -- 1 IP address (1 host up) scanned in 3 seconds
```

No TCP port was opened on the user workstation.

- Web proxy

```
nmap –P0 –sT –p 1-65535 –e eth0 192.168.3.3

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.3.3:
(The 65533 ports scanned but not shown below are in state: filtered)
Port       State       Service
80/tcp     open        http
443/tcp    open         https

Nmap run completed -- 1 IP address (1 host up) scanned in 3 seconds
```

Only TCP ports 80 and 443 are opened on the web proxy.

a3. From Admin network

This scan ran with the notebook placed between the admin network and firewall. Target is the firewall secondary firewall interface and the external service network.

- firewall secondary FW interface

```
nmap –P0 –sT –p 1-65535 –e eth2 100.1.1.1
```

```
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 100.1.1.1:
(The 65534 ports scanned but not shown below are in state: filtered)
Port       State       Service
22/tcp     open        ssh

Nmap run completed -- 1 IP address (1 host up) scanned in 4 seconds
```

Onlt TCP port 22 was opened on the firewall.

The scans for the reverse web proxy, SMTP relay, external DNS also show TCP port 22 was opened on these servers.

b. UDP traffic

The nmap UDP scans take an average of 1 second per port. For 65535 ports, it would take 18 hours or more. So, we will only test a subset of 1-1024 ports.

b1. From Internet

The nmap UDP port scans from outside the router/firewall to the external service network returns no open ports for all the hosts except:

- External DNS

```
nmap –P0 –sU –p 1-1024 –e eth0 100.1.1.20

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 100.1.1.20:
(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 1141 seconds
```

Only UDP port 53 is opened on the external DNS.

b2. From External service network

The nmap UDP port scans from the external service network, returns no open ports for all the internal service network hosts except:

- NTP server

```
nmap –P0 –sU –p 1-1024 –e eth0 192.168.1.5

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on 192.168.1.5:
(The 65534 ports scanned but not shown below are in state: filtered)
```

```
Port       State       Service
123/udp     open        ntp

Nmap run completed -- 1 IP address (1 host up) scanned in 1036 seconds
```

Only UDP port 123 is opened on the NTP server.

b3. From Admin network

The scans ran with the notebook placed between the admin network and firewall. Target is the firewall secondary firewall interface and the external service hosts. There are no open UDP ports.

c. ICMP traffic

We use ping command to test ICMP traffic.

c1. From Internet

- ping firewall, no reply was received:

```
PING 100.1.1.1 (100.1.1.1): 56(84) bytes of data.

--- 100.1.1.1 ping statistics ---
3 packets transmitted, 0 received, 100% loss, time 1999ms
```

- ping external service host (reverse web proxy), no reply was received:

```
PING 100.1.1.18 (100.1.1.18): 56(84) bytes of data.

--- 100.1.1.18 ping statistics ---
5 packets transmitted, 0 received, 100% loss, time 4017ms
```

c2. From External service network

- ping firewall, no reply was received:

```
PING 100.1.1.17 (100.1.1.17): 56(84) bytes of data.

--- 100.1.1.17 ping statistics ---
3 packets transmitted, 0 received, 100% loss, time 2015ms
```

- ping internal service host (web server), no reply was received:

```
PING 192.168.1.2 (192.168.1.2): 56(84) bytes of data.

--- 192.168.1.2 ping statistics ---
4 packets transmitted, 0 received, 100% loss, time 2997ms
```

c3. From Admin network

o ping firewall from system administrator's workstation, reply was received:

```
PING 192.168.0.1 (192.168.0.1): 56(84) bytes of data.
64 bytes from 192.168.0.1: icmp_seq=1 ttl=63 time=0.227 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=63 time=0.179 ms
64 bytes from 192.168.0.1: icmp_seq=3 ttl=63 time=0.193 ms
64 bytes from 192.168.0.1: icmp_seq=4 ttl=63 time=0.184 ms
64 bytes from 192.168.0.1: icmp_seq=5 ttl=63 time=0.183 ms

--- 192.168.0.1 ping statistics ---
5 packets transmitted, 5 received, 0% loss, time 3996ms
```

o ping external service host (external DNS) from system administrator's workstation, reply was received:

```
PING 100.1.1.20 (100.1.1.20): 56(84) bytes of data.
6 packets transmitted, 6 received, 0% loss, time 4999ms
64 bytes from 100.1.1.20: icmp_seq=1 ttl=63 time=0.376 ms
64 bytes from 100.1.1.20: icmp_seq=2 ttl=63 time=0.353 ms
64 bytes from 100.1.1.20: icmp_seq=3 ttl=63 time=0.351 ms
64 bytes from 100.1.1.20: icmp_seq=4 ttl=63 time=0.353 ms

--- 100.1.1.20 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
```

Reviewing the logging process and procedures

Sizable logs were captured by the scans. The logs were reviewed and found to be consistent with the firewall ruleset. All dropped packets are logged with appropriate identifiers.

3.4 Summary of Results

The nessus scan results showed no vulnerabilities on the firewall.
The nmap scans to verify the firewall ruleset showed no unexpected results.
All dropped packets are logged.

Due to limitation of time, the UDP port scans were restricted to the privileged ports (1-1024). This port range is sufficient to verify the UDP services set out in the firewall policy.

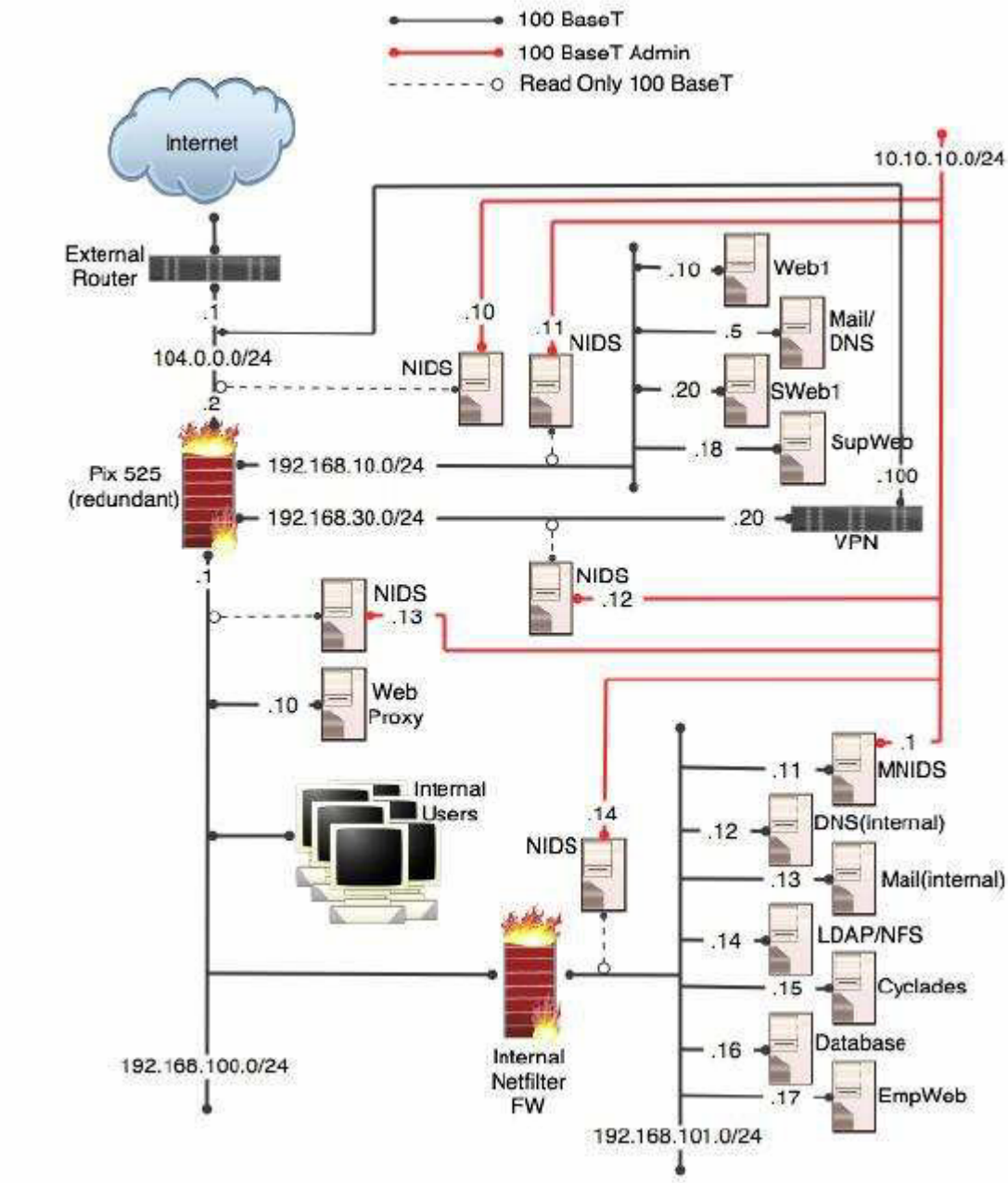
Having met all the requirements, the firewall policy is considered verified.

Assignment 4 – Design under Fire

4.1 Introduction

The practical submitted by Ben Nelson, analyst number 0438, at http://www.giac.org/practical/GCFW/Ben_Nelson_GCFW.pdf [28], is selected for the requirements of this section.

His network design is reproduced here:



To research this part, we used the Security Focus website http://www.securityfocus.com/bid [29]. This houses a comprehensive repository on vulnerabilities for a very wide range of hardware, operating systems and applications. Not only are vulnerabilities listed, but information on exploits and solutions are also available. Other useful vulnerability repositories are CERT http://www.cert.org/nav/index_red.html [30] and Security Tracker http://www.securitytracker.com/ [31].
The Cisco security website at http://www.cisco.com/warp/public/707/advisory.html was also referenced.

4.2 Attack against Firewall

Background
The primary firewall used in Ben Nelson's design is a redundant pair of Cisco Pix 6.2.2 firewalls. From the Security Focus website, this recent vulnerability discovered by John Airey was found [32]:

Security Focus bugtraq #8754 (http://www.securityfocus.com/bid/8754) of 3 Oct 2003:
Cisco PIX ICMP Echo Request Network Address Translation Pool Exhaustion Vulnerability

"A problem has been reported in Cisco PIX network firewalls when global IP address pools are exposed to ICMP traffic. This may result in a denial of service to network resources."

On Security Tracker, this vulnerability is assigned alert ID 1007877 [33] http://www.securitytracker.com/alerts/2003/Oct/1007877.html with the following information:

Description: A denial of service vulnerability was reported in the Cisco PIX firewall. A remote user can cause the firewall's pool of network address translation (NAT) addresses to become exhausted.

It is reported that a remote user can send a large amount of ICMP echo request packets to the global pool IP addresses on the target firewall to cause the firewall to prevent the release of the associated NAT address.

According to the report, a reboot or reload of the firewall will cause the device to return to normal operations.

Impact: A remote user can cause all available NAT addresses to be used, preventing further NAT connections.

Solution: No solution was available at the time of this entry. The vendor is reportedly working on a fix.

John Airey's report and Cisco's response [34]:
http://lists.netsys.com/pipermail/full-disclosure/2003-October/011356.html
http://lists.netsys.com/pipermail/full-disclosure/2003-October/011379.html

Cisco bug ID CSCec47609 has been opened to investigate this issue. This is associated with the Security Notice about the "Nachi Worm Mitigation Recommendations" [35] http://www.cisco.com/warp/public/707/cisco-sn-20030820-nachi.shtml#pix.

Plan
Launch a DDoS ICMP flood attack against the PIX to deplete the NAT pool.

Execution
Download the TFN2K code from packetstorm [40], install. Launch the attack on the client host:

```
./tfn -f servers.txt -c 6 -d 5 -i 104.0.0.1
```

where the option:
-f specifies the file for the server IP addresses
-c 6 specfies ICMP echo-reply attack
-d specfies number of decoy packets to be sent
-i specifies the IP address of the target host

Not long after the attack launch, we expect a flood of ICMP packets targeting the target's network, disabling it to legitimate traffic.

The attack may work even though Ben has designed redundancy into the firewall by having a pair of PIX, as the backup PIX is still susceptible to the same vulnerability.

Countermeasures
Cisco has discussed workaround information in the Nachi Worm Mitigation Recommendations notice [35] and listed here:

A workaround suggested by Cisco is to use a global PAT address.

The default behavior of the PIX is to block traffic from lower security level interfaces (OUTSIDE) to higher security level interfaces (INSIDE) unless the affected ports and protocols have been explicitly permitted by an access list or conduit.

In addition, Cisco recommends blocking traffic from higher security level interfaces (INSIDE) to lower security level interfaces (OUTSIDE).

Customers should deny outbound attempts to these ports:

```
access-list acl_inside deny icmp any any echo
access-list acl_inside deny icmp any any echo-reply
access-list acl_inside deny tcp any any eq 135
access-list acl_inside deny udp any any eq 135
access-list acl_inside deny udp any any eq 69
access-list acl_inside deny tcp any any eq 137
access-list acl_inside deny udp any any eq 137
access-list acl_inside deny tcp any any eq 138
access-list acl_inside deny udp any any eq 138
access-list acl_inside deny tcp any any eq 139
access-list acl_inside deny udp any any eq 139
access-list acl_inside deny tcp any any eq 445
access-list acl_inside deny tcp any any eq 593

! --- insert previously configured acl statements here,
! --- or permit all other traffic out

access-list acl_inside permit ip any any

access-group acl_inside in interface inside
```

The corresponding outbound lists may be applied, however, ACLs are strongly recommended in lieu of outbound lists.

Cisco Security Agent (CSA)

Using the Desktop and Default Server policies within CSA successfully mitigates this vulnerability/worm.

4.3 Distributed Denial of Service

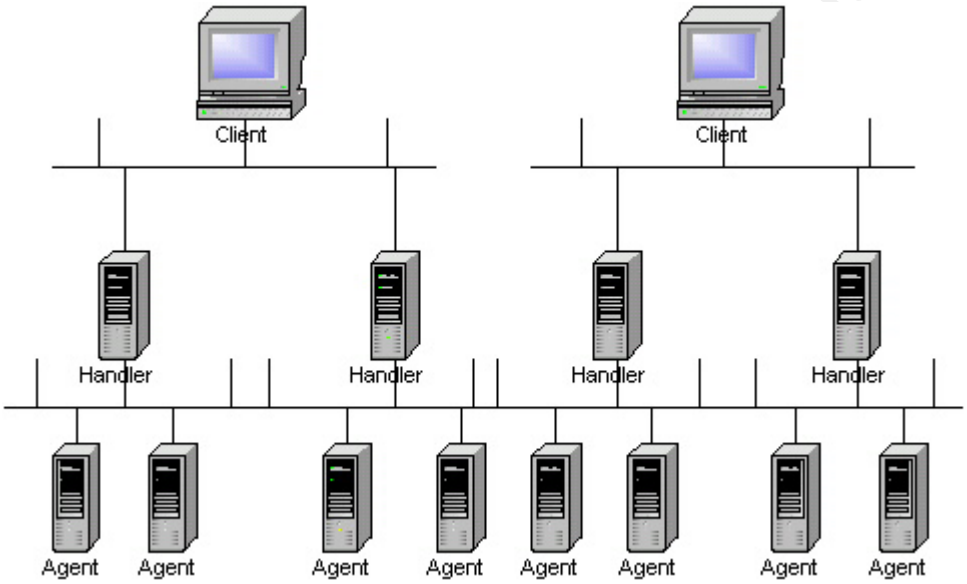
Background

A distributed Denial of Service (DDoS) attack is to be launched against the selected target from 50 compromised modem/DSL hosts. In a Denial of Service (DoS), the service or access to the service is disabled/disrupted, and it is not available to legitimate users of the service. One attack to achieve DoS is to "flood" the network, thereby preventing legitimate network traffic to the service. A DDoS, uses many machines, in distributed locations, from which to launch the attack. Hundreds or even thousands of

machines may be involved in the DDoS. According to the following Cisco white paper [36] http://www.cisco.com/warp/public/707/newsflash.html, the DDoS may be divided into the following steps:

- a scan phase, in which a large number of hosts are probed for a known vulnerability
- compromise vulnerable hosts to gain access
- DDoS tool installed on each host
- compromised hosts used in further scanning and compromises

The main players in a DDoS are: client hosts that orchestrates the attack, handlers/ masters which are compromised hosts that control multiple agents, as seen in the following diagram taken from the Cisco white paper [36]. Daemon processes running on these agents are responsible for generating packets targeted at the victim.



There are 4 programs commonly used in DDoS attacks. These are Trinoo, TFN, TFN2K and Stacheldraft.

Plan

For our DDoS on the selected network, we will use TFN2K. Four attack styles are possible: TCP/SYN, UDP, ICMP Echo request or Broadcast Ping (SMURF) packet flood. We choose to use SYN flood attack on TCP port 80 of the target, which is confirmed to be opened by connecting to the site through a web browser. Communication between handlers and agents is via TCP, UDP, ICMP or a random mix. It is in one direction, from the client to the daemon, no replies are needed. Detection is difficult [39].

TFN2K code is available for Windows, various unix and linux platforms. The TFN2K code used is by Mixter, downloaded from http://packetstormsecurity.nl/groups/mixter/ [40]. The zipped tarball unzips to a README file and several files containing c source codes. Before compiling, the OS is selected in the Makefile. The configuration file

config.h defines the default constants and parameter values used, e.g., child_max controls the number of targets handled by a server. We do not change the default settings in the configuration file, which is shown here for reference.

```
/*
 * Tribe FloodNet - 2k edition
 * by Mixter <mixter@newyorkoffice.com>
 *
 * config.h - user defined values
 *
 * This program is distributed for educational purposes and without
any
 * explicit or implicit warranty; in no event shall the author or
 * contributors be liable for any direct, indirect or incidental
damages
 * arising in any way out of the use of this software.
 *
 */

#ifndef _CONFIG_H

#define HIDEME "tfn-daemon"     /* background process name */
#define HIDEKIDS "tfn-child"    /* flood/shell thread names */
#define CHLD_MAX 50             /* maximum targets a server handles at
a time */
#define DELIMITER "@"           /* to separate ips and broadcasts
(host1@host2@.
..) */
#define REQUIRE_PASS            /* require server password to be
entered and
                                   verified before the client will
work? */

#undef ATTACKLOG "attack.log"   /* keep server side logs of attacked
victims */

/* Note: the password is not defined here, but at compile time. The
   requests will be encrypted anyways, you DON'T need to change this
*/

#define PROTO_SEP       '+'     /* session header separator, can be
anything */
#define ID_SHELL        'a'     /* to bind a root shell */
#define ID_PSIZE        'b'     /* to change size of udp/icmp packets
*/
#define ID_SWITCH       'c'     /* to switch spoofing mode */
#define ID_STOPIT       'd'     /* to stop flooding */
#define ID_SENDUDP      'e'     /* to udp flood */
#define ID_SENDSYN      'f'     /* to syn flood */
#define ID_SYNPORT      'g'     /* to set port */
#define ID_ICMP         'h'     /* to icmp flood */
```

```
#define ID_SMURF        'i'     /* haps! haps! */
#define ID_TARGA        'j'     /* targa3 (ip stack penetration) */
#define ID_MIX          'k'     /* udp/syn/icmp intervals */
#define ID_REXEC        'l'     /* execute system command */

#define _CONFIG_H
#endif
```

Execution

The TFN2k server (daemon) td is installed on the compromised hosts as root. The TRN2K client process tfn is installed on the client host. The TFN2K server IP addresses are captured in a text file servers.txt. On the client host, the TFN2K attack is launched by executing the tfn process:

```
./tfn –f servers.txt –c 5 –d 5 –i 104.0.0.1
```

where the option:
-f specifies the file for the server IP addresses
-c 5 specfies SYN flood attack
-d specfies number of decoy packets to be sent
-i specifies the IP address of the target host

Not long after the attack launch, we expect a flood of SYN packets targeting the target's network, disabling it to legitimate traffic.

Countermeasures

There is no known way to defend against TFN2K. It uses stealth client/serevr communication: spoofed source addresses, advanced encryption, one-way communication, random IP protocols and decoy packets. There are no reliable attack signatures. The most effect countermeasure is to prevent the network from being used as clients or servers. Basically, take measures to ensure that your systems are not vulnerable to attacks that would allow intruders to install TFN2K.

Jason Barlow and Woody Thrower [39] suggest the following:

- Use a firewall that exclusively employs application proxies. This should effectively block all TFN2K traffic. Exclusive use of application proxies is often impractical, in which case the allowed non-proxy services should be kept to a minimum.

- Disallow unnecessary ICMP, TCP, and UDP traffic. Typically only ICMP type 3 (destination unreachable) packets should be allowed. If ICMP cannot be blocked, disallow unsolicited (or all) ICMP_ECHOREPLY packets.

- Disallow UDP and TCP, except on a specific list of ports.

- Spoofing can be limited by configuring the firewall to disallow any outgoing packet whose source address does not reside on the protected network.

Other measures are detailed in the Cisco white paper [36] listed above.

4.4 Attack on Internal System

Background
The chosen target internal system for attack is the internal mail server. This runs Sendmail 8.12.8 on a RedHat Linux box. A number of "prescan" vulnerabilities associated with this version of Sendmail have been found on searching the Security Focus and CERT websites:

Security Focus bugtraq #7230 (http://www.securityfocus.com/bid/7230) of 27 Sep 2003 [41]: Sendmail Address Prescan Memory Corruption Vulnerability
It was reported by Michal Zalewski and the following description was given:

"A vulnerability has been discovered in Sendmail which could be exploited remotely to execute arbitrary code. The flaw is present in the prescan() procedure, one that is used for processing e-mail addresses in SMTP headers. It has been confirmed that this condition may be exploited by remote attackers to execute instructions on target systems. This vulnerability is due to a logic error in the conversion of a char to an integer value. It is eliminated in Sendmail version 8.12.9."

Another vulnerability different from above is:
CERT Vulnerability Note VU#784980 (http://www.kb.cert.org/vuls/id/784980) [42]
Sendmail prescan() buffer overflow vulnerability
This was reported for Sendmail version 8.12.9 on 17 Sept 2003 by Michal Zalewski. Here is the description given:

"When processing email messages, sendmail creates tokens from address elements (user, host, domain). The code that performs this function (prescan() in parseaddr.c) contains a vulnerability that could allow a remote attacker to overwrite memory structures and execute arbitary code. The attacker could exploit this vulnerability using an email message with a specially crafted address. Such a message could be passed through MTAs that are not vulnerable."

CERT Advisory CA-2003-25 Buffer overflow in Sendmail [43] gives another description (http://www.cert.org/advisories/CA-2003-25.html):

"The email attack vector is message-oriented as opposed to connection-oriented. This means that the vulnerability is triggered by the contents of a specially crafted email message rather than by lower-level network traffic. This is important because an MTA that does not contain the vulnerability may pass the malicious message along to other MTAs that may be protected at the network level. In other words, vulnerable sendmail servers on the interior of a network are still at risk, even if the site's border MTA uses

software other than sendmail. Also, messages capable of exploiting this vulnerability may pass undetected through packet filters or firewalls. “

A writeup from Michal Zalewski taken from the web site [44]: http://archives.neohapsis.com/archives/fulldisclosure/2003-q3/4119.html:

Overview

There seems to be a remotely exploitable vulnerability in Sendmail up to and including the latest version, 8.12.9. The problem lies in prescan() function, but is not related to previous issues with this code.

The primary attack vector is an indirect invocation via parseaddr(), although other routes are possible. Heap or stack structures, depending on the calling location, can be overwritten due to the ability to go past end of the input buffer in strtok()-alike routines.

Attack details

Local exploitation on little endian Linux is confirmed to be trivial via recipient.c and sendtolist(), with a pointer overwrite leading to a neat case of free() on user-supplied data, i.e.:

eip = 0x40178ae2
edx = 0x41414141
esi = 0x61616161

SEGV in chunk_free (ar_ptr=0x4022a160, p=0x81337e0) at malloc.c:3242

0x40178ae2 <chunk_free+486>: mov %esi,0xc(%edx)
0x40178ae5 <chunk_free+489>: mov %edx,0x8(%esi)

Remote attack is believed to be possible.

Workaround / fix

Vendor was notified, and released an early patch attached below. There are no known workarounds.

```
Index: parseaddr.c
===================================================================
=
RCS file: /cvs/src/gnu/usr.sbin/sendmail/sendmail/parseaddr.c,v
retrieving revision 1.16
diff -u -r1.16 parseaddr.c
--- parseaddr.c 29 Mar 2003 19:44:01 -0000 1.16
+++ parseaddr.c 16 Sep 2003 17:37:26 -0000
@@ -700,7 +700,11 @@
                                        addr[MAXNAME] = '\0';
```

```
        returnnull:
                                if (delimptr != NULL)
+ {
+ if (p > addr)
+ p--;
                                        *delimptr = p;
+ }
                                CurEnv->e_to = saveto;
                                return NULL;
                        }
```

Plan
Verify the version of Sendmail by checking the mail banner (example, by telneting to port 25 of the mail server).

Look for codes that exploit these vulnerabilities. For the first Sendmail vulnerability (bugtraq #7230), the Security Focus website has a "proof of concept" exploit at http://downloads.securityfocus.com/vulnerabilities/exploits/bysin2.c [45].

No exploit codes were found for the second Sendmail vulnerability.

Execution
Unfortunately due to absence of the exploit codes and limitations of time and experience, we could not really test this vulnerability. Nevertheless, the numerous vulnerabilities reported for Sendmail, does seem to imply future opportunities.

Countermeasures
Keeping Sendmail patched and up-to-date. These vulnerabilities are fixed in the latest version of Sendmail (8.12.10).

References

Books
[1] Northcutt, Stephen, et. al. Inside Network Perimeter Security. Boston: New Riders Publishing, 2003.

[2] SANS Track 2 - Firewalls, Perimeter Protection and VPNs. SANS Institute, 2003.

[3] Ziegler, Robert L., Constantine, Carl B. Linux Firewalls. Boston: New Riders Publishing, 2002.

Web sites

[4] “IANA IP v4 address space.” 5 April 2003.
URL: http://www.iana.org/assignments/ipv4-address-space (17 Sept 2003).

[5] “SANS Computer Security Resources.” URL: http://www.sans.org/resources

[6] “Titan Security Toolkit.” URL: http://www.fish.com/titan (17 Sept 2003).

[7] “Bastille Linux.” URL: http://www.bastille-linux.org (17 Sept 2003).

[8] “Zone Labs.” http://www.zonelabs.com/store/content/home.jsp (17 Sept 2003).

[9] “The Center for Internet Security.” URL: http://www.cisecurity.org (17 Sept 2003).

[10] “Nessus.” 30 Aug 2003. URL: http:// www.nessus.org (17 Sept 2003).

[11] Boran, Sean. “Hardening Unix Applications.” 8 Oct 2002.
URL:http://www.boran.com/security/sp/application_hardening.html#Email%20gateways%20(smtp) (23 Oct 2003).

[12] “Beyond-Security’s SecuriTeam.com SWATCH.”
URL: http://www.securiteam.com/tools/5RP062K5HM.html (18 Sept 2003).

[13] “Cisco routers.” URL: http://www.cisco.com/en/US/products/hw/routers/index.html (22 Sept 2003).

[14] “Netfilter.” URL: http://www.iptables.com (23 Sep 2003).

[15] “Linux Firewall and Security Site.”
URL: http://linux-firewall-tools.com/linux/ (23 Sept 2003).

[16] “Sun iForce VPN/Firewall appliance.”
http://www.checkpoint.com/products/choice/platforms/sun_iforce.html (23 Sept 2003).

[17] “Check Point Express.” URL: http://www.checkpoint.com/products/express/
URL: http://www.checkpoint.com/products/downloads/express_datasheet.pdf (23 Sept 2003).

[18] “Sun ONE Web Proxy Server.”
URL: http://developers.sun.com/prodtech/webproxy/ (25 Sept 2003).

[19] “Sun Java System Web Proxy Server 3.6.”
URL: http://wwws.sun.com/software/products/web_proxy/home_web_proxy.html
URL: http://wwws.sun.com/software/products/web_proxy/ds_web_proxy.html (25 Sept 2003).

[20] “Sun Fire 120 Server.”
URL: http://www.sun.com/servers/entry/v120/ (25 Sept 2003).

[21] “Squid Web Proxy Cache.” URL: http://www.squid-cache.org (25 Sept 2003).

[22] “Squid version 2.5.” URL: http://www.squid-cache.org/Versions/v2/2.5/ (25 Sept 2003).

[23] “Snort.” 6 Nov 2003. URL: http:// www.snort.org (9 Nov 2003).

[24] “Trend Micro Interscan VirusWall.”
URL: http://www.trendmicro.com/en/products/gateway/isvw/evaluate/overview.htm (28 Sept 2003).

[25] “Sendmail.” URL: http://www.sendmail.org (30 Sept 2003).

[26] “PostgreSQL.” 6 Nov 2003. URL: http://www.postgresql.org/ (9 Nov 2003).

[27] “Nmap.” URL: http://www.nmap.org (12 Oct 2003).

[28] Ben Nelson’s GCFW practical.
URL: http://www.giac.org/practical/GCFW/Ben_Nelson_GCFW.pdf (18 Oct 2003).

[29] Security Focus Vulnerabilities.
URL: http://www.securityfocus.com/bid (20 Oct 2003).

[30] CERT Vulnerabilities, incidents & fixes.
URL: http://www.cert.org/nav/index_red.html (20 Oct 2003).

[31] Cisco Product Security Advisories and Notices.
URL: http://www.cisco.com/warp/public/707/advisory.html (20 Oct 2003).

[32] Security Focus Vulnerabilities. “Cisco PIX ICMP Echo Request Network Address Translation Pool Exhaustion Vulnerability.” 3 Oct 2003.
URL: http://www.securityfocus.com/bid/8754 (20 Oct 2003).

[33] Security Tracker Archives. “Cisco PIX NAT Pool Can Be Consumed With ICMP Echo Request Packets.” 3 Oct 2003.
URL: http://www.securitytracker.com/alerts/2003/Oct/1007877.html (20 Oct 2003).

[34] “Potential denial of service bug in Cisco Pix Firewall IOS 6.2.2 and 6.3.(3.102).” 3 Oct 2003.
URL: http://lists.netsys.com/pipermail/full-disclosure/2003-October/011356.html
URL: http://lists.netsys.com/pipermail/full-disclosure/2003-October/011379.html
(22 Oct 2003).

[35] Cisco Security Notice. “Nachi Worm Mitigation Recommendations.” 10 Nov 2003. URL: http://www.cisco.com/warp/public/707/cisco-sn-20030820-nachi.shtml#pix (15 Nov 2003).

[36] Cisco White Paper. “Strategies to protect against Distributed DoS attacks.” 29 Apr 2003. URL: http://www.cisco.com/warp/public/707/newsflash.html (25 Oct 2003).

[37] “CERT Advisory CA-1999-17 Denial of Service Tools.” 3 Mar 2000. URL: http://www.cert.org/advisories/CA-1999-17.html (25 Oct 2003).

[38] “CERT Note IN-99-07 Distributed Denial of Service Tools.” 15 Jan 2001. URL: http://www.cert.org/incident_notes/IN-99-07.html#tfn (25 Oct 2003).

[39] Barlow, Jason and Thrower, Woody. “TFN2k – An Analysis.” URL: http://www.chi-publishing.com/portal/backissues/pdfs/ISB_2000/ISB0502/ISB0502JBWT.pdf (25 Oct 2003).

[40] Packet Storm Archives. “Tkn2k.tgz.” URL: http://packetstormsecurity.nl/groups/mixter (26 Oct 2003).

[41] Security Focus Vulnerabilities. “Sendmail Address Prescan Memory Corruption Vulnerability.” 27 Sep 2003. URL: http://www.securityfocus.com/bid/7230 (29 Oct 2003).

[42] “CERT Advisory CA-2003-25 Buffer overflow in Sendmail.” 29 Sep 2003. URL: http://www.cert.org/advisories/CA-2003-25.html (29 Oct 2003).

[43] CERT Vulnerability Note VU#784980. “Sendmail prescan() buffer overflow vulnerability.” 29 Sep 2003. URL: http://www.kb.cert.org/vuls/id/784980 (29 Oct 2003).

[44] Neohapsis Archives. “Sendmail 8.12.9 prescan bug (a new one) [CAN-2003-0694].” 27 Sep 2003. URL: http://archives.neohapsis.com/archives/fulldisclosure/2003-q3/4119.html (31 Oct 2003).

[45] Security Focus Vulnerabilities. “bysin2.c.” URL: http://downloads.securityfocus.com/vulnerabilities/exploits/bysin2.c (31 Oct 2003).