



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

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

Practical Assignment
GIAC Certified Firewall Analyst (GCFW)
Capital SANS
Washington, D.C.
December 10-15, 2000

Brett Gordon

Table of Contents

<u>Introduction</u>	3
<u>Part I – Design the architecture</u>	3
<u>Network drawing</u>	4
<u>Devices</u>	6
<u>Part II – Security Policy</u>	6
<u>Router</u>	7
<u>Firewall</u>	9
<u>Firewall rules</u>	14
<u>Part III – Audit the design</u>	19
<u>Part IV – Design under fire</u>	23
<u>References:</u>	29

© SANS Institute 2000 - 2002, Author retains full rights.

Introduction

Pursuit of the GCFW certification requires completion of a practical assignment prior to taking the actual certification exams. This document represents my submittal.

The Assignment

PART I- Design the Architecture

A \$200 hundred million dollar a year e-business, GIAC Enterprises sells fortune cookies to customers throughout the world. As is common in the age of the internet GIAC Enterprises will conduct a majority of its business via the web. Customers, vendors and partners will all have varying degrees of access to the GIAC Enterprises corporate offices. While the ideal security model would isolate the corporate network from the outside world business needs dictate otherwise.

Our first step must then be to determine what type of access is driven by business need. If we look at all the people that may require some level of access into or out of our network we can classify them into several distinct groups:

Corporate Users- This group represents the standard non-privileged employee at GIAC Enterprises. Subject to the internal corporate security policies concerning acceptable network and internet use, allowed general network and internet access.

Privileged Corp. Users- Certain GIAC employees require access to several key servers where financial and HR data is stored. For security reasons the privileged users and their related servers are on their own network segment behind an internal firewall.

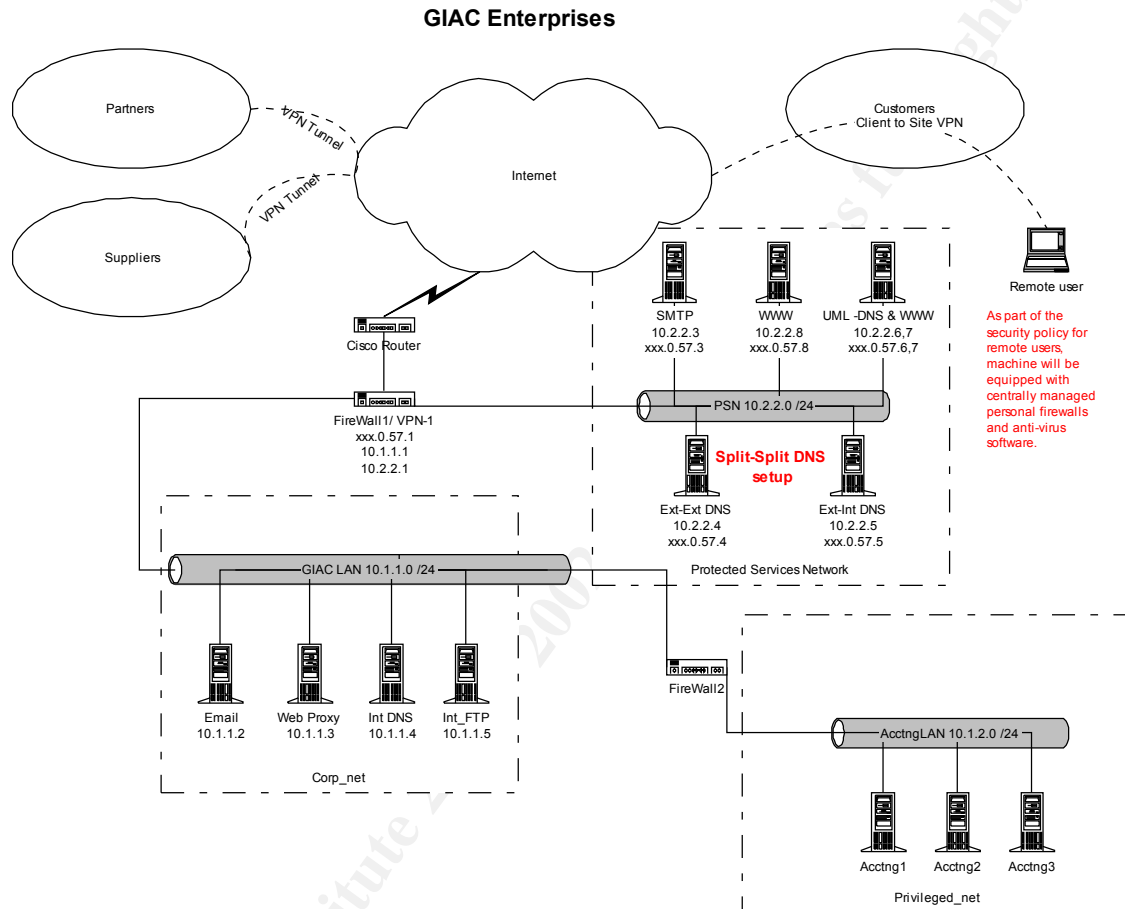
Partners Sites- GIAC Enterprises conducts business with certain partners. Partners are the international businesses that translate and resell fortunes. Partners will require 2-way access to the SQL servers where the relevant data is stored. SQL server replication will need to take place between partners and the GIAC servers.

Supplier Sites- The authors of fortune cookie sayings that connect to supply fortunes. Suppliers will require FTP access to the internal FTP servers in order to upload data updates.

Customers /Web users- The companies that purchase bulk online fortunes. For all intents and purposes, customers will be considered un-trusted and subjected to the same access control as any web user. A secure website will be provided to support SSL for actual online transactions and account information.

Remote Users- While out of the office GIAC employees are allowed access to the corporate LAN via an established telecommuter policy. The policy defines a set of standards for a client to site VPN. Since we will be using Checkpoint Firewall-1 / VPN-1

as our firewall product, SecuRemote will be the software installed on the telecommuter machines. Among other things the 'telecommuter policy' defines requirements for personal firewalls that limit inbound access and antiviral software with regular updates.



Using the above requirements we can build a simple matrix. Plotting source against destination, we map out our access requirements.

Source	Protected Services Network	Destination															
							Protected Services Network						Corporate Network				
			Partner Site	Supplier Site	Customer / Internet	Telecommuters	UML1-WWW	UML2-DNS	Ext_ext_DNS	Ext_int_DNS	Ext SMTP Relay	Ext Web	SQL server	FTP server	DNS server	Email Server	Proxy server
		Partner Site	N/A	No Access	No Access	No Access	TCP 80 TCP 443	UDP 53 TCP 53	UDP 53 TCP 53	No Access	TCP 25	Http Https	SQL traffic	No Access	UDP 53 TCP 53	TCP 25	No Access
		Suppliers Site	No Access	N/A	No Access	No Access	TCP 80 TCP 443	UDP 53 TCP 53	UDP 53 TCP 53	No Access	TCP 25	Http Https	No Access	FTP 20 FTP 21	No Access	No Access	No Access
		Customers / Internet	No Access	No Access	Not Applicable	TCP 80 TCP 443	UDP 53 TCP 53	UDP 53 TCP 53	UDP 53 TCP 53	TCP 25	Http Https	No Access	No Access	No Access	No Access	No Access	
		Telecommuter	No Access	No Access		No Access	No Access	No Access	No Access	No Access	No Access	No Access	SQL traffic	No Access	UDP 53 TCP 53	No Access	
Protected Services Network	UML1-DNS	No Access	No Access	No Access	No Access	Not Applicable						No Access	No Access	No Access	No Access	No Access	
	UML2-WWW	No Access	No Access	No Access	No Access							No Access	No Access	No Access	No Access		
	Ext_ext_DNS	No Access	No Access	UDP 53 TCP 53	No Access							No Access	No Access	UDP 53 TCP 53	No Access	No Access	
	Ext_int_DNS	No Access	No Access	UDP 53 TCP 53	No Access							No Access	No Access	UDP 53 TCP 53	No Access	No Access	
	xt-SMTP relay	No Access	No Access	No Access	No Access							No Access	No Access	No Access	TCP 25	No Access	
	xt-WWW	No Access	No Access	No Access	No Access							No Access	No Access	No Access	No Access	No Access	
	Corp LAN	SQL	SQL traffic	No Access	No Access							No Access	No Access	No Access	No Access	No Access	No Access
FTP Server		No Access	No Access	No Access	No Access	No Access	No Access	No Access	No Access	No Access	No Access						
DNS server		No Access	No Access	UDP 53 TCP 53	No Access	No Access	UDP 53 TCP 53	UDP 53 TCP 53	UDP 53 TCP 53	No Access	No Access						
Email server		No Access	No Access	TCP 25	No Access	No Access	No Access	No Access	No Access	No Access	No Access						
Web Proxy		No Access	No Access	Http Https FTP	No Access	No Access	No Access	No Access	No Access	No Access	No Access						

Specific devices

Several devices in our architecture require special mention.

GIAC-FW1- Checkpoint FW-1 with VPN-1 strong v4.1 sp2 running on NT4.0 sp6

GIAC-FW2- Checkpoint FW-1 with VPN-1 strong v4.1 sp2 running on NT4.0 sp6

Router1- Cisco 2620 router

UML1 & 2 DNS & WWW -Using User Mode Linux setups we will create two honeypot virtual devices. The content of these devices will be exact replicas of the actual external DNS and web servers. The purpose will be to potentially distract a would be intruder. A port scan of the external GIAC IP network will reveal two DNS and two WWW servers.

Ext-ext-DNS- Located in the protected services network (PSN) this is the DNS server which the internet will query for publicly available GIAC devices. This is the first of three DNS servers in our split-split-DNS setup.

Ext-int-DNS- Located in the PSN, this server will be the DNS server that resolves public addresses for internal users. This lowers the possibility that someone can poison our DNS cache because this server will not take external queries.

SMTP relay - Located in the PSN, the SMTP relay box will forward inbound email to the internal mail server. The primary purpose of this box is to provide a layer between the internet and the email server.

Ext Web server- The home of www.giac-enterprises.com.

Int-DNS-server- This DNS server is located on the internal corporate LAN. It will serve as the nameserver for internal systems. It will not be accessible from the outside. When necessary it will query the Ext-int-DNS server for outside resolution.

Web Proxy- Internal users will access the internet via a proxy server. Direct outbound traffic will only be allowed on a case by case basis. Http, Https & FTP will all take place via the proxy.

FTP- Suppliers will connect to this server to upload data. The connections will only be via the VPN. There will be no other external access.

Part II - Security Policy

In this section we will explore the actual security policies of the perimeter devices. Specifics regarding the access control lists (ACL) are outlined below.

Routers

As part of the basic configuration of our router we take some simple steps to harden the IOS.

Globally on the router we will remove un-needed services:

```
no ip finger
no ip http server
no ip bootp server
no service tcp-small-services
no service udp-small-services
no cdp run
```

Specifically on the WAN interface (Serial 0) we will prevent source-routed packets and using our network as a smurf-amplification site.

```
no ip source-route
no ip directed-broadcast
```

As the actual edge device between GIAC and the ISP, we will perform our anti-spoof filtering here. We approach anti-spoofing rules by determining what traffic we should NEVER see on an interface. So what should we never see coming IN from the internet?

- The three networks that are reserved space according to the RFC 1918
 - 10.0.0.0 / 8
 - 172.16.0.0 /12
 - 192.168.0.0 /16
- The multicast range 224.0.0.0 - 239.255.255.255
- The loopback address 127.0.0.0 /8
- A source address of 0.0.0.0
- Additionally the following range are not assigned publicly; 169.254.0.0/16, 192.0.2.0/24, 240.0.0.0/5 & 248.0.0.0/5
- Last but certainly not least, traffic from the outside should never have a source address that comes from the address on range of the inside interface.

Note: Although we may actually have connectivity to remote networks that use some of the addresses we are explicitly blocking (e.g.; partners or suppliers using RFC 1918 addresses internally) the incoming traffic will be tunneled via a VPN. As the traffic enters our perimeter it will be encapsulated in packets that are from publicly routable addresses.

The actual router configuration for the WAN (serial 0) interface is as follows:

From global configuration mode we build the actual access-list;

Note: Access lists are order dependant. The first rule that a packet matches will cause it to exit the access-list. Rules that will be referenced more often should be placed higher up in the order for performance reasons. In our case we expect the 'permit any' rule will be hit most often. Unfortunately placing it any higher up in the order will render everything below it moot.

```
router(config)#access-list 101 deny ip xxx.0.57.0 0.255.255.255 any log
router(config)#access-list 101 deny ip 10.0.0.0 0.255.255.255 any log
router(config)#access-list 101 deny ip 127.0.0.0 0.255.255.255 any log
router(config)#access-list 101 deny ip 172.16.0.0 0.15.255.255 any log
router(config)#access-list 101 deny ip 192.168.0.0 0.0.255.255 any log
router(config)#access-list 101 deny ip 224.0.0.0 15.255.255.255 any log
router(config)#access-list 101 deny ip 169.254.0.0 0.0.255.255 any log
router(config)#access-list 101 deny ip 192.0.2.0 0.255.255.255 any log
router(config)#access-list 101 deny ip 240.0.0.0 31.255.255.255 any log
router(config)#access-list 101 deny ip 248.0.0.0 31.255.255.255 any log
router(config)#access-list 101 deny ip host 0.0.0.0 any log
router(config)#access-list 101 permit any any ← Note: Cisco ACLs have an
implicit deny. If we did not specify to allow all other traffic nothing would get in.
```

From the interface specific configuration we apply the access-list;

```
router(config)# interface s0
router(config-if)# ip access group 101 in
```

So what traffic should we NEVER see coming into the internal (Ethernet 0) interface? In this case we should never see anything with a source address other than the xxx.0.57.0/24 network. For this access list we will allow that traffic and deny all else.

From global configuration mode we build the actual access-list;

```
router(config)#access-list 20 allow xxx.0.57.0 0.255.255.255
router(config)#access-list 20 deny any any log
```

From the interface specific configuration we apply the access-list;

```
router(config)# interface e0
router(config-if)# ip access group 20 in
```

Lastly we do not want to allow any terminal access to the router from anywhere other than our xxx.0.57.0/24 network. To accomplish this we will add the following lines to our config file:

```
! Allow traffic from the xxx.0.57.0/24 network and deny all else
access-list 10 permit xxx.0.57.0 0.255.255.255
```

```
access-list 10 deny any log
!
! Apply access list to terminal lines
!
line vty 0 4
access-class 10 in
```

Firewalls

The firewall we will be using will be Checkpoint Firewall-1/VPN-1 software running on a Windows NT 4.0 operating system.

Before we install the Checkpoint software onto the NT box we should complete a few simple tasks in an effort to harden the OS.

- Install NT4.0 as a stand-alone server. Resist the temptation to join the domain. You are building a firewall not a server.

Note: One of the authentication methods which Checkpoint supports is “OS Password”. In many cases this will require the box have several of the services enabled which we will be disabling. While my preference is not to use OS password as an authentication method, your business needs may vary.

Additionally I chose not to install SNMP. SNMP was never designed to be a secure protocol and should therefore not find a home on our perimeter security device. It should be noted that without SNMP certain Checkpoint management feature will not work. In my estimation, the users will let me know the firewall is down well before the Status monitor will.

- TCP/IP should be the only protocol installed.
- Apply latest NT service pack.
- Disable the following services:
 - o Workstation
 - o Server
 - o Alerter
 - o Messenger
 - o Computer Browser
 - o TCP/IP Netbios Helper
 - o Wins Client (TCP/IP) (Under Control Panel/Devices)

Tip: If you disable the services instead of removing them you will not be prompted to ‘install networking’ every time you attempt to right click on network neighborhood.

- Install Checkpoint software.
- Apply latest Checkpoint service packs.
- Re-install NT service pack. (Better safe then sorry)

Tip: Under the environment tab in the computer properties, modify the path variable to include the FW's \bin directory. It's not a security benefit but it will make your life easier.

Using our matrix as a guide we can begin to define the actual rules for the firewalls devices. Outlined below are the steps involved in creating a security policy on a Checkpoint firewall.

Prior to creating the actual access rules you must define an object for each item you will referencing in the rule-base. For our design we will need to create the following objects:

(For illustration purposes we will assume one partner and one supplier site.)

Networks-

GIAC_net, Partner_site, Supplier_site

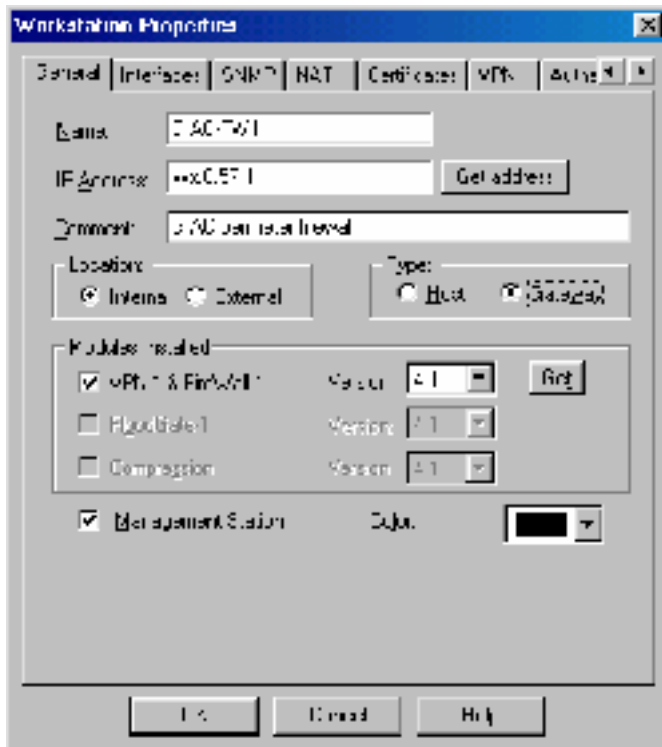
Devices-

GIAC-FW1, Partner_FW, Supplier_FW, PSN_UML_DNS, PSN_UML_WWW, PSN_SMTP, PSN-Ext-ext-DNS, PSN-Ext-int-DNS, PSN_WWW, Int_SQL, Int_SMTP, Int_Proxy, Int_FTP

Tip: When creating the rules you will be choosing from the list of objects you have previously created. Since this list will be alphabetically ordered, using a naming convention that will group like objects will make object selection easier.

Creating the objects

Since we will be referencing the firewall itself in some of the rules, we will start by defining the firewall object. Open the Policy Editor. Choose Manage/Network Objects/ then select New/ Workstation. Insert the following data in the proper fields.



Name:GIAC-FW1

IP Address: xxx.0.57.1

Note: Although our firewall will be multi-homed and have several IP addresses, it is important to use the external/public address here. This is for licensing as well as some functional reasons.

Choose

Gateway - Multi-homed vs single homed Workstation

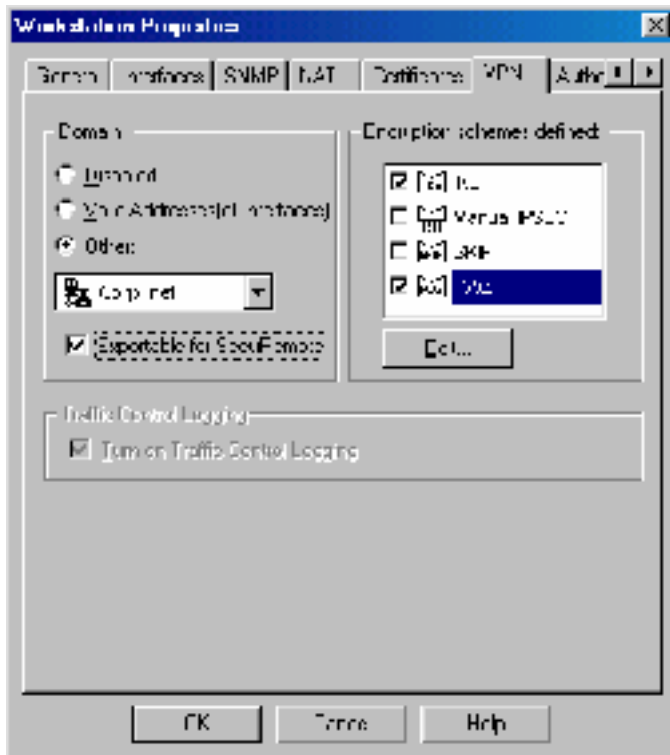
FW1 installed - Has Checkpoint Firewall-1 installed.

v4.1 - FW1 version

Internal - Managed by this management module.

Management Station. – In our design we are using a single gateway product from Checkpoint vs. a distributed setup. This box will function as a firewall module and management module. Don't get too concerned with all that right now, it merely reflects the complicated nature of Checkpoint's licensing process.

Since our firewall will also be our VPN device we must configure some encryption related parameters. For our site-to-site VPNs to our partners and suppliers we will be using the Isakamp/Oakley (IKE) encryption scheme with TripleDES with an MD5 hash. Compared to the other algorithms that Checkpoint supports TripleDES is more secure than DES, more supported than CAST and FWZ .



Note: FWZ is a proprietary algorithm, which means it was not open to public scrutiny and is only supported on Checkpoint products. We choose to use it because in some cases it is the only encryption scheme that will work with SecuRemote.

For Encryption domain we will specify the Corp_LAN. This effectively tells the firewall what potential networks will be on its side of any VPN. The actual access control for the specific VPNs will be handled by the specific rules we create later.

Using the steps above we will create the rest of the 'workstation' items with the following details:

- Partner_FW, Address= 111.111.111.1, Type=Gateway, External, IKE w/TripleDES and MD5, Encryption domain= Partner_net
- Supplier_FW, Address= 222.222.222.1, Type=Gateway, External, IKE w/TripleDES and MD5, Encryption domain= Supplier_net
- PSN_SMTP, Address 10.2.2.3, Type=Host, Internal, Nat= xxx.0.57.3
- PSN_ext_ext_DNS, Address= 10.2.2.4, Type=Host, Internal, Nat= xxx.0.57.4
- PSN_ext_int_DNS, Address= 10.2.2.5, Type=Host, Internal, Nat= xxx.0.57.5
- PSN_UML_DNS, Address= 10.2.2.6, Type=Host, Internal, Nat= xxx.0.57.6
- PSN_UML_WWW, Address= 10.2.2.7, Type=Host, Internal, Nat= xxx.0.57.7
- PSN_WWW, Address= 10.2.2.8, Type=Host, Internal, Nat= xxx.0.57.8
- Int_SMTP, Address= 10.1.1.2, Type=Host, Internal
- Int_Proxy, Address= 10.1.1.3, Type=Host, Internal
- Int_DNS, Address= 10.1.1.4, Type=Host, Internal
- Int_Ftp, Address= 10.1.1.5, Type=Host, Internal

The next type of object we create are those that represent the networks we will reference in our rulebase.

From within the 'Security Policy' gui, select Manage/Network Objects then New/Network. Use the following details:

- Corp_net; Network= 10.1.1.0 Mask= 255.255.255.0, Internal
- Partner_net, Network= 10.10.10.0, Mask 255.255.255.0, External
- Supplier_net, Network= 10.20.20.0, Mask 255.255.255.0, External

The firewall security policy consists of two components, the rules and the address translation. While the rules control access the address translation defines how packets will be NATted as they cross the firewall.

Rule creation on a Checkpoint firewall is a simple, graphical process. Selecting from our database of objects we previously created we select source, destination, service (tcp or udp port number), action (accept, drop, reject, encrypt, etc..) and logging.

Add the following rules:

© SANS Institute 2000 - 2002, Author retains full rights.

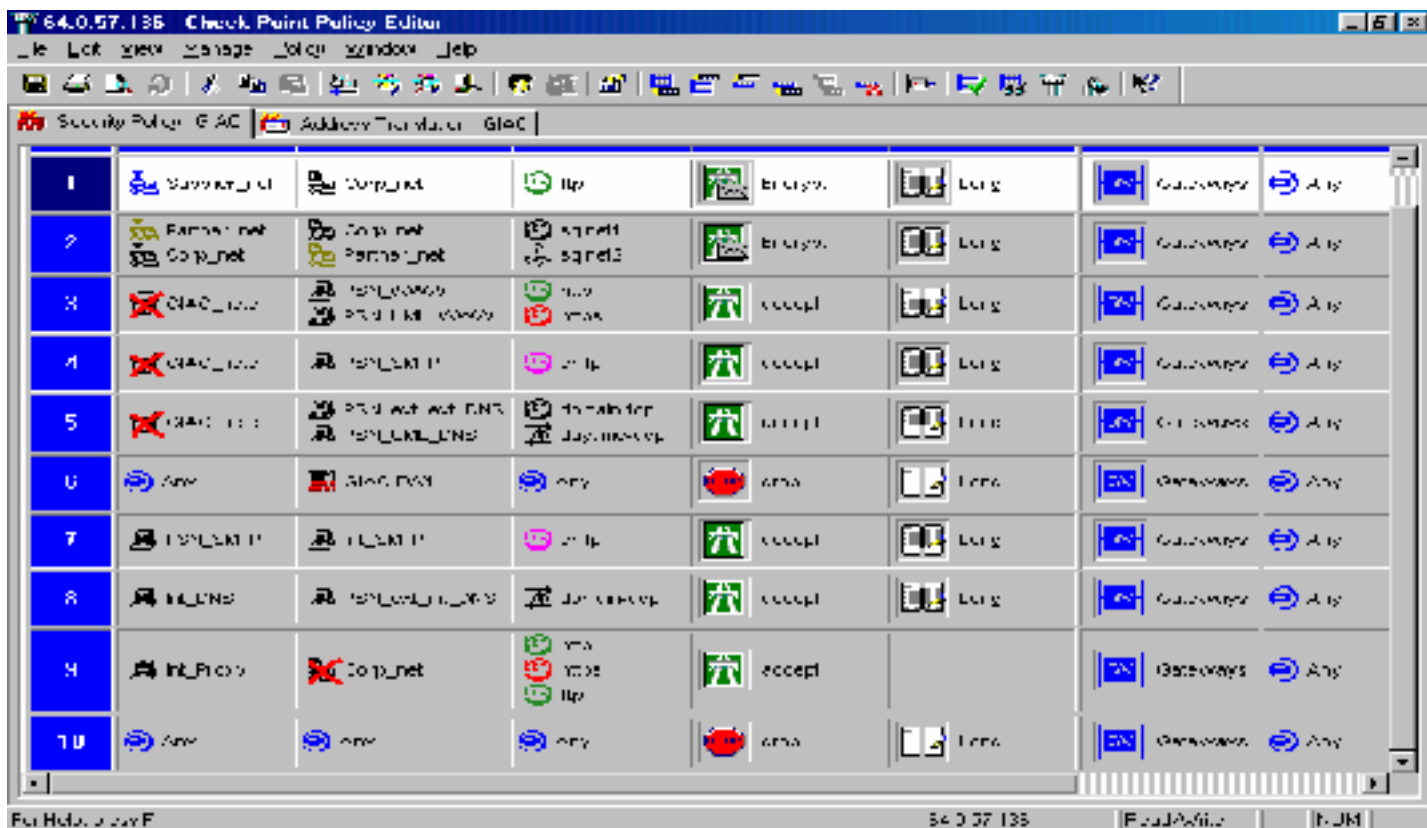
Firewall ruleset

Source	Destination	Service	Action	Track	Description
Supplier_net	Int_FTP	TCP 20 TCP 21	Encrypt*	Long	Allow suppliers to FTP files via VPN.
Partner_net Int_SQL	Int_SQL	TCP 1521 (Oracle SQL*Net Version 1)	Encrypt*	Long	Allow partners to connect to SQL servers via VPN.
Internet	PSN_UML_WWW PSN_WWW	Http Https	Allow	Long	Access to web servers.
Internet	PSN_SMTP	TCP 25	Allow	Long	Allow mail to relay.
Internet	PSN_UML_DNS PSN_Ext_ext_DNS	UDP 53	Allow	Long	Allow access to DNS. No zone transfers allowed.
UML_DNS Ext_ext_DNS Ext_int_DNS	Internet	UDP 53	Allow	***	Allow DNS servers to query the internet.
Any	GIAC_FW	Any	Drop	Long	Hide the firewall.
PSN_SMTP Int_SMTP	PSN_SMTP Int_SMTP	TCP 25	Allow	Long	Allow mail from relay to mail server.
Int_Proxy	Internet	Http Https Ftp	Allow	**	Allow outbound web & FTP traffic via proxy.
Int_DNS	Ext_int_DNS	UDP 53	Allow	***	Allow internal DNS to query ext-int DNS
Any	Any	Any	Drop	Long	Drop everything and log it.

* **Encrypt**- When we defined the firewalls we specified what encryption methods are supported. When we define the actual rules we define the specifics for the VPN.

** We will be performing logging of our internet traffic on the proxy server. Checkpoint logging isn't the greatest and besides our firewall is doing enough work already.

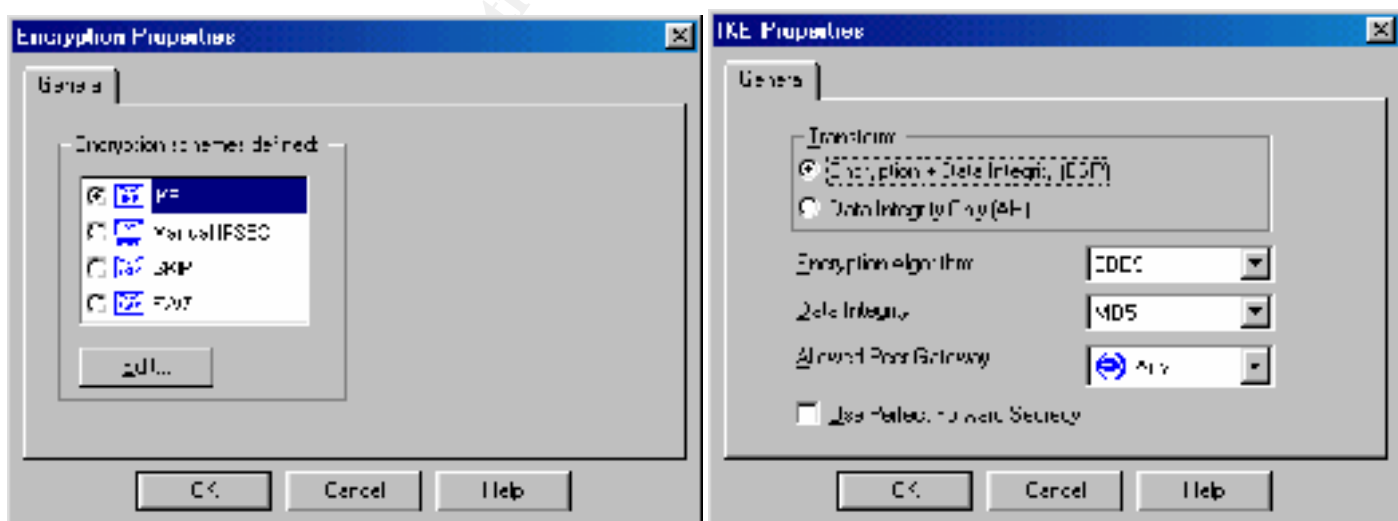
*** Our logs will grow quickly, logging our users DNS requests is just adding insult to injury.



The actual rulebase will look something like this when done.

Once the basic rules are in place we can move to the always exciting task of creating the actual VPNs.

Selecting the “[Encrypt](#)” in the action column for the first VPN we can edit the VPN’s properties.



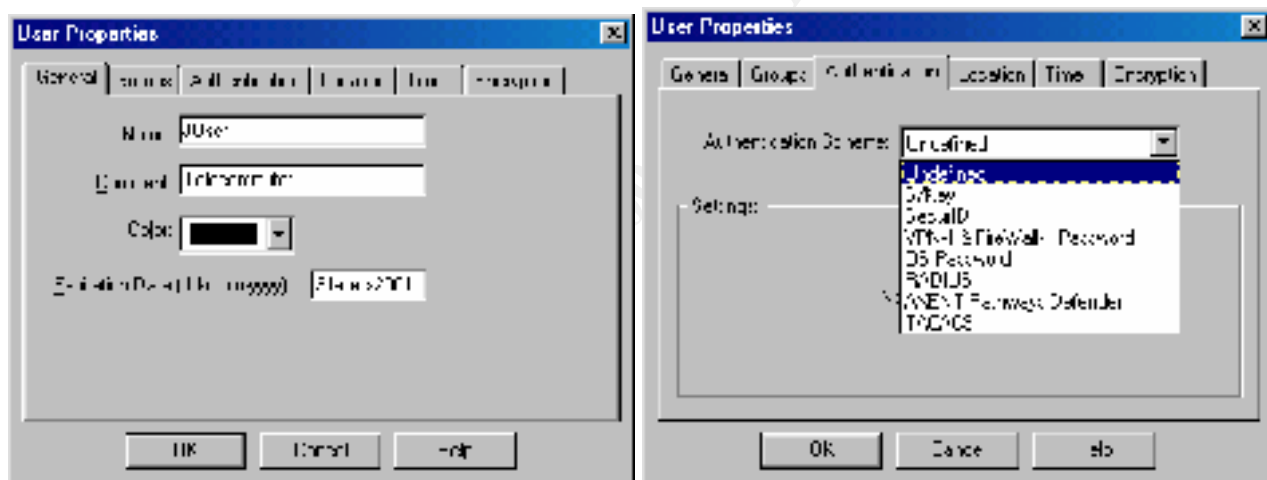
To complete the VPNs, the peer gateways (suppliers and partners) need to be configured with the same parameters. IKE w/ TripleDES and MD5. A shared secret also needs to be agreed upon. The actual secret is put into the workstation properties when the objects were created.

Telecommuters:

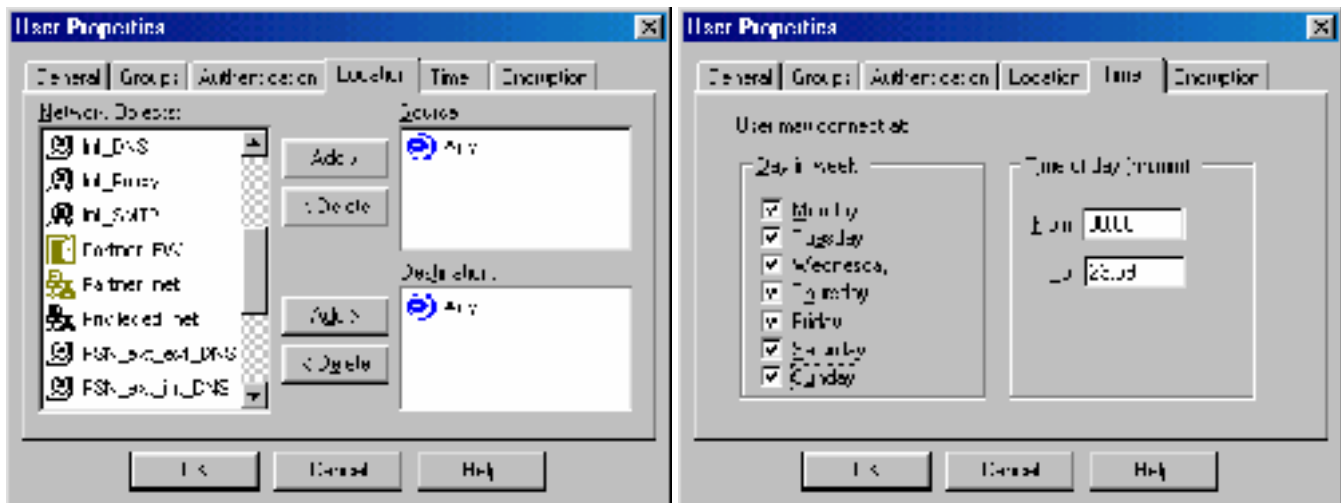
We will be using Secure Remote to facilitate our client to site VPNs. On the firewall side we need to...

- Create users
 - o Username
 - o Authentication method
- Create a Secure Remote rule

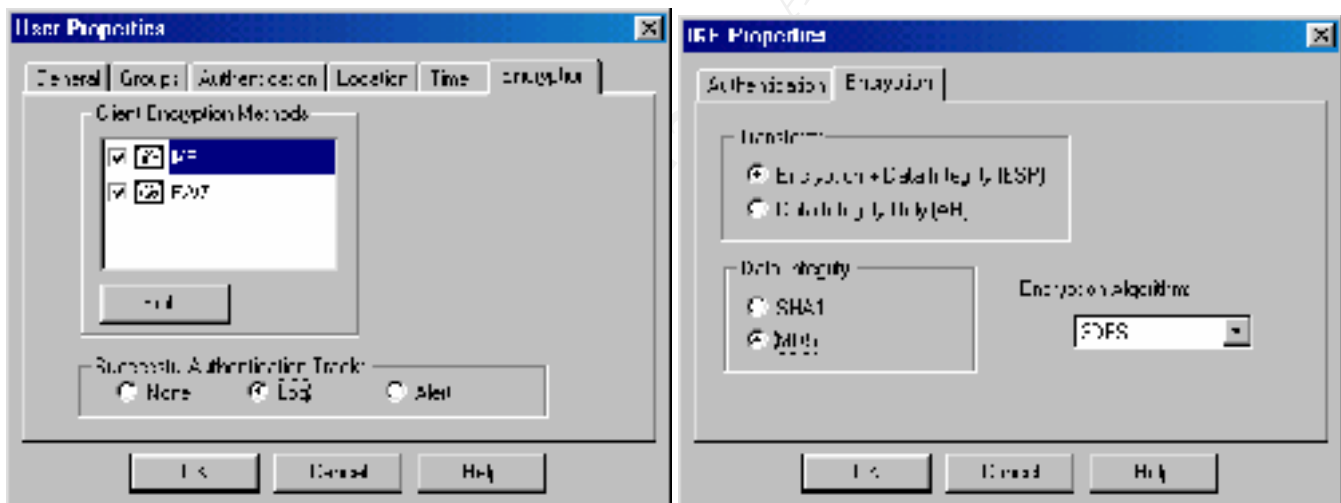
Similar to our previous creations of the other objects...



Give the user a name and expiration date. Specify an authentication method. Checkpoint supports several good options.



Specify a location for the user. Any / Any is usually what you want. You can also limit the times that a user can connect.



Select the encryption methods the user will support. While IKE is preferable, sometimes FWZ is all that will work. Remember, VPNs are a relatively new technology. As they mature they will be predictable and reliable, for now they are just plain buggy. Under each encryption method you can specify the type of transform and encryption algorithm.

For the actual VPN rule we set up something like this...

Source	Destination	Service	Action	Track	Description
Users@Any	Corp_net	Any	ClientEncryp	Long	Telecommuter VPN rule

Client setup:

Setting up Secure Remote on a client machine is so simple even a user can do it. The Secure Remote software can be downloaded from <http://www.checkpoint.com>. Essentially you run setup, choose to install on all adapters or just dial-up, and point it to the corporate firewall. To ease the already simple install you may want to publish a DNS record that resolves to the firewall. It will be far easier for the user to enter fw.giac-enterprises.com vs. xxx.0.57.1.

© SANS Institute 2000 - 2002, Author retains full rights.

Part III - Auditing the design

Does our security policy accomplish what we have designed it to? If we view the matrix we created earlier as our requirements for the policy, we can design our audit to test that these requirements are being met.

The tools we will employ to do our scanning are nmap, Cheops, firewalk & hping

While the tools are quite extraordinary they cannot audit this document. GIAC Enterprises is fictitious and the design is theoretical. In several cases we will create a mock-up of various parts of the architecture. Where available I will supply screen shots and/or simulated output to illustrate usage of the tools.

Nmap: (Available from <http://www.insecure.org/nmap/>)

Nmap will allow us to scan our perimeter for open tcp & udp ports. When scanning with nmap we will hit all 65,535 tcp and udp ports. A port is a port, well known or otherwise. Scans ran against our public network range (xxx.0.57.0/24) should only reveal the ports we left open by design.

The port scan:

`nmap -sS -P0 -p 1-65535 xxx.0.57.1/24` ← for the TCP port scan

`nmap -sU -P0 -p 1-65535 xxx.0.57.1/24` ← for the UDP port scan

Effectively we have told nmap to scan all ports on each of the addresses in our Class C range. The actual commands break down like this:

(The following was excerpted from the online nmap man page at http://www.insecure.org/nmap/nmap_manpage.html)

- sS** TCP SYN scan: This technique is often referred to as "half-open" scanning, because you don't open a full TCP connection.
- sU** UDP scans: This method is used to determine which UDP (User Datagram Protocol, RFC 768) ports are open on a host.
- P0** Do not try and ping hosts at all before scanning them. This allows the scanning of networks that don't allow ICMP echo requests (or responses) through their firewall.
- p <port ranges>** This option specifies what ports you want to specify.

If we view tcpdump output as we run our TCP portscan we can see that nmap is hard at work.

```
14:13:04.251703 eth0 > 192.168.168.24.37097 > xxx.0.57.2.47224: S
1992316254:1992316254(0) win 1024
14:13:04.251703 eth0 > 192.168.168.24.37097 > xxx.0.57.2.16679: S
1992316254:1992316254(0) win 1024
14:13:04.251703 eth0 > 192.168.168.24.37097 > xxx.0.57.2.38854: S
1992316254:1992316254(0) win 1024
14:13:04.261703 eth0 > 192.168.168.24.37097 > xxx.0.57.2.36970: S
1992316254:1992316254(0) win 1024
14:13:04.261703 eth0 > 192.168.168.24.37097 > xxx.0.57.2.56265: S
1992316254:1992316254(0) win 1024
14:13:04.261703 eth0 > 192.168.168.24.37097 > xxx.0.57.2.23605: S
1992316254:1992316254(0) win 1024
14:13:04.261703 eth0 > 192.168.168.24.37097 > xxx.0.57.2.36048: S
1992316254:1992316254(0) win 1024
14:13:04.261703 eth0 > 192.168.168.24.37097 > xxx.0.57.2.24229: S
1992316254:1992316254(0) win 1024
```

Note: As you can see, all the connection attempts originate from the same source port. This is nmap's default behavior. If we were conducting a true penetration test and wanted our scans to be a little less obvious, we could randomize the source ports and vary the time between scans.

A full scan of the entire Class C range of both the 65,535 TCP and UDP ports would take hours. Considering this is a fictitious company and a theoretical design actual testing is difficult. Using a small lab we can do an actual scan of one of the servers in our PSN. An example of the scan from the internet towards the PSN SMTP relay server yields:

```
Starting nmap V. 2.53 by fyodor@insecure.org ( www.insecure.org/nmap/ )
Interesting ports on (xxx.0.57.3):
(The 65535 ports scanned but not shown below are in state: filtered)
Port      State  Service
25/tcp    open   smtp
```

Nmap run completed -- 1 IP address (1 host up) scanned in 20090 seconds

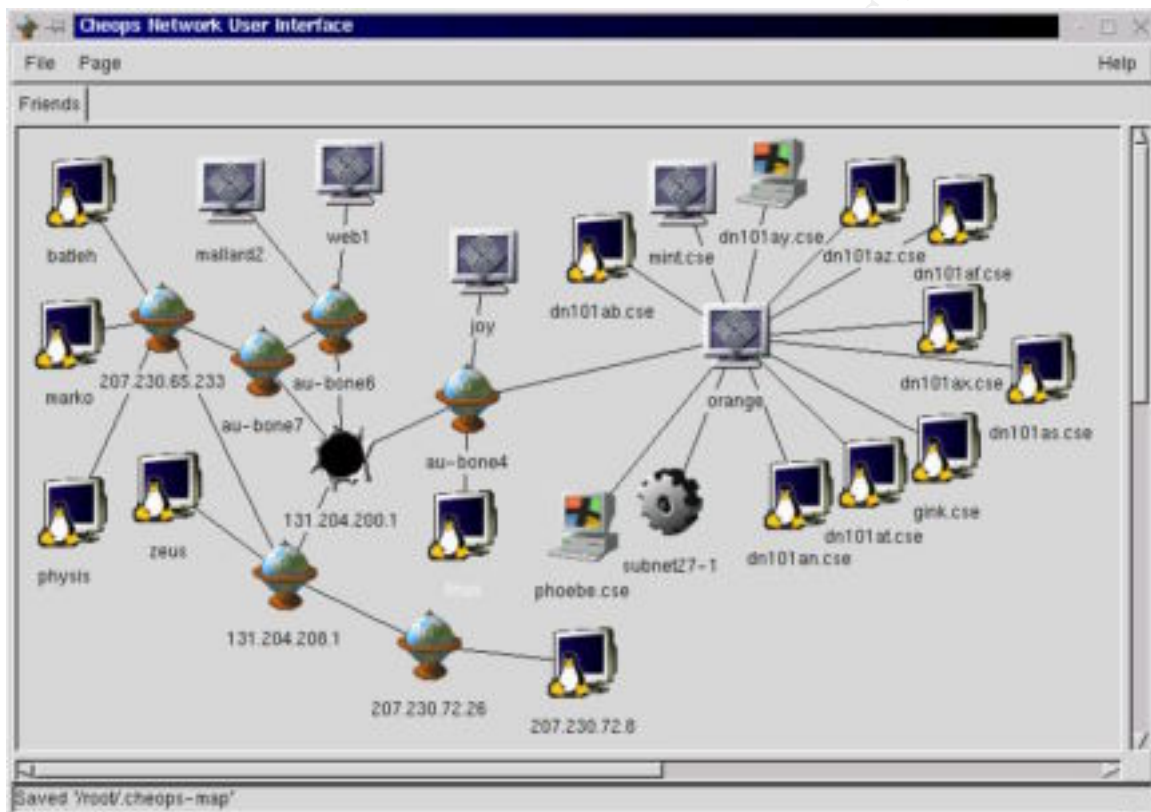
As expected only the SMTP port is "open" on host xxx.0.57.3 when viewed from the internet. If the actual design was implemented, we would run the scans

- From the internet against the Class C range (xxx.0.57.0/24).
- From the PSN against the Corp_net (10.1.1.0/24)
- From the Corp_net (10.1.1.0/24) against the PSN (10.2.2.0/24)
- From the Corp_net (10.1.1.0/24) against the firewall IP address.

Cheops

(Available from <http://www.marko.net/cheops/>)

Cheops will help us get a graphical representation of our network. Since we are auditing our own network we are not concerned with the less than subtle way cheops operates. According to the author, Cheops is the “Swiss army knife” of network tools. Cheops runs via a GUI. As seen below it does a nice little diagram for us. As pointed out earlier, our design is theoretical. While Cheops can do many things it cannot scan this document. In lieu of an actual scan, I have provided a screen-shot of Cheops at work. Our expectation would be that the actual scan would be able to map out the router, the firewall and the devices on our protected services network with accuracy. If a non-publicly device were discovered via this method we would have a serious concern.



Sample output from Cheops. This is NOT our network.

Firewalk: (http://packestorm.securify.org/UNIX/audit/firewalk/)

Firewalk is a tool that allows a person to map out a firewall’s ruleset. This is something that a potential intruder may use to enumerate our network. As a sanity check we will point firewalk at our perimeter and see how it interprets our filtering. At worst it will confirm our design at best it will not map accurately.

The concept behind firewalk is simply an extension on the traceroute program. Effectively, like traceroute, firewalk will send echo-requests towards a host using an increasing ttl count. Once the proper hop count is identified, the actual scan begins. Using

the proper ttl, TCP /UDP packets are sent to the desired ports. If the port is open a “time expired” will be allowed to return. If the port is closed then nothing returns. In our case we would use something similar to...

```
firewalk -n -P1-1024 -pTCP xxx.0.57.1 xxx.0.57.3
```

The above command tells firewalk to check for filtering to ports 1-1024 by the gateway at xxx.0.57.1 to the host xxx.0.57.3.

hping2:

(<http://www.kyuzz.org/antirez/hping/>)

hping2 is a network tool able to send custom ICMP/UDP/TCP packets and to display target replies like ping does with ICMP replies. Using hping2, you can perform spoofed port scanning. Using a command such as the one below we can test our router's anti-spoofing filters.

```
hping -a 192.168.168.5 -S -p 80 www.giac-enterprises.com
```

If all is well we would expect to see an event logged that looks something like this:

```
Feb 19 21:07:49 6X:router.giac-enterprises.com 5372: 14w5d: %SEC-  
6-IPACCESSLOGP: list 101 denied tcp 192.168.168.5(1052) -> xxx.0.57.8  
(80)
```

Explanation This message is logged when an IP packet is denied by the parameters you specified in the access list with the ID 101.

In conclusion...

The theoretical audit of our network design confirms our access matrix. The one concern we have is that everything we have done was done without privileged access. Any person with internet access can get the same tools and gather the same network info. The only effective way to prevent such mapping is to block ICMP from entering and leaving the network. Business needs prevent us from doing this.

Costs:

The time involved in such an audit would probably be two working days. The command line tools can be scripted and run overnight over the weekend. This would minimize the adverse affect the scans may have on network performance. Also there is always a chance that a simple scan will bring down a complicated server.

The cost of such a scan by a professional consultant would probably run about \$4-5,000. Alternatively, the knowledge required to perform such an audit can be acquired by attending a hands-on curriculum such as SANS Institute Track 4 – Incident Handling and Hacker Exploits. Even with travel expenses and course fees, the SANS conference would prove to be a better investment. The knowledge would be useful far beyond the initial audit.

Part IV – Design under fire

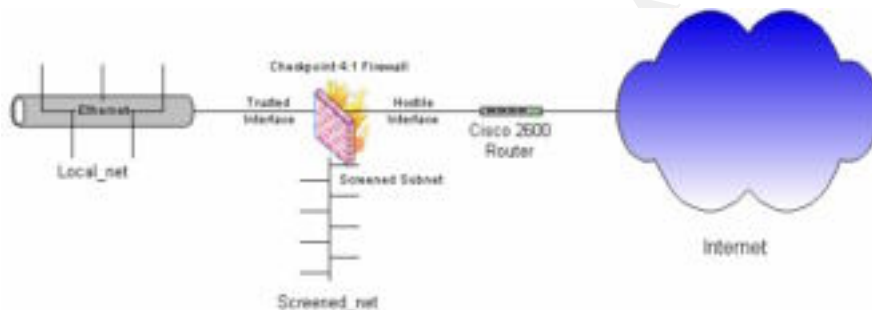
The fourth part of the assignment calls for attacking a design proposed by a previous GCFW candidate. For this section I have chosen Rick Dreger's design.

http://www.sans.org/y2k/practical/Rick_Dreger.doc .

Note: Ricks's practical concentrates on the firewall design. Specific references for mail, DNS, web and other servers do not exist. There is also no reference to actual IP addresses. I will go on the assumption that it represents the design for GIAC Enterprises and uses the same xxx.0.57.0/24 addressing that I used in my design.

Some specifics from Rick's practical:

Perimeter Design Diagram:



Perimeter Overview:

In order to implement perimeter security for this project the following choices were made:

- The firewall will be implemented using Checkpoint 4.1 running on a Nokia IP330.
- The firewall has three interfaces: one hostile (Internet) interface, one trusted (internal) interface, and one screened subnet interface.
- The firewall is sitting behind a Cisco 2600 router running IOS 12.x.
- The client is not using NAT and all local_net and screened_net IP addresses are Public.

Routing between the router and firewall will be done using static routes

My approach here will follow the typical steps a hacker might use to compromise a victim.

Recon:

`whois giac-enterprises.com` gets us the following nameservers


```
NS1.GIAC-ENTERPRISES.COM
NS2.GIAC-ENTERPRISES.COM
```

```
Nslookup
>server xxx.0.57.4
>set type=ANY
>ls giac-enterprises.com

www.giac-enterprises.com
mail.giac-enterprises.com
ns2.giac-enterprises.com
ftp.giac-enterprises.com
.
```

Pinging some of these names gets us the following info;

```
PING mail.giac-enterprises.com (xxx.0.57.3) from 192.168.168.5 : 56(84) bytes of data.
64 bytes from xxx.0.57.3: icmp_seq=0 ttl=242 time=512.574 msec
64 bytes from xxx.0.57.3: icmp_seq=1 ttl=242 time=529.756 msec
64 bytes from xxx.0.57.3: icmp_seq=2 ttl=242 time=529.754 msec
64 bytes from xxx.0.57.3: icmp_seq=3 ttl=242 time=539.748 msec
```

```
--- mail.giac-enterprises.com ping statistics ---
5 packets transmitted, 4 packets received, 20% packet loss
round-trip min/avg/max/mdev = 512.574/527.958/539.748/9.774 ms
```

A quick trip to www.arin.net and we find out that GIAC has the xxx.0.57.0 /24 network.

The Scan:

Using similar techniques to those we used earlier in our assessment we run nmap & Cheops against our network range. Since we are going against someone else's network, theoretically, we will modify our nmap scan a little.

```
nmap -sS -P0 -O xxx.0.57.2/24
```

A handy little feature of nmap is the OS fingerprinting option. With a simple -O switch nmap will perform a series of connection attempts to a host. When and how a machine responds tells a great deal about the running operating system.

```
Starting nmap V. 2.53 by fyodor@insecure.org (
www.insecure.org/nmap/ )
Interesting ports on (xxx.0.57.3):
(The 1521 ports scanned but not shown below are in state:
filtered)
Port      State      Service
25/tcp    open       smtp
```

```
TCP Sequence Prediction: Class=trivial time dependency
                        Difficulty=4 (Trivial joke)
Remote operating system guess: Nokia IPxxx running Checkpoint
Firewall-1

Nmap run completed -- 1 IP address (1 host up) scanned in 399
seconds
```

As we can see nmap found the open port for mail but mis-identified the OS as the firewall that was filtering the traffic. Not what we were looking for but useful nonetheless.

In addition to nmap we will also run Nessus against all the publicly available servers. Nessus is a free upgradeable remote vulnerability scanner. Using a library of known vulnerabilities (plug-ins) Nessus searches all specified ports for any of those goodies that hackers salivate over. For web servers specifically we will run a cgi-scanner such as, "whisker" by "rain forest puppy" (<http://www.wiretrip.net/rfp>).
or
"cis" by "mnemonix" (<http://www.cerberus-infosec.co.uk>)

Never underestimate the value of a simple *telnet* session. If we were to telnet to the mail and web servers on the related ports we would have seen something like..

```
Telnet mail.giac-enterprises.com 25

220 mail.giac-enterprises.com ESMTP Server (Microsoft Exchange
Internet Mail Service 5.5.2653.13) ready
```

Exploit:

So what did we find out so far?

- Checkpoint Firewall-1 running on a Nokia Ipxxx box. (from nmap)
- Apache web server (from Nessus)
- Microsoft Exchange 5.5 sp3 mail server running on NT4.0 (from Telnet)
- An FTP server, Linux box (from Telnet)

To the web we go. <http://www.securityfocus.com/> and we do a search for the systems that we have discovered.

- Checkpoint / Nokia
 - o Check Point Firewall-1 4.1 Denial of Service Vulnerability at <http://www.securityfocus.com/bid/2238> tells of a DoS where Checkpoint may stop passing traffic if it perceives too many users behind it. When you license the Firewall-1 you specify a user count. If the firewall hears too many unique

addresses it will stop passing traffic. This attack needs to be initiated by an internal host.

- [Nokia IP440 Buffer Overflow Vulnerability](http://www.securityfocus.com/bid/2054) at <http://www.securityfocus.com/bid/2054> tells of a buffer overflow of a Nokia box. If a URL is sent to the device's administration interface which contains a large number of characters it can overflow the relevant buffer and create a segmentation fault. As with any buffer overflow, this has the potential to allow arbitrary code execution, but this result has not been reported in this case.
- [Multiple Firewall Vendor FTP Server Vulnerability](http://www.securityfocus.com/bid/979) at <http://www.securityfocus.com/bid/979> tells of a vulnerability whereby the firewall has support for passive FTP connections. In short, if a network has an FTP server accessible behind a FireWall-1 firewall, that they allow the outside world access to, it may be possible for an attacker to open TCP connections to certain ports on that FTP machine.
- Apache web server
 - Many script related vulnerabilities were found. Our earlier scans with whisker and cis already pointed out the presence of such exploitable scripts.
- Exchange server
 - As with the web server there are numerous vulnerabilities that exist with the mail server. An interesting one is [Microsoft Outlook / Exchange Blank Headers DoS Vulnerability](http://www.securityfocus.com/bid/1333) at <http://www.securityfocus.com/bid/1333> . Microsoft Outlook and Exchange are both vulnerable to denial of service attacks through incoming email if both bcc: and Reply-to: or Return-Path: and From: fields are left blank. Outlook will crash upon the delivery of these particular email messages and Exchange will produce an error stating that the message is not deliverable and to check for sufficient memory or disk space.

Note: This listing could go on and on. The number of relevant vulnerabilities we were able to turn up with a 5 minute search of one security related site was overwhelming. It is not hard to see why 'script kiddies' are a dime a dozen.

First we will attempt to compromise the firewall itself.

Perhaps a Trojan hidden in a holiday greeting card can initiate a series of pings with spoofed addresses directed at the internal interface of the firewall. Since Checkpoint will add each new IP address to a file we do not need to get greedy with the pings. We can run them slowly over a series of days. Once the number of unique addresses exceeds the license count, traffic stops.

Maybe we can use the buffer overflow weakness to actually gain privileged access to the box. While only authenticated users can perform this attack, a firewall administrator account with even minimal permissions, is all that is necessary. A malformed url pointed at the firewalls administrative port and shell access perhaps?

If our intentions are just to bring down the network we have other options. Perhaps we use similar reconnaissance techniques on an ISP. A few Arin searches and we may discover the address range that a DSL provider uses for its clients. We run a port scan on the range and discover a series of machines that are always connected and never

protected. (hey it rhymes I like that, I may use that again) Maybe we get a couple of zombies placed in the startup folder of some of those unsuspecting users. They reboot and boom they're mine.

Now that I have some loyal followers I can launch a Distributed Denial of Service attack. Perhaps an ICMP flood using Tribal Flood Network (<http://packetstorm.securify.com/distributed/>) originating from spoofed yet legal addresses and directed at the router or firewall. Bandwidth quickly gets saturated or resources quickly get exhausted waiting for responses that never come.

Defending against something like this is difficult. Keeping system patches up to date can limit the chance that a malformed packet can cause the DoS but preventing a depletion of resources is a bit more problematic. Redundant links, redundant firewalls, redundant routers, BGP, load balancers and heavy prayer and perhaps you can reduce the probability that it will happen to your site. Making use of a co-location facility for key servers is one way of getting access to resources you may not be able to afford otherwise.

Ultimately our goal would be to compromise an internal system and possibly divulge proprietary information. Recently there have been several high profile compromises of corporate networks. Anecdotal evidence point to lax security for telecommuters as being the primary way attackers gained access. Orchestrating such an attack is the work of an experienced predator. Fortunately for us we found a better way.

From Rich's practical..

Original Recommendation:

Block "spoofed" addresses-- packets coming from outside your company sourced from internal addresses or private (RFC1918 and network 127) addresses. Also block source routed packets.

While this was his intention if we look at the actual ACL on the router where the anti-spoof filtering is taking place...

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

```
no ip source-route
```

... we see that Rich did not do accomplish what he set out to. Rich had stated that NAT was not being employed in this design. It is possible that the administrator had trouble with subnetting and could not figure how to break up his Class C without using NAT for the internal net. Whatever the reason, this could be our in.

Packets could be crafted that would spoof an internal address and get through the firewall. While we wouldn't see the response to these spoofed packets we may be able to initiate an outbound connection on an open port. Working blindly we may be able to tftp netcat (<http://www.l0pht.com/~weld/netcat/>) to the box. We could then have netcat dump the password file via an open outbound port to a waiting server on the outside.

With the password file in hand we perform an offline crack and now have the keys to the kingdom.

Keep Access:

Once access is achieved we will take steps to insure we don't lose it. Add a user, add a service that initiates an outbound connection or perhaps that admin password you cracked also works on the firewall.

Note: The accounts that you use to manage your perimeter devices should be unique and exclusive to that device only.

Covering the tracks:

Modify the access logs and if you were quiet enough maybe nobody will ever know you were there.

© SANS Institute 2000 - 2002, Author retains full rights

References

RFC 1918

<http://www.ietf.org/rfc/rfc1918.txt>

Defenses Against Distributed Denial of Service Attacks by Gary C. Kessler

<http://www.sans.org/infosecFAQ/threats/DDoS.htm>

Phoneboy

<http://www.phoneboy.com>

Top Ten Blocking Recommendations Using Cisco ACLs Securing the Perimeter with Cisco IOS 12 Routers by Scott Winters

http://www.sans.org/infosecFAQ/firewall/blocking_cisco.htm

<http://www.packetfactory.net/Projects/Firewalk/firewalk-final.html>

<http://www.cisco.com/warp/public/707/21.html>

<http://www.nessus.org/>

<http://www.securityfocus.com>

<http://www.marko.net/cheops/>

© SANS Institute 2000 - 2002, Author retains full rights.