Global Information Assurance Certification Paper

Firewalls, Perimeter Protection, and VPNs
GCFW Practical Assignment
Version 1.5d

Ken Colson

Assignment 1 - Security Architecture (25 Points)
Define a security architecture for GIAC Enterprises, a growing Internet startup that expects to earn $200 million per year in online sales of fortune cookie sayings, and which has just completed a merger/acquisition. Your architecture must specify filtering routers, firewalls, VPNs to partners, secure remote access, and internal firewalls. Be explicit about the brand and version of each perimeter defense component. Produce a diagram or set of diagrams with explanatory text that define how to use perimeter technologies to implement your security architecture.
You must consider and define access for:

- Customers (the companies that purchase bulk online fortunes);
- Suppliers (the authors of fortune cookie sayings that connect to supply fortunes);
- Partners (the international partners that translate and resell fortunes).

The Proposed
GIAC Network Design

Figure 1

The GIAC Network

Routers

I have designed the GIAC Enterprise network to use a single Cisco 2600 router that will be connected to 2 different Internet Service Providers (ISPs). By connecting the router to 2 completely separate Internet connections, we will have a more reliable Internet presence. Our Internet presence will be more reliable

since we will not be subject to connectivity outages by a single ISP and we will also be more resilient to Denial of Service attacks than if we just had a single connection to the Internet. For example, if an attack is originating from one of the ISP's connections we can sever that connection while the attack is going on and still have Internet connectivity via the other ISP's connection.

The router will also be configured to block some of the basic trouble ports and services as prescribed by SANS. The router will be configured to log messages to the internal syslog server, so that routine monitoring can be accomplished without have to actually connect to the router.

External Firewall

The external firewall will be a Linux machine configured with the latest stable kernel and all of the needed security patches as listed in the vendor's support site. A Linux firewall will provide for an inexpensive, yet very flexible firewall that does not require a huge up-front investment and also is not locked into a proprietary solution. The Linux community is constantly assessing the security of the kernel and the services that run on the Linux platform. Instead of relying on vendors to provide timely patches, which can be a problem from time to time depending on the vendor, GIAC Enterprises will be responsible for their own security updates and patches. By paying close attention to security bulletins from SANS and CERT, GIAC Enterprises will be able to respond to known and new security problems with in house staff.

VPN

A Cisco VPN 5000 will be placed in parallel with the external firewall. The Cisco VPN will be used to establish an IPSec VPN between GIAC Enterprises and our trusted partners. The Cisco VPN 5000 supports a wide variety of clients including Linux, Solaris and Windows. For individual access into GIAC private network resources, we will employ the RSA SecureID authentication. This 2 factor authentication will ensure that the clients connecting into our VPN know their username and password and are in possession of the SecureID authenticator.

Internal Firewall

The internal firewall is going to be a Linux based firewall, but unlike the external firewall it will be configured to allow outgoing services, but very limited inbound services. The internal firewall of course will be hardened in the same manner described for the external firewall.

The internal firewall is configured to allow common outbound services such as HTTP, HTTPS, SMTP, DNS and other services required for GIAC Enterprise employees to perform their daily business.

Publicly Accessible Machines
Although not directly part of the security scheme, each one of the public machines must be installed as securely as possible, since they will be the most likely targets of attacks. All of the machines must be configured to perform remote logging via syslog to a machine that is located behind the internal firewall.

The base install of each of these machines, will be the stripped down install of Linux, exactly like the installation of the external firewall. After a hardened install of each of these machines, the applications that they need to perform their tasks will be installed. The best accepted practices for installing a web server, nameserver and mailserver will be followed. For example, our nameserver install will employ the use of a 'spilt-horizon', meaning that the external nameserver will not have the internal addresses of our private network. With our mailserver install we will pay close attention to the latest Sendmail versions and configuration options to insure the more secure installation possible. The router and the firewall cannot secure a faulty setup of any publicly accessible machine.

Intrusion Detection
Network intrusion detection must be performed to monitor all of network traffic that has been allowed into the screened and private networks. I decided to place the intrusion detection device in the screened and private networks of GIAC Enterprises. With this configuration the device will be able to monitor traffic on either side of our internal firewall.

The intrusion detection device that I have chosen to employ is the NFR Network Intrusion Detection (NID). The NFR NID can monitor for known attacks, abnormal behavior and policy infringement. Abnormal traffic is logged and notifications can be configured for suspicious events.

Syslog server
The syslog server is maintained on the private network to monitor public machines. A log checker such as Logcheck or Swatch can optionally be installed in this machine. No casual user accounts will exist on this machine. The logs can be stored on read only media on a periodic basis to provide a permanent audit trail of activity. This machine will also be installed with a hardened Linux install.

Conclusion
Each piece of the GIAC network is crucial to the security of our network. Keeping in mind that the network is only as secure as its weakest link, I have tried to identify all areas of vulnerabilities and address each one. During the auditing phase of our network installation, we will verify that all of the pieces are in fact working together to secure the GIAC Enterprises network.

Assignment 2 - Security Policy (25 Points)
Based on the security architecture that you defined in Assignment 1, provide a security policy for AT LEAST the following three components:

- Border Router
- Primary Firewall
- VPN

You may also wish to include one or more internal firewalls used to implement defense in depth or to separate business functions.

Border Routers

Physical Security

We are going to place the router in a physically secure location. The router is the heart of our network and we cannot assume that everyone that is employed at GIAC enterprises will treat the router with the respect that it deserves. This physical access should mean that the router is placed behind locked doors of some type and that only a limited amount of trusted and trained personnel have access into this space. The inadvertent disconnection of the network by cleaning personnel or a misguided technician is just as disastrous as a well planned and executed Internet attack.

Access into the router

We will need to limit inbound access into the router over the network. Since changes to the router will not have to occur a regular basis, we are going to disable any type of network access into the router and allow access in only via the console.

We will begin securing our router by setting up a password for console login:

```
Line console 0
Login
Password “xxxxxxxx”
```

“xxxxxxxx” represent the well chosen password for the router.

We are going to also configure some other basic security options, such as a session timeout for 2 minutes and password encryption with the following commands:

```
exec-timeout 2
service password-encryption
enable secret
```

Now we log into the router and provide the password that we just configured.

```
User Access Verification
```

```
Password:xxxxxxxx
Router>
```

Now we are going to enable the password for 'privileged' mode by entering the command:

```
Router>enable-password "xxxxxxxx"
```

Now to access any of the privileged commands you will need to use the password that we just created. To enter the privileged mode we enter the commands:

```
router>enable
Password:
router#
```

The prompt now changes, showing that you are now in privileged mode

Now we are going to stop vty telnet access into the router.

```
router#access-list 1 deny any
router#line vty0 4
router#access-class 1 in
```

and stop telnet into the aux ports on the router

```
router#line aux 0
router#access-group 1 in
```

Block unneeded and dangerous ports and services

With the physical access and login access to the router secured we are going to restrict the traffic that the router is going to allow into our network.

I will list each of the rules that we are going to apply to the router and then explain the reason for blocking each of the ports and or services. Not all of the ports that are recommended to be blocked by SANS will be blocked at the router, they will be blocked by the firewall instead.

```
router>no ip source-route
```

Explanation: we will stop source routed packets at the router

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

Explanation: this will block access to several unneeded services such as echo, chargen

```
router>no service finger
```

Explanation: this will stop the 'finger' service that could provide attackers with information concerning network information.

Some of the other types of packets that we are going to block at the router will require the use of a standard access-list.

```
router>access-list 11 deny 10.0.0.0       0.255.255.255
router>access-list 11 deny 127.0.0.0      0.255.255.255
router>access-list 11 deny 172.16.0.0     0.15.255.255
router>access-list 11 deny 192.168.0.0    0.0.255.255
router>access-list 11 deny host 0.0.0.0
```

Explanation: This access-list that we just created will stop packets at the border that could not have originated from outside of the router. These would be considered 'spoofed' packets.

We are also going to block some of the ICMP messages that are being sent towards our router and keep them from entering our network

```
router>access-list 11 deny icmp any any echo
router>access-list 11 deny icmp any any echo-reply
router>access-list 11 deny icmp any any time-exceeded
router>access-list 11 deny icmp any any host-unreachable
```

Now that we have stopped some of the more dangerous and annoying type of Internet traffic at the router we need to let the rest of the traffic in and let the firewall do the rest of the work

```
router>access-list 11 permit tcp any any
router>access-list 11 permit udp any any
```

Now we have defined all of our rules we will assign then to the interface

```
router>interface serial 0/0
router>ip access-group 11 in
```

These same exact rules would have to be defined for the other ISP connection as well.

```
router>interface serial 0/1
router>ip access-group 11 in
```

External Firewall

Securing a Linux machine
Before we apply any type of firewall rules or even start thinking about the rules, the machines that we are going to use need be secured. Just as we secured access to the router we are going to secure the Linux machines.

[**Note:** All of the steps that are described here to secure our Linux firewalls will also used to secure all of the publicly accessible machines as well as the syslog server.]

Although securing a Linux host is beyond the scope of this document, I will list the basic steps that I have taken to secure the machines.

Step 1 – Install Base OS

We are going to install the machine from a vendor supplied distribution, preferably from a CD. We want to install from a CD, so that network connectivity is not enabled until we are ready for the machine to be on the network.

Step 2 – Apply OS Patches

After we have installed the OS we are going to check the vendor's site for a list of security patches that we need to install for our OS version. Check the signature and checksum of each of these patches prior to installation. Obtain any security patches that the vendor recommends or requires. Put these patches on a CD, do not apply them over the network.

Step 3 – Compile a new Kernel

The kernels that are provided on vendor's sites and distributions are generally unsuitable for use as a firewall machine. The kernels will may not allow for firewall rules or routing or they will load any and all kernel modules that are requested. We will compile our own kernel to better suit our needs. Obtain the latest stable Linux kernel.

Step 4 – Install security applications
We are going to install the latest versions of these pieces software.

- Bastille OS hardening
- openSSH secure login
- logcheck automating log checking/parsing
- Tripwire file integrity

Some of these applications are provided with some vendor's distributions. Install as needed. Install any other applications that are needed for security that are not included on the above list. Only install applications that are needed. Install Tripwire last so that it can obtain a checksum for all of the applications that are installed.

Step 5 – Remove all unneeded services from startup scripts

Remove all unneeded services such as inetd and NFS, named, etc. Configure openSSH to run on the machine at startup.

By following these steps, in this order, we now have a machine that has never been exposed to any network traffic and has a high quality checksum of every installed package and a way to verify and notify an administrator of any changes. With these basic security pieces installed we can now enable the Ethernet cards on the machine and configure the rules that we need for GIAC Enterprises.

The Firewall Rules

We are going to start by blocking everything and setting the 'policies' for each of the possible rules, the Input, the Output, the Forward. It is much safer to deny everything and allow only the specific services that we want than to try to think of everything that we may need to block.

[**Note:** In the start up files for a Linux machine we will have to enable the Ethernet cards before we apply any rules to them, so in the startup files we are going to start the Ethernet cards and *immediately* start the firewall rules. We will not configure the Ethernet cards to route by default, we will turn on the routing in our firewall script after some of the basic deny rules are in place)

The default policy will be to block everything:

```
ipchains -F
ipchains -P input DENY
ipchains -P output REJECT
ipchains -P forward DENY
```

This is the best policy to follow, we can always add rules to allow for extra needed services, but we can't be expected to know everything to block with all of the rules that it would take to think of everything. This is the safest starting stance for any firewall.

We are going to block some odd packets:

```
# tcp syn cookie protection
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

Explanation:
By enabling this kernel module, we will afford ourselves some protection from a SYN flood. The technical details of just how this works are beyond the scope of this document, but this module will apparently allow the servers to continue to respond even when the incoming queue is full.

```
# defrag
echo 1 > /proc/sys/net/ipv4/ip_always_defrag
```

Explanation:
We want to reassemble any packets that are fragmented in order to inspect them as a whole, not as fragments. Only when the packets are reassembled can all of the rules be applied to them. This will also stop an attacker from being able to overlap packets to obscure the true intent of the packet.

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

Explanation:
We do not want to be a bad neighbor and respond to a ICMP broadcast. This is referred to as a Smurf attack and could possibly lead to a Denial of Service attack being run against GIAC Enterprises.

```
# bad error message protection
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
```

Explanation:
The firewall is not going to accept any ICMP traffic unless specifically allowed. We want to ignore any unsolicited messages.

```
# spoofing protection
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
        echo 1 > $f
done
```

Explanation:
This will allow for the routing to drop all packets that “look as sourced at a directly connected interface, but were input from another interface”

```
#disable icmp redirect acceptance
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
        echo 0 > $f
done
```

Explanation:

Under no conditions will we allow any ICMP redirects to our firewall. All of the routes that we are going to use on the firewall will be static routes. ICMP redirects could possibly allow a hacker to fool our system into believing that a remote address is actually a local address or could send all of our traffic into a black hole.

```
#disable source routed packets
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
        echo 0 > $f
done
```

Explanation:
Very few if any legitimate uses exist for source-routed packets and we certainly don't want to allow any into GIAC Enterprises. Source routed packets could possibly fool our firewalls and routers into believing that the packet has originated locally.

```
#log spoofed packets, source routed or redirected
for f in /proc/sys/net/ipv4/conf/*/log_martians; do
        echo 0 > $f
done
```

Explanation:
This will log addresses that have impossible and/or illegal addresses. We need to know of any bogus address that have made their way into the GIAC network.

```
# enable ip forwarding
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Explanation:
Finally we turn on the routing on the firewall to allow outbound packets access to the Internet. Although we have enable routing, no packets are being sent through the machine because of our deny all policy. Now we have to allow access for the only the services that we want to let in and out of our network.

Allowing inbound packets
For some of the publicly accessible machines that live behind the external firewall we need to allow for inbound connections. The rules and their application described are for the purpose illustrating how to correctly allow inbound traffic to these machines. The rules once defined can be applied to any type of service that is allowed in from the Internet, simply changing the source and destination ports to suit the desired service.

Before we add any rules we are going to define a few variables so that the rules read a little more like English. I do like keeping the actual ports numeric for ease of searching and locating when I am editing the rules.

The variable definitions:

```
EXT_INTERFACE=”eth0”
INT_INTERFACE=”eth1”
ANYWHERE=”any/0”
PRIVPORTS=”0:1023”
UNPRIVPORTS=”1024:65535”

OUR_NAMESERVER=”xxx.xxx.xxx.xxx”
OUR_MAILSERVER=”xxx.xxx.xxx.xxx”
OUR_WEBSERVER=”xxx.xxx.xxx.xxx”
```

All of the rules will follow the same basic syntax, first the rule to allow inbound connections and then the rule to allow for the response back from our machines.

Notice that on each rule the first rule that allows for the inbound connection does not check the status of the TCP flags (the –y), but the second rule only allows for our machine to respond to an inbound request (!-y means the ACK must be set)

Web Access
We are going to allow inbound access to our webserver for HTTP and HTTPS

```
ipchains -A input -i $EXT_INTERFACE -p tcp \
        -s $ANYWHERE $UNPRIVPORTS -d $OUR_WEBSERVER 80 -j ACCEPT

ipchains -A output -i $EXT_INTERFACE -p tcp ! -y \
        -s $OUR_WEBSERVER 80 -d $ANYWHERE $UNPRIVPORTS -j ACCEPT
```

Rule	HTTP
Source IP	Anywhere
Source Port	Unprivileged ports
Destination IP	External address assigned to our Web server
Destination Port	80 (HTTP)
Protocol	TCP
Description	This will allow inbound connections to the firewall on Port 80 that are destined for our webserver

We are going to allow HTTPS into our network as well, the rules are applied exactly the same with the only difference being the destination port on the request and the destination port on the reply.

```
ipchains -A input -i $EXT_INTERFACE -p tcp \
        -s $ANYWHERE $UNPRIVPORTS -d $OUR_WEBSERVER 443 -j ACCEPT
```

```
ipchains -A output -i $EXT_INTERFACE -p tcp ! -y \
        -s $OUR_WEBSERVER 443 -d $ANYWHERE $UNPRIVPORTS -j ACCEPT
```

DNS

All inbound name service requests are allowed in to the nameserver, we are only going to allow for inbound queries, no inbound TCP connections will be allowed on port 53

```
ipchains -A input -i $EXT_INTERFACE -p udp -s $ANYWHERE $UNPRIVPORTS \
        -d $OUR_NAMESERVER 53 -j ACCEPT

ipchains -A output -i $EXT_INTERFACE -p udp -s $OUR_NAMESERVER 53 \
        -d $ANYWHERE $UNPRIVPORTS -j ACCEPT
```

Rule	Nameservice
Source IP	Anywhere
Source Port	Any unprivileged port
Destination IP	The address assigned to our nameserver
Destination Port	53
Protocol	UDP
Description	Any inbound request for the nameserver is accepted

SMTP Mail

We need to make a rule that will allow for inbound SMTP mail

```
ipchains -A input -i $EXT_INTERFACE -p tcp \
        -s $ANYWHERE $UNPRIVPORTS \
        -d $OUR_MAILSERVER $ANYWHERE 25 -j ACCEPT

ipchains -A output -i $EXT_INTERFACE -p tcp ! -y \
        -s $OUR_MAILSERVER 25 \
        -d $ANYWHERE $UNPRIVPORTS -j ACCEPT
```

Rule	Mail Incoming
Source IP	Anywhere
Source Port	Any unprivileged port
Destination IP	The external address assigned to our mailserver
Destination	25

Port	
Protocol	TCP
Description	Any inbound request is accepted for the mailserver

This rule will allow for our mailserver to send outbound mail

```
ipchains -A input -i $EXT_INTERFACE -p tcp ! -y \
        -s $ANYWHERE 25 \
        -d $OUR_MAILSERVER $UNPRIVPORTS -j ACCEPT

ipchains -A output -i $EXT_INTERFACE -p tcp \
        -s $OUR_MAILSERVER $UNPRIVPORTS \
        -d $ANYWHERE 25 -j ACCEPT
```

Rule	Mail outgoing
Source IP	Our mailserver address
Source Port	Any unprivileged port
Destination IP	Anywhere
Destination Port	25
Protocol	TCP
Description	Any outbound request from our mailserver is allowed out.

The final rule is the rule to deny everything and log the packets. With this final rule in place you can watch the log messages to help troubleshoot the rules that you have written, as well as detect possible attacks.

```
ipchains -A input -i $EXT_INTERFACE -j REJECT -l
ipchains -A output -i $EXT_INTERFACE -j REJECT -l
ipchains -A forward -i $EXT_INTERFACE -j REJECT -l
```

Internal Firewall

The same rules concerning spoofed packets and the like are loaded (see External firewall) and the base policy is to deny everything. Now we need to configure the outbound services that we need and the limited inbound services.

There are no inbound connections allowed into the private GIAC Enterprise network, except for the traffic destined for the syslog server. Services will have to be allowed out of the firewall, such as web traffic and nameservice requests. The firewall will also perform IP masquerading. Other than the IP masquerading, the syntax of the firewall rules will follow that style of the rules above. [Note: I

am not going to list all of the outbound traffic from the internal network. Suffice to say that the actual traffic should be configured on an as needed basis and that inbound services should not be allowed.]

The only rule that is needed is to allow out any traffic that has originated on our network and apply the IP masquerading rule:

Allow our internal traffic out:

```
ipchains –A input –i $INT_INTERFACE –d $ANYWHERE –j ACCEPT
ipchains –A output –i $INT_INTERFACE –d $ANYWHERE –j ACCEPT
```

Now the masquerading rule:

```
ipchains –A forward –i $EXT_INTERFACE –s $INT_NETWORK –j MASQ
```

Now all of the traffic from the internal network will be allowed out and will have the source address of the external interface of the internal firewall. Although this has allowed for any type of outbound traffic, a rule still needs to be created to allow an outbound service. This is because there is not a rule that allows for the return of any packets.

```
ipchains -A input -i $EXT_INTERFACE -p udp -s $ANYWHERE $UNPRIVPORTS \
        -d $OUR_NAMESERVER 53 -j ACCEPT
```

Now we are going to configure the inbound syslog traffic:

In order to allow traffic through the internal firewall from the screened network into the private network, we will have to use a kernel module that will allow for 'port forwarding'. This will allow for the syslog messages in the screened network to be directed to the external interface of the internal firewall and then be directed to a machine that has a private internal address. [This ability will have to be compiled into the kernel.]

```
ipmasqadm portfw -a -P upd -L $OUR_EXTERNAL_ADDDRESS 514 –R \
$OUR_SYSLOG 514
```

Even though we are forwarding this packet through the firewall, we still need to make a rule that will allow it to pass:

```
ipchains -A input -i $EXT_INTERFACE -p udp -s $OUR_MACHINES 514 \
        -d $OUR_SYSLOG 514 -j ACCEPT
```

With this rule, I would configure it to only accept connections from our machines that we are expecting to receive syslog messages from. The rule of course would be exactly the same, except for the source port of the packet.

VPN

Trusted partners are going to be allowed to connect to internal GIAC Enterprises resources via an IPSec VPN. We will setup a tunnel for our partners and also inbound secure connections for individual access into our network.

Basic setup of the Cisco VPN 5000
First we are going to change the default passwords that come with the VPN device.

```
VPN>configure General
VPN>Password = "xxxxxxxx"
VPN>EnablePassword = "xxxxxxxx"
```

Now are going to change the logging of the VPN to go to the internal syslog server that we setup for monitoring.

```
VPN>LogToSyslog = On
VPN>SyslogIPAddress = xxx.xxx.xxx.xxx
```

xxx.xxx.xxx.xxx is of course the IP address of internal syslog machine.

Trusted Partner Sites
VPN access into the GIAC Enterprises will only be allowed from our trusted partner's locations. This will allow us to specify which other VPN device that we will allow of VPN to connect to.

```
VPN>VPNGateway = xxx.xxx.xxx.xxx
```

xxx.xxx.xxx.xxx is the IP address our Trusted GIAC partners VPN device.

We now need to configure how we are going to authenticate, encrypt and perform a key exchange. Since we are going to be connecting to another VPN 5000 we will issue the following commands to configure the IKE Policy.

Our IPSEC policy will be defined as:

- Authentication – SHA
- Encryption – Triple DES
- IKE – Diffie-Hellman 1024

To set the above policy on the VPN 5000 we will enter the following commands:

```
VPN>configure IKE Policy
VPN>Protection=SHA_3DES_G2
```

This will offer the most security for our VPN. This policy will setup a tunnel between GIAC Enterprises and our trusted partners. This tunnel will allow anyone from our partner to connect and have access into our private resources.

Now we need to allow the actual VPN access:

```
VPN>configure Tunnel Partner VPN 1
VPN>Partner = xxx.xxx.xxx.xxx [remote IP address of trusted partner]
VPN>BindTo = Ethernet 1
VPN>KeyManage = Auto
```

In the above steps we have configured the VPN 5000 to authenticate with our trusted partners and to encrypt all of the traffic between the two sites.

Secure Remote Access
For individuals that need access into the GIAC private network, we will utilize our VPN with RSA Secure ID (ACE/Server).

First we will need to create a group of users that are allowed to connect to the VPN 5000

```
VPN>configure VPN Group "partners"
VPN> MaxConnections = 10
VPN>SecureIDRequired = On
VPN>configure SecureID
VPN>Enabled = On
VPN>PrimaryServer = xxx.xxx.xxx.xxx [ip address of the ACE/Server]
```

The connections into the VPN device will now utilize the ACE/Server for authentication. This 2 step authentication where the user needs to know the username and password and also be in possession of the SecureID number will be enough security for access into the private resources of GIAC Enterprises.

Assignment 3 - Audit Your Security Architecture (25 Points)
You have been assigned to provide technical support for a comprehensive information systems audit for GIAC Enterprises. You are required to audit the Primary Firewall described in Assignments 1 and 2. Your assignment is to:

1. Plan the assessment. Describe the technical approach you recommend to assess your perimeter. Be certain to include considerations such as what shift or day you would do the assessment. Estimate costs and level of effort. Identify risks and considerations.
2. Implement the assessment. Validate that the Primary Firewall is actually implementing the security policy. Be certain to state exactly how you do this, including the tools and commands used. Include screen shots in your report if possible.

3. Conduct a perimeter analysis. Based on your assessment (and referring to data from your assessment), analyze the perimeter defense and make recommendations for improvements or alternate architectures. Diagrams are strongly recommended for this part of the assignment.

Note: DO NOT simply submit the output of NMAP or a similar tool here. It is fine to use any assessment tool you choose, but annotate the output.

Auditing the GIAC network

The audit of GIAC Enterprises will be conducted during non-peak hours so that the audit does not negatively impact the performance of their production machines. Root access to critical machines will have to be obtained by audit personnel in order to attempt to crack user's passwords. After the audit all root account passwords should be changed. Denial of service attacks will not be performed. The audit will take approximately one full day to complete with an audit staff of 2-3 security associates.

The audit of the network will include the following assessments:

- Does the firewall(s) block the ports it is designed to and allow access to the resources that is supposed to?
- Are unneeded services disabled on the publicly accessible machines?
- Are the user accounts on critical machines secured with good passwords?
- Does the Intrusion detection device detect the use of our audit tools?
- Are all of the logs of our public machines being recorded remotely on our syslog server?

Evaluating the External Firewall

NMAP was used to determine what ports were open on the external firewall. NMAP was also used to perform a ping sweep of a range of addresses assigned to GIAC Enterprises to determine if any other machines or backdoors had been installed on the network.

We also checked the firewall logs that were being kept to insure that the logs reported the scanning and we also verified that the Intrusion Detection device reported that we were being scanned.

Figure 2

Logging of events

The firewall logs showed all of the attempted access into the network and they were correctly forwarded to our syslog host. This is just a small sample of the firewall rules in action.

telnet denied

May 20 13:34:29 pluto kernel: Packet log: input REJECT eth0 PROTO=6 61.134.126.1 38:42465 $OUR_ADDRESS:23 L=40 S=0x00 I=4158 F=0x4000 T=232 (#19)

rpc denied

May 20 15:04:46 pluto kernel: Packet log: input REJECT eth0 PROTO=6 65.28.77.96: 4494 $OUR_ADDRESS:111 L=60 S=0x00 I=60101 F=0x4000 T=48 SYN (#19)

The public machines

Nessus was run against each of the public machines to determine if any unneeded services were still running on those machines. The passwords that were created on those machines for administrative access were also attempted to be cracked.

Results of Nessus on the external firewall:

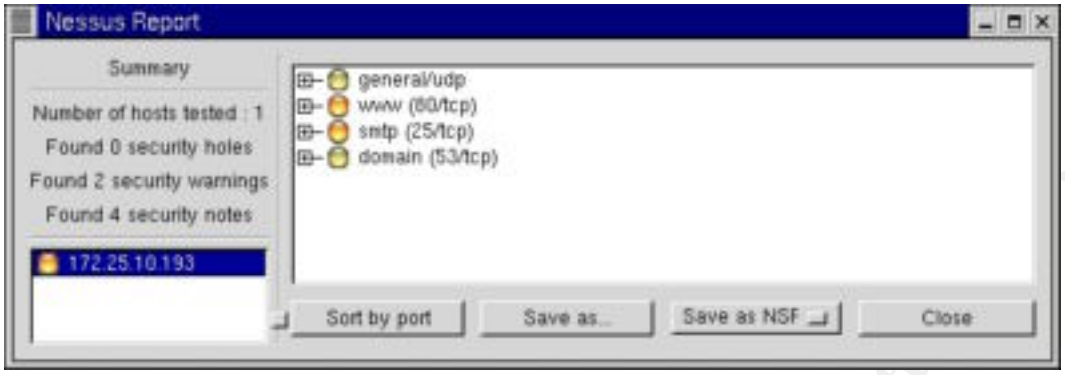


Figure 3

The warnings that were generated by Nessus were implemented by our audit staff, which included configuring our Apache server to not return the version that is running.

Perimeter Analysis

An analysis was performed on the perimeter of the network. The most glaring problem with the setup used is the use of a single router. This single point of failure makes the proposed GIAC network vulnerable. The design was implemented in this manner to keep the design as simple as possible while trying to keep it resistant to Denial of Service attacks as well.

Another design that would add to the robustness of the network would be to utilize two completely separate routers and possibly even two different external firewalls. The tradeoff to this design of course is the added complexity and the add cost of the hardware.

Alternative GIAC Network Design

Figure 4

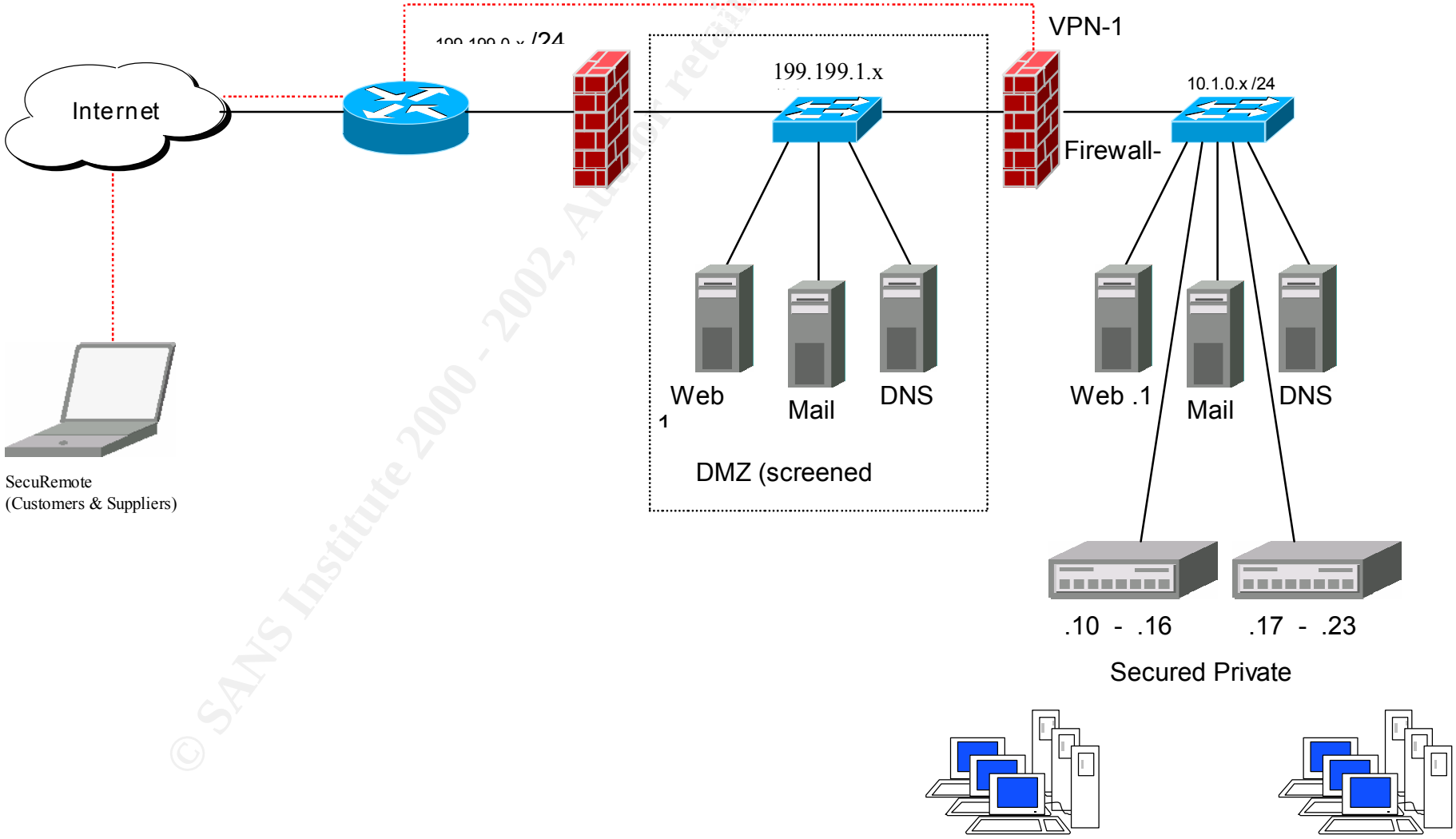
A business decision by the top GIAC staff would have to determine if the extra cost of these proposed measures is justified by the increase in network performance and reliability.

Assignment 4 - Design Under Fire (25 Points)
The purpose of this exercise is to help you think about threats to your network and therefore develop a more robust design. Keep in mind that the next certification group will be attacking your architecture!
Select a network design from any previously posted GCFW practical (http://www.sans.org/giactc/gcfw.htm) and paste the graphic into your submission. Be certain to list the URL of the practical you are using.

Design Under Fire

For this portion of assignment I chose the practical that was submitted by Bob Hockensmith.

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



Attacking the Firewall(s)
Cisco PIX
The firewall that was chosen by Bob Hockensmith was a Cisco PIX 515. Searching for obvious vulnerabilities on the Cisco PIX produced very few results. The CERT site produced vulnerabilities that effected devices that are running a certain version of Cisco IOS, but these specifically excluded the Cisco PIX. The Cisco site produced no results either while looking for vulnerabilities on the PIX.

After extensive searching on the web and on numerous security sites, I finally located this attack. This description of an attack was found with the search phrase of "pix vulnerable".

http://groups.google.com/groups

The basic idea of the attack is that a Cisco PIX 515 configured for authenticating against a TACAS server (exactly the configuration used by Bob Hockensmith) could be overwhelmed with authentication requests thus causing the firewall to either crash or consume all of the firewall resources.

The attack only needed to generate a little over 400 requests in 3 seconds to cause the firewall to crash. The resulting discussion on the Google newsgroups did not lead to a conclusion to the problem, although there are some responses from Cisco engineers. It would appear that any Cisco PIX running 5.1x series of code is vulnerable to this attack as well as a few other vulnerabilities. The Cisco response and patch to fix this problem is apparently not publicly available or publicly discussed at this time. A patch may well exist for Cisco customers, but I could not locate a reference to a fix or workaround.

Checkpoint Firewall 1
The most promising attack against the internal firewall is the IP fragmentation attack as described by Lance Spitzner. This attack of course would have to be made blind, at the internal firewall since an attacker would have no immediate way of knowing what type or brand the internal firewall was unless one of the machines in the DMZ could be compromised and a direct attack could be launched against the internal firewall.

Denial of Service attack
Both of the above mentioned attacks are surely prime examples of catastrophic DOS attacks against this site.

Planning the Attack
By examining the open ports on the firewall it would be obvious that SMTP, HTTP and DNS are the services that are provided by the machines in the screened network. With the setup as described by Bob Hockensmith I believe that the best denial of service attack against this site would be attacking the nameserver that he has setup in the screened subnet. This attack would accomplish a number of goals if successful:

- Stop external clients from being able to resolve hosts that live in this domain
- Outbound queries from the internal network would cease
- Mail services would probably cease or be severely limited, if the mailserver performs name resolution on inbound connections or if it uses the nameserver to lookup outgoing pieces of mail
- The web server could possibly experience problems if it uses reverse lookup to log inbound connections.

If the attack against the nameserver is successful, most if not all of GIAC Enterprises would cease to function.

The Attack
With 50 machines that are my zombies, I would install a program that could send queries to the nameserver, not just a blind spewing of UPD packets to random ports. The program that I found that targeted nameservers is 'ddnsf'.

http://packetstorm.securify.com/distributed

I downloaded this program and verified that I could get it to remotely execute commands. I calculated that the number of queries that a single machine could generate would be in the range of 8,000-10,000 per minute. With the very basic tests that I ran against a well configured nameserver, it would seem that one machine running this piece of code could significantly slow down the response of a nameserver, where 50 machines running this attack simultaneously would certainly render the nameserver under attack useless.

Preventing these attacks
The attack against the firewalls can only be prevented by having vigilant network administrators that are looking for security bulletins and vendors that can and will supply patches to such egregious problems.

I see little that any site could do to prevent the denial of service attack against the nameserver, given the 50 zombie machines and the type of traffic that could be generated by them. Only by recognizing the attack and working with your ISP could you possibly stop all of the inbound packets and even then not until the zombies could be stopped, I think the GAIC Enterprises network and any other entity on the Internet is vulnerable to this type of attack.

Compromise of Internal machine
The operating system of all of the machines in the DMZ would have to be determined. I believe that the Web server would have to be considered the most vulnerable of the machines that have been placed in the DMZ. Since the OS is not specified, it is hard to tell whether or not an attack would be successful. If I were to focus on this machine the operating system would of course determine the next step as far as what attack to use.

With all of the recent Windows IIS exploits, I could only hope that a Windows OS and webserver are in use.

Other possible avenues

A possible problem that I did find in the rules that Bob Hockensmith created for his firewalls was the rule that he mentions in section 2.3 of his internal firewall. He states that he will allow 'inbound and outbound DNS queries (UDP, TCP 53). This will be a obvious security problem, since we can assume that no one from outside of the internal firewall would have a need to perform a DNS zone transfer, we have found a way to learn about every machine that is in the internal network of GIAC Enterprises. I believe that the internal firewall should have only allowed outbound UDP and TCP port 53, no TCP DNS traffic originating from the DMZ should be allowed in to the internal network. This would have to be considered a major security risk and internal machines could possibly be exploited by this type of information leaking out.

List of References:

Ziegler, Robert. Linux Firewalls. Indianapolis, Indiana: New Riders, 2000

Zwicky, Elizabeth, Simon Cooper, D. Brent Chapman. Building Internet Firewalls. 2nd ed. USA: 2000

Northcut, Stephen, Judy Novak. Network Intrusion Detection. 2nd ed. Indianapolis, Indiana: New Riders, 2000

Perlmutter, Bruce, Johnathan Zarkower. Virtual Private Networks. New Jersey: Prentice Hall, 2000

Mann, Scott, Ellen Mitchell. Linux System Security. New Jersey: Prentice Hall, 2000

List of Additional Resources:

Tripwire	http://www.tripwire.org
logcheck	http://www.psionic.com
Nessus	http://www.nessus.org
NMAP	http://www.insecure.org/nmap
openSSH	http://www.openssh.com
Bastille	http://bastille-linux.sourceforge.net
Kernel source	http://www.kernel.org
Linux Documentation	http://www.linuxdoc.org
RSA	http://www.rsa.com
NFR	http://www.nfr.com