Global Information Assurance Certification Paper

GIAC Enterprises Perimeter Security

Written By: Seth Summersett
Date: 2002, Jan 29
Version 1.6a

Specific host access outside of DNS, SMTP, WEB, and IDS is beyond the scope of this paper. The paper will not dwell on host based security configurations, I believe that is covered in papers produced by the UNIX and WINDOWS Certification tracks. Security is a large subject, hence the many GIAC Certifications. This paper will assume (I know bad to do in security) that all host have been properly secured and hardened in the necessary ways for the services they will be running by the teams responsible for that host, people certified in GIAC Windows, Unix, and IDS of course. The paper will concentrate on the proper configuration and hardening of perimeter security. This includes such things as firewalls, border routers, and VPN's. Lastly due to the requirement to audit the configuration diversity will be limited to the equipment that I own, not necessarily the best product for the job. With that said and understood, lets move on.

Security Architecture

1. Connections

Customers

Customers will connect to GIAC Enterprises using the Internet. We will allow online ordering through our web server. The web server will support SSL (Secure Sockets Layer) to allow secure ordering. This functionality will be afforded through Apache-SSL. Each company will have secure usernames and passwords. GIAC Enterprises will select usernames and the customers will select the passwords. GIAC Enterprises will inform all customers that passwords will be tested on a regular basis and will be held to a high standard. The security policy will define this standard along with the ramifications of a password being broken.

The ability to open an account that can order will be done by GIAC Enterprises and will not be a function that is allowed dynamically online. This will help to prevent bogus accounts allowing the ordering process to be as secure as possible. Allowing the functionality and convenience of online ordering will help to grow and expand customer base.

Suppliers

Suppliers shall all connect through dedicated circuits to GIAC Enterprises. The routers placed on suppliers' sites shall be owned and controlled by GIAC Enterprises to allow GIAC Enterprises full control of site security. Suppliers will be treated as EXTERNAL to GIAC Enterprises network. There will be a "suppliers network" setup that allows supplier access to only necessary hosts and areas of the GIAC network. All other parts of the GAIC network shall be protected by a firewall with a "deny all" policy on traffic coming from "suppliers network."

Suppliers' traffic coming across the leased lines shall be verified using IPSec AH (Authentication Header). This will help to prevent any possible packet spoofing coming from a "man in the middle" attack. Also using AH there will be less processing overhead than using ESP (Encapsulating Security Payload). It is understood that AH will not give confidentiality due

to its unencrypted characteristics. AH will be used only between routers, therefore only the "un-secure" point-to-point link will be verified by AH and the entire IPSec tunnel will be under GIAC's control. By using tunnel mode across only the Point-to-Point link we are taking the security risk that the data can be viewed from anywhere within the Partners and Suppliers' network. Therefore, if someone were to compromise that network they would be able to manipulate the data before it reaches the router and is sent over the IPSec tunnel. The IPSec tunnel will be setup on the Cisco Routers that connect the sites.

Suppliers' Telecommuting employees will be required to connect to their own internal networks and then come over to GIAC's network. They will **NOT** be granted "back door" access to GIAC Enterprises "suppliers network." If this type of connection is explicitly required it can be handled on a case-by-case basis but is against my security recommendation.

Once the suppliers have entered the GIAC network they will have access to only the machines within the "suppliers network" they have been explicitly granted access to. This will be enforced using access-lists on the Cisco Routers. They will not be able to "surf" the web via GIAC Enterprises.

Partners

Partners shall all connect in the same fashion as explained above, but will have a "partners network" they will use. This way the partners and suppliers traffic will not intermix and cause any security risks. This will be accomplished via VLAN's. I understand that VLAN's are not a complete security package, but this is all that shall be required to keep the traffic separated.

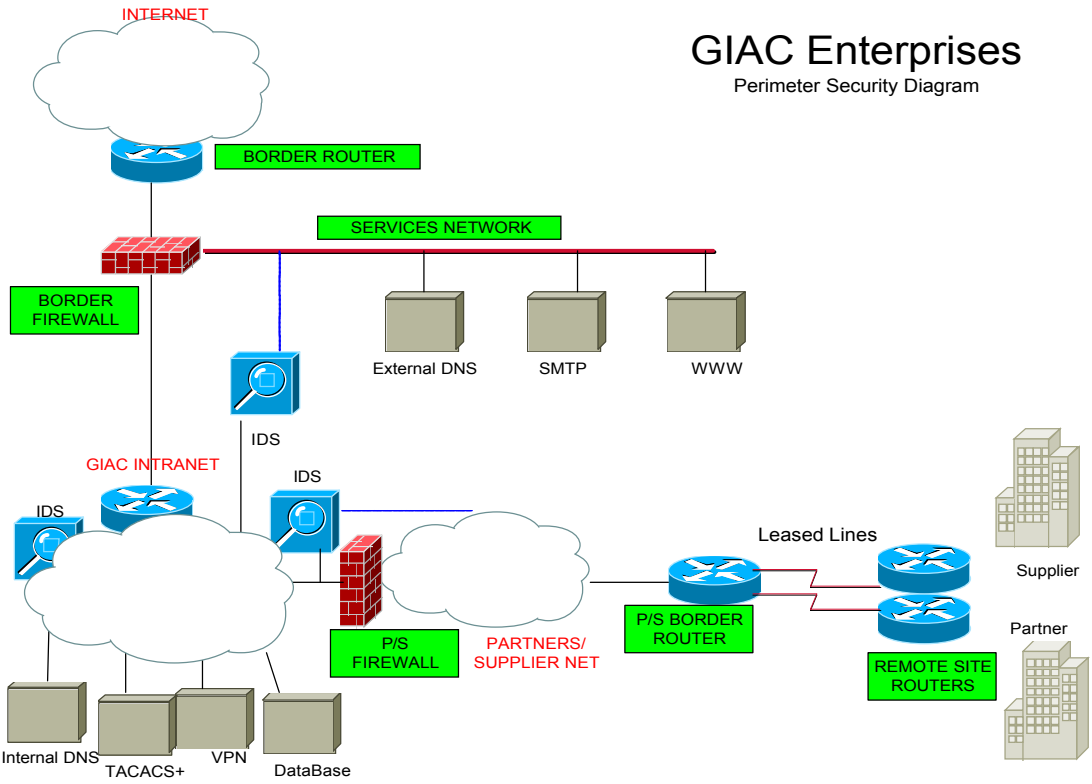
GIAC Employees

Internal Employees of GIAC enterprises come in more than one flavor. They can be full-time telecommuters, traveling telecommuters (employees who travel) and full-time on-site employees. Traveling telecommuters create a difficult situation because they need access from many different locations. Traveling salespeople are in many different locations during any given week and will need access to the GIAC network from each of these varying locations.

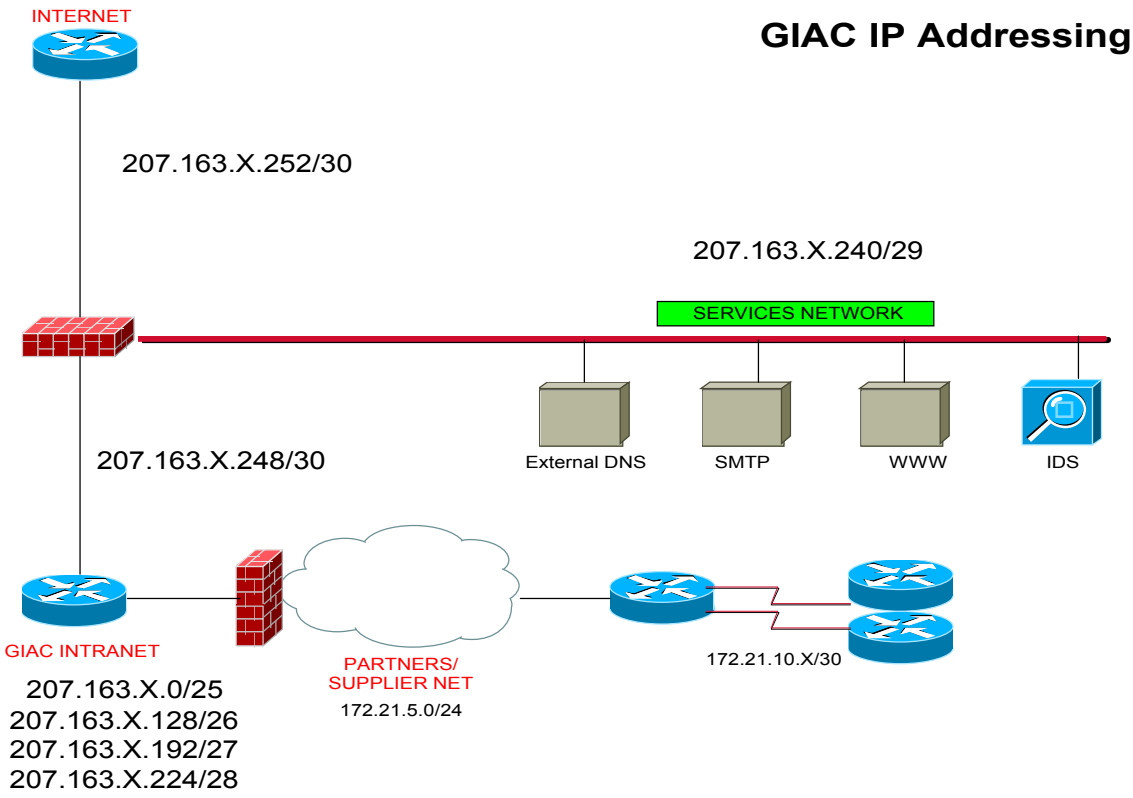
To allow for easier administration, GIAC has opted to pass up any direct connections to the network such as ISDN, DIAL-UP, or DSL. They will make all remote users whether full-time or traveling telecommuters enter the GIAC network using a VPN. The VPN will be IPSec ESP based. In using ESP we have confidentiality through encryption and we also have verification through hashing. ESP has more overhead than AH, but will also give confidentiality while it traverses many "un-secure" networks.

Regular on-site employees and telecommuters once part of the GIAC network will be held to the employees access "security policy." This policy declares what access rights employees shall have to internal and external machines via the GAIC network. This tells what services are allowed inside and outside of GIAC's network.

Network Components and Architecture



Above (GIAC Security Architecture) – Below (IP Addressing Scheme)



Border Router

Device: Cisco 3620 Enterprises from the outside world. This router will provide packet Software: c3620-io3-mz.122-6.bin - This software image includes IP, Firewall Feature Set and also Cisco's router IDS. This will allow for the possibility of future expansion of the responsibilities or usefulness of the border router.

The Border Router is the first line of defense protecting GIAC Enterprises from the outside world. This router will provide packet based filtering. In doing so it will lighten the load on the Primary Firewall while also acting as a mini-firewall itself. Although the router can do stateful filtering it will not be used in this capacity. The Primary Firewall will complete the stateful filtering function of the layered security design. Also the Primary Firewall most likely has better and more proficient logging than does the router.

This device will connect directly to the ISP servicing GIAC Enterprises. Then it will connect to the Primary Firewall.

Primary Firewall

Device: Slackware 8.0 → Kernel 2.4.16
Software: netfilter/iptables-1.2.5

The Primary Firewall is the main line of defense against the outside world. GIAC has a layered defense architecture, but the firewall is the heart of these layers. The firewall will provide redundancy for the filtering performed by the Border and Internal Routers, in case either has a

bug and lets packets through. Also, this device will provide stateful packet filtering. This firewall will define what traffic patterns are acceptable, what services (ports) certain devices will be allowed to receive packets for.

The primary firewall will sit behind the LAN interface of the Border Router and the LAN interface of the firewall will connect to an internal router that will distribute traffic to the local GIAC network. The firewall also has a third interface that connects to the GIAC Services Network. This network holds all GIAC's public services such as the GIAC web server, GIAC DNS, and the GIAC email server.

VPN

Device: Cisco VPN 3030 Concentrator
Software: Cisco VPN 3000 Concentrator 2.5.2(F)

Full-time or part-time telecommuters enter the GIAC network using a Virtual Private Network. This will allow telecommuters to use their own Internet Service Providers (ISP's) to connect to the Internet and then using the Internet they are able to enter the corporate network in a secure fashion. Using a VPN will encrypt all traffic traversing the Internet, allowing our users to have secure communications with the internal network from remote locations.

The VPN device will sit on the internal LAN behind the firewall. This way a user will have to be able to circumvent the firewall to attack the VPN device to gain access to the GIAC LAN via our VPN device.

P/S Border Router

Device: Cisco 3660
Software: c3660-ik8o3s-mz.122-6.bin – This software package includes IP and IPSec as needed for the current implementation. The package also has support for Firewall Feature Set and IDS if these services should become necessary in the future.

The P/S Border Router will provide the connection to the Partner / Suppliers networks. This device, along with the remote site router, will provide ingress and egress packet filtering. This will help to ensure that only allowed communications between the Partners / Suppliers Network and GIAC P/S Network is occurring. As an added security feature there will be a VPN setup between this border router and each of the remote site routers, helping to secure the link between these sites. The 3660 was chosen due to high port density and faster processor for the VPN.

P/S Firewall

Device: Slackware 8.0 → Kernel 2.4.16
Software: netfilter/iptables-1.2.5

This firewall is to protect GIAC's internal network from malicious users within the Partners or Suppliers' networks. Also this is used to protect the GAIC network from a machine within the Partners or Suppliers' networks that has been compromised and is being used to launch an attack into our network.

This firewall will be placed between the GIAC Internal Network and the Partner/Supplier Network. The firewall will only allow a one machine from GIAC's internal network to communicate with the Partners / Suppliers' network. These machines will only communicate for very specific tasks such as data synchronization.

IDS

Device: FreeBSD 4.4
Software: Snort 1.8.3

The Intrusion Detection Systems will help us to see attacks occurring on our networks. There is one that is attached to the Internal LAN, one on the P/S Network and also one connected to the Services Network. Note that the interfaces on the IDS connected outside of the Internal GIAC LAN have no TCP/IP stack installed on those interfaces. This will hopefully prevent these boxes from being attacked and used to circumvent the respective Firewalls.

Written Security Policy

1. Policy takes "deny everything" stance. Everything will be denied unless deemed to be necessary for business operations. To be deemed necessary for business operations service must go through the approval policy procedures set forth by management.
2. External to Internal LAN traffic
    a. Allow port 53 (udp) to "internal DNS" server only
    b. Allow port 22 (tcp) to allow remote administration
    c. Allow port 49 (tcp) from BRD_RTR to TACACS Server for router authentication
    d. Allow port 500 (udp) for IKE of IPSec
    e. Allow protocol 50 for ESP of IPSec
    f. Deny all that is not established
3. External to Services traffic
    a. Allow only ports necessary for specific servers
        i. WWW Server
            1. HTTP (tcp 80) and HTTPS (tcp 443)
        ii. MAIL Server
            1. SMTP (tcp 25)
        iii. DNS Server
            1. DNS (53) udp
            2. DNS (53) tcp only from secondary server ip address
    b. Deny all else
4. Internal LAN to External traffic / Service Network
    a. HTTP (tcp 80), HTTPS (tcp 443), and SSH (tcp 22)
    b. POP3 (tcp 110) to Services Network
    c. DNS (udp 53) from Internal DNS Server
    d. Established
        i. Authen (tcp 49) from TACACS Server to BRD_RTR
        ii. IKE (udp 500) from VPN Server

iii. ESP (protocol 50) from VPN Server
   e. Oracle Database (tcp 1521) from Database Server to Web Server
   f. Deny all else
5. Services to External traffic
   a. Established traffic on allowed ports from section 3 above
   b. Deny all else
6. Service to Internal traffic
   a. Established traffic on allowed ports from section 5 above
   b. Deny all else
7. Internal to P/S traffic
   a. Allow only ports used for data synchronization, keeping the subset of machines very small that are allowed this communication with a small subset of P/S hosts.
   b. Deny all else
8. P/S to Internal traffic
   a. Established traffic from allowed machines and ports from section 8 above
   b. Deny all else
9. Authentication traffic
   a. Network will use TACACS+ to authenticate users logging onto routers and to the Remote Access VPN. To accommodate this the firewall will allow necessary traffic specified in host format to travel across port 49.

Security Policy

Border Router

To limit the number of logs the administrator must filter through on a daily basis. There is no compelling reason to log the usage of bad source addresses coming from the Internet. Therefore the Ingress filter will not log.

Before adding any packet filtering capabilities to the router and especially before connecting this router to the Internet we must harden the router itself.

1. Controlling Access to the Router
   a. VTY- Virtual Terminal Lines allows access through telnet to the router – only allow access to these lines by administrators. Do not allow outgoing telnet sessions from router. If offered on router enable ssh and telnet.

i. Border_Router(config)#aaa new-model
ii. Border_Router(config)#aaa authentication login TELNET group tacacs+ local
iii. Border_Router(config)#access-list 10 permit host 207.163.X.12
iv. Border_Router(config)#access-list 10 permit host 207.163.X.15
v. Border_Router(config)#access-list 11 deny any
vi. Border_Router(config)#line vty 0 4
vii. Border_Router(config-line)#service tcp-keepalive-in
viii. Border_Router(config-line)#access-class 10 in

ix. **Border_Router(config-line)#access-class 11 out**
x. **Border_Router(config-line)#login**
xi. **Border_Router(config-line)#authentication TELNET**

b. CON – Console port allowing physical access to router
i. Border_Router(config)#line con 0
ii. Border_Router(config-line)#login
iii. Border_Router(config-line)#password donthurtme

c. AUX – Allow remote connections to router from mode. This will not be used at GIAC enterprises so will be disabled as best as possible
i. Border_Router(config)#line aux 0
ii. Border_Router(config-line)#no login
iii. Border_Router(config-line)#exec-timeout 0 0

d. All lines shall have timeouts applied to limit the likelihood of someone stumbling across an open connection and being granted access to device
i. Border_Router(config-line)#exec-timeout 5 30
1. says allow connection to be idle for 5 minutes and 30 seconds before dropping the connection

2. Services on Router
a. Password encryption
i. Config Password encryption
1. Border_Router(config)#service password-encryption
ii. Enable – always use "enable secret" instead of "enable password" because it used better hashing
1. Border_Router(config)#enable secret enablepassword

b. HTTP Administration – Always turn this off has security bug **
i. Border_Router(config)#no ip http server

c. CDP – Cisco Discovery Protocol. Protocol to identify neighboring Cisco devices. Can be disabled on a per-interface basis but we will globally disable it
i. Border_Router(config)#no cdp run

d. NTP – Network Time Protocol. Protocol used to keep clocks on many network devices synchronized. Not necessary for our router.
i. Border_Router(config)#no ntp enable

e. SNMP Administration
i. If using this function DO NOT USE DEFAULT community strings of public. Be sure to change strings. Also if possible only allow RO strings and not RW strings. Use access lists to only allow certain hosts to execute these functions.
ii. This is disabled on GIAC Routers
1. Border_Router(config)#no snmp community public

f. Misc services most likely not necessary
i. Border_Router(config)#no service udp-small-servers
ii. Border_Router(config)#no service tcp-small-servers
iii. Border_Router(config)#no ip finger
iv. Border_Router(config)#no ip source-route

v. **Border_Router(config)#no service pad**
vi. **Border_Router(config)#no ip bootp server**
vii. **Border_Router(config)#no ip unreachables**
viii. All Interfaces – help to eliminate participation in Smurf Attacks (1)
1. **Border_Router(config-int)#no ip directed-broadcast**
2. **Border_Router(config-int)#no ip redirects**
3. **Border_Router(config-int)#no ip unreachables**
4. **Border_Router(config-int)#no ip proxy-arp**

3. Banners
   a. **Border_Router(config)#banner C**
      UNAUTHORIZED ACCESS TO THIS DEVICE IS PROHIBITED. C

Packet filtering will be used on the border router to allow the router to take some of the load off of the Primary Firewall. The router will disallow traffic from Private IP address ranges (RFC 1918), IANA reserved addresses, packets sourced from a GIAC internal ip address, packets destined to something not in GIAC network address range, and packets addressed to loopback. Since we will be checking both source and destination addresses we will be forced to use slower extended access lists, but in applying the access-lists to both internal and external interfaces. This will allow a long standard access-list to be applied to one interface and then a shorter extended access-list to be applied to the other interface.

Syntax

access-list [list number] [permit | deny] [source-addr wildcard-mask] [log] [options]

Ingress Filter

See Appendix B.1 reference access-list 5

To allow the long list of source address checking to be the faster "standard" access-list has been applied to the WAN interface to filter these addresses. We also want to filter on destination address therefore; I have made this small "extended" access-list to check the destination address. This access-list will have to be applied to the LAN interface because each interface is only allowed one incoming list and one outgoing list. Therefore we will check source address incoming at the external INTERNET interface and check the destination address at the internal LAN interface.

Syntax

access-list [list number] [permit | deny] [protocol | protocol keyword] [source-addr wildcard-mask] [source port] [dest-addr wildcard-mask] [dest port] [log] [options]

access-list 115 permit ip any 207.163.X.0 0.0.0.255

Also in packet filtering we will have an egress filter that disallows any packets sourced from something other than an internal ip address range from exiting. These packets will also be logged allow administrators to reference back to what machines were causing these spoofed packets. This filter is redundant to what the firewall will most likely already be doing, but if the firewall has a bug and allows some of these through, hopefully they will be caught here at the router.

Egress Filter

access-list 15 permit 207.163.X.0 0.0.0.255
access-list 15 deny any log

Apply Filters

Border_Router(config)#int serial1
Border_Router(config-int)#ip access-group 5 in
Border_Router(config-int)#int ethernet0
Border_Router(config-int)#ip access-group 115 in
Border_Router(config-int)#ip access-group 15 out

The Border Router will help to assist the Primary Firewall in preventing TCP SYN attacks by using "tcp intercept". There are two possible modes for this functionality, intercept and watch. With intercept mode the router will hold and respond to the SYN packet itself waiting for the ACK from the initiating device; once this is received the router will forward the original packet on and remove itself from the stream. In watch mode the router will watch the TCP 3-way handshake to see if the remote device sends the ACK, if in 30 seconds (default) the router doesn't see an ACK packet it will send a RST to the local device. GIAC's Border Router will use TCP intercept in "watch" mode, configuring tcp intercept is simple. First an access-list is used to define the traffic we are interested in. Then use the "tcp intercept" command telling what access-list to use, lastly since we are not using the default mode, we declare the mode as "watch".

Border_Router(config)#access-list 150 permit tcp any 207.163.X.0 255.255.255.0
Border_Router(config)#ip tcp intercept list 150
Border_Router(config)#ip tcp intercept mode watch

The border router only has two interfaces (external and internal), unless required by ISP we will use static routing. This will reduce possible security breaches in dynamic routing protocols, while giving us a faster convergence time in case of link or router failure.

Entire router configuration can be found in Appendix B.1.

Primary Firewall

Since the firewall is a Linux based NETFILTER firewall it is necessary to harden Linux itself. A description on how to harden Linux is beyond the scope of this paper, but I will say that all services shall be shutdown on this box. No services shall be running except those necessary

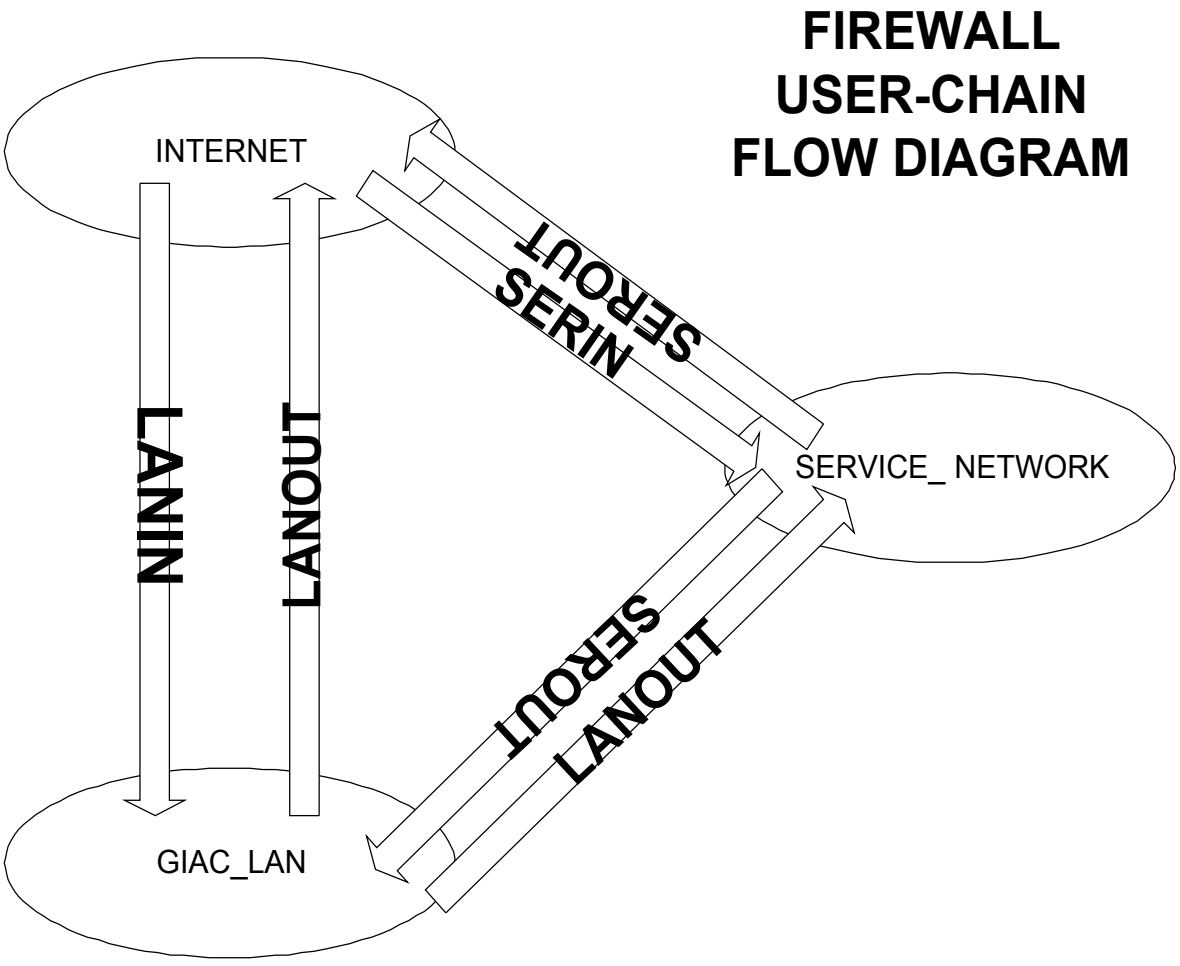
for running NETFILTER/iptables. This Linux box will not have remote configuration capabilities so all remote administration services shall be turned off. Administration to this box shall occur locally, limiting administration vulnerabilities beyond the Linux distribution and NETFILTER to physical security.

Syntax

iptables –NXPLFZAIRD [chain-name] [policy | match condition –j [DROP | ACCEPT | REJECT | NAT | LOG | TOS | usr table]

For a better understanding of the iptables syntax please reference the tutorial in Appendix A.

Below is a flow diagram showing how traffic if forwarded to the user defined chains in the ruleset.



Internet → GIAC Internal LAN (LANIN)

We will allow the services stated above in the security policy to enter our network. We will do this by explicitly opening these ports. Then we will allows sessions that have been created from the inside to flow in both directions; lastly we will drop anything that doesn't meet

the previous two statements.
First we will check the source address and make sure that it is not IANA Reserved or private.

iptables -A LANIN -j SRC_CHECK

Next we will explicitly allow the services permitted by our security policy above, ssh, authentication, and VPN. The first rule allows ssh in from anywhere to anywhere. Next, we allow TACACS traffic to go from the Boarder Router to the Tacacs Server. The last two rules allow VPN sessions to be created. UDP 500 is for IKE and the protocol 50 is for ESP.

```
iptables -A LANIN -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A LANIN -p tcp --dport 49 -s $BRD_ROUTER -d $TAC_SERVER \
        -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A LANIN -p udp --dport 500 -d $VPN_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A LANIN -p 50 -d $VPN_SERVER -m state --state \
        NEW,ESTABLISHED -j ACCEPT
```

Now that we have allowed all the necessary individual services we shall check to make sure the rest of the incoming packets are valid. First we make sure that all tcp and udp packets have a destination port higher than 1024. Then we drop any packets that do not have a valid session in the state table. Next, we will allow all packets that have gotten this far and have a valid session in the state table to pass.

```
iptables -A LANIN -p tcp,udp --dport ! 1024:65535 -j DROP
iptables -A LANIN -m state --state INVALID -j DROP
iptables -A LANIN -m state --state ESTABLISHED -j ACCEPT
```

Finally to finish off this table we will log any packets that have made it this far and then drop them. This will allow the firewall administrator to understand what traffic the firewall ruleset is not accounting for.

```
iptables -A LANIN -m limit --limit 5/minute -j LOG \
        --log-prefix "LANIN nondrop"
iptables -A LANIN -j DROP
```

Internet → Service Network (SERIN)

First as above, we check to make sure that the source address is not IANA Reserved or Private. Then we will strictly control what ports and protocols can go to which servers within the Services Network. First we allow tcp 25, SMTP to go to the mail server. Next we allow tcp 80 and 443 to the web server, so that it can service HTTP and HTTPS requests. Last we allow udp based DNS request to be directed at the DNS server.

```
iptables -A SERIN SRC_CHECK
iptables -A SERIN -p tcp --dport 25 -d $MAIL_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A SERIN -p tcp --dport 80 -d $WWW_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A SERIN -p tcp --dport 443 -d $WWW_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A SERIN -p udp --dport 53 -d $DNS_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
```

Now that we have explicitly allowed what we want to come into our Services Network we will log and drop all other traffic as we did above in the LANIN table.

```
iptables -A SERIN -m limit --limit 5/minute -j LOG \
        --log-prefix "SERIN nondrop"
iptables -A SERIN -j DROP
```

GAIC LAN → Internet and Service Network (LANOUT)

Similar to the other tables we will first check the source address. This time it is a little different though. We want to make sure that the source address is from the range of IP Addresses that we have allocated to the network. Also we are **very** interested if one of these packets gets caught not having the correct source address. So we will log then drop any packets that do not have the correct range of IP Addresses. Then we will check to make sure that the destination is a valid address, making sure that it is not IANA_Reserved or Private range.

```
iptables -A LANOUT -s ! $INTERNAL_NET -j LOG --log-prefix "SPOOFING"
iptables -A LANOUT -s ! $INTERNAL_NET -j DROP
iptables -A LANOUT -j DST_CHECK
```

Next we will let out all the services declared in the security policy above. First is ssh, then http and https. Then we let out pop3, but only if it is destined for the mail server. We will allow tacacs authentication traffic through if it is destined for the BRD_RTR, coming from the tacacs Server, and the session is in the state table. Now we allow the Database Server to talk to the web server using port 1521, Oracle port. Next we allow the Internal DNS server to make udp DNS queries and last we allow established IPSec session information through using udp 500 and protocol 50 if it is sourced from the VPN Server.

```
iptables -A LANOUT -p tcp --dport 22 -m state --state \
        NEW,ESTABLISHED -j ACCEPT
iptables -A LANOUT -p tcp --dport 80 -m state --state \
        NEW,ESTABLISHED -j ACCEPT
iptables -A LANOUT -p tcp --dport 443 -m state --state \
        NEW,ESTABLISHED -j ACCEPT
```

```
iptables -A LANOUT -p tcp --dport 110 -d $MAIL_SERVER  -m state --state \
        NEW,ESTABLISHED -j ACCEPT
iptables -A LANOUT -p tcp --sport 49 -d $BRD_ROUTER -s $TAC_SERVER \
        -m state --state ESTABLISHED -j ACCEPT
iptables -A LANOUT -p tcp --sport 1521 -s $DBASE -d $WWW_SERVER -j ACCEPT
iptables -A LANOUT -p udp --dport 53 -s $INTERNAL_DNS -j ACCEPT
iptables -A LANOUT -p udp --sport 500 -s $VPN_SERVER -m state --state \
        ESTABLISHED -j ACCEPT
iptables -A LANOUT -p 50 -s $VPN_SERVER -m state --state ESTABLISHED -j \
        ACCEPT
```

Now that we have allowed all the necessary services through the firewall we will log anything that has not already been dropped for later analysis.

```
iptables -A LANOUT -m limit --limit 5/minute -j LOG \
        --log-prefix "LANOUT nondrop"
iptables -A LANOUT -j DROP
```

Service Network → Internet and GIAC LAN (SEROUT)

First we check to make sure that our Service Network is not spoofing packets, if so log and drop. We will also make sure the destination address is valid.

```
iptables -A SEROUT -s ! NOT_INTERNAL -m limit --limit 5/minute \
        -j LOG --log-prefix "SERVICE SPOOFING"
iptables -A SEROUT -s ! NOT_INTERNAL -j DROP
iptables -A SEROUT -j DST_CHECK
```

Now we will check to make sure that only the web server is sending web traffic, only the mail server is only sending mail traffic, and the dns server is only sending dns traffic. Also we make sure that ssh and pop3 traffic is only destined for the internal lan and has already been established. We also support the Oracle traffic destined from the web server to the database server.

```
iptables -A SEROUT -p tcp --sport 80 --dport 1024:65535 -s $WWW_SERVER \
        -m state --state ESTABLISHED -j ACCEPT
iptables -A SEROUT -p tcp --sport 443 --dport 1024:65535 -s $WWW_SERVER \
        -m state --state ESTABLISHED -j ACCEPT
iptables -A SEROUT -p tcp --sport 25 -s $MAIL_SERVER -m state \
        --state ESTABLISHED -j ACCEPT
iptables -A SEROUT -p tcp --sport 22 -d $INTERNAL_NET -m state \
        --state ESTABLISHED -j ACCEPT
iptables -A SEROUT -p tcp --sport 110 -d $INTERNAL_NET -s MAIL_SERVER -m \
        state --state ESTABLISHED -j ACCEPT
```

iptables -A SEROUT -p tcp --dport 1521 -s $WWW_SERVER -d $DBASE -j ACCEPT
**iptables -A SEROUT -p udp --sport 53 --dport 1024:65535 -s $DNS_SERVER **
-m state --state ESTABLISHED -j ACCEPT

Next we log then drop any outgoing syn packets. There should be no reason a device inside the service network should be sending any syn packets this far down in the rule base. It is likely if we get this log we have been compromised.

**iptables -A SEROUT –tcp --syn -m limit --limit 5/minute -j LOG **
--log-prefix "SERVICE SYN"
iptables -A SEROUT –tcp --syn -j DROP

Lastly, as in all other tables we will log and drop any remaining traffic for later analysis.

**iptables -A SEROUT -m limit --limit 5/minute -j LOG **
--log-prefix "SEROUT nondrop"
iptables -A SEROUT -j DROP

There are more tables used in the complete firewall configuration, which can be found in the appendix.
Entire firewall configuration can be found in Appendix B.2.

VPN

The VPN will be IPSec ESP based. IPSec is actually a two-fold process. First the two endpoints will establish an IKE session to control and provide key management for the IPSec session. After the IKE tunnel is completely up it will negotiate the parameters of the IPSec tunnel. Then once the IPSec tunnel is up data will pass through the IPSec tunnel.

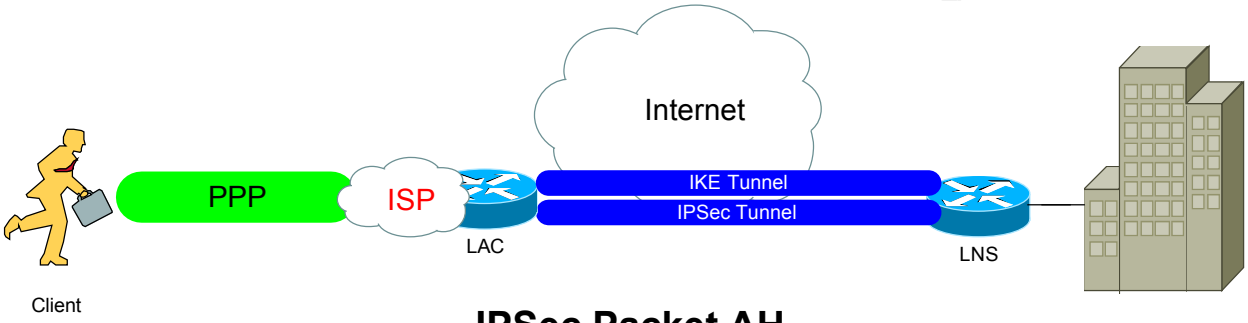
Establishing the IKE tunnel has two different modes Main and Aggressive. Main mode takes a little longer, but has the added feature of providing identity protection for the initiating device. Aggressive mode on the other is fast, but does not provide identity protection. To establish the IKE session we will use **Main Mode** for the benefit of added security.

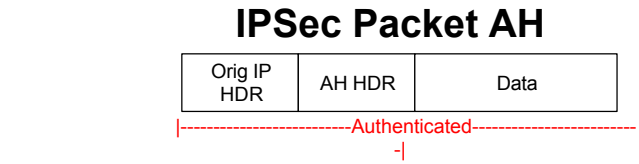
For the IPSec tunnel we have two different protocols we can choose and three different combinations we can make with those protocols. IPSec has AH (Authentication Header) and ESP (Encapsulating Security Payload). The three possible combinations for the tunnel are AH only, ESP only, or both AH and ESP. AH is a lightweight protocol designed with the option of using other VPN protocols below it, such as L2TP or IPSec ESP. AH adds a cryptographic checksum generated from the contents of the IP packet. As seen in the picture below, AH provides no confidentiality.

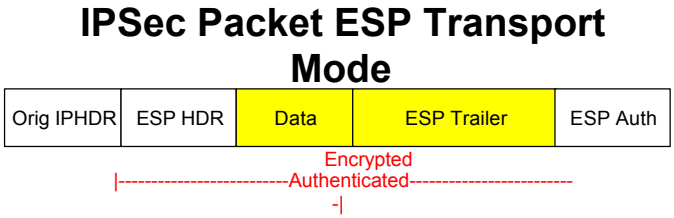
ESP on the other hand does provide confidentiality. ESP will encrypt parts of the IP packet. The algorithms used with ESP are a decision that is left up to the manufacturer, per RFC 2406. Also, as seen in the picture below there are two possible modes with IPSec ESP, transport and tunnel. With transport the original IP header is used in sending the packets, therefore the endpoints of the tunnel communication are known. With tunnel on the other hand, the original IP header is encrypted and the IPSec device places a new header on the IP packet. This provides

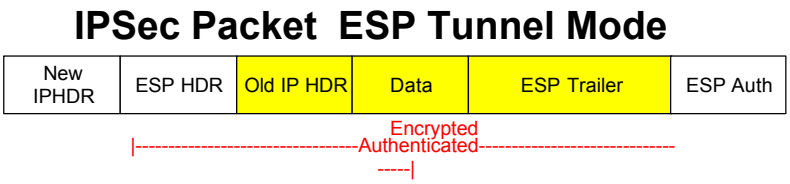
a little more security because a person sniffing packets could not be sure that the IP addresses on the packets were the true beginning and end of the communications channel.

In our implementation of IPSec we have chosen to use just **ESP**. ESP will not authenticate the header, but it will authenticate the rest of the packet. In using ESP we have confidentiality through encryption and we also have data integrity through hashing. In having confidentiality we know that no one can read our data, also with data integrity we have the ability to insure that nobody has tampered with the data we are receiving. ESP has more overhead than AH, but will also give confidentiality while it traverses many "un-secure" networks.









We will use 3DES to encrypt the data and MD5 to hash(authenticate). 3DES uses a 112-bit key and under intense security scrutiny only "theoretical" vulnerabilities have been discovered. Split tunneling will be disabled in all cases, **NO** exceptions. The ability to be on the "secure" network, while also being on the Internet via an "unsecure" network, is a risk GIAC is not willing to take. Also users accessing the VPN will be using one-time passwords to access the VPN device. This will help to make it more difficult for an attacker that has physical access to a remote machine to logon to the VPN. They would need to have the password to the device that gives out the one-time passwords before they could access the VPN.

Partner / Suppliers Border Router

For the P/S Border Router we shall use the same hardening techniques as above. Here our configuration changes. The ingress filtering will all happen on the remote routers that GIAC places at Partners and Suppliers sites. The egress filtering will happen on this router though. The connection from the Partners or Suppliers site will be a dedicated circuit that will be using IPSec AH to verify the validity of the data we are receiving. Below we will discuss the egress filtering and the VPN that is setup between the routers.

The ingress filtering on the remote routers will define what computers on the partner/supplier network can access computers on the GIAC P/S network. This will help to limit the traffic that travels the length of the dedicated circuit and must go though the VPN. Also this will help to offload filtering work to the onsite routers verses having the P/S Boarder Router handle all network filtering responsibilities. Traffic that is being dropped at these sites will be logged, to help assist administrators in deciding whether one of the Partners or Suppliers' networks is trying to access information it should not be accessing.

This egress filter will be located on the Cisco 3660 onsite. This will only allow traffic sourced from within the GIAC network to exit. Since we are using extended access-lists we have the ability to verify the destination addresses, which should always be from one of the Partner/Suppliers' network addresses. This list will change as Partners and Suppliers change.

Egress Filter

access-list 101 permit ip 172.16.5.0 0.0.0.255 10.10.10.0 0.0.0.255
access-list 101 permit ip 172.16.5.0 0.0.0.255 110.10.10.0 0.0.0.255
access-list 101 permit ip 172.16.5.0 0.0.0.255 210.10.10.0 0.0.0.255
access-list 101 permit ip 172.16.5.0 0.0.0.255 120.10.10.0 0.0.0.255
access-list 101 permit ip 172.16.5.0 0.0.0.255 60.10.10.0 0.0.0.255
access-list 101 permit ip 172.16.5.0 0.0.0.255 30.10.10.0 0.0.0.255
access-list 101 deny ip any any log

VPN Configuration

First we will create an access-list that will define the traffic to be encrypted and sent across the tunnel. This will be applied later in the configuration process.

PS_Gateway(config)#

access-list 155 permit ip 172.21.5.0 0.0.0.255 10.10.10.0 0.0.0.255

Now we must create the ISAKMP policy. This defines what type of authentication will be used. The configuration between sites will be static therefore; the router will use pre-shared keys.

PS_Gateway(config)#crypto isakmp policy 10
PS_Gateway(config-isakmp)#authentication pre-share
PS_Gateway(config)#crypto isakmp key I_AM_KEY address 172.16.1.2

Next we will define the transform set. This will define whether we use AH, ESP or both. Also

this will tell what type of hashing and encryption algorithms are to be used. The routers will only use AH, with MD5 hashing. We also define the mode of the IPSec session as tunnel or tranport, tunnel in our configuration.

PS_Gateway(config)#crypto ipsec transform-set AH_MD5 ah-md5-hmac
PS_Gateway(cfg-crypto-trans)#mode tunnel

Next we will create the crypto mapping. This section of the configuration brings most everything together. Here we will define the remote peer for this mapping. Also we will declare the IPSec policy (created in the last step). Last we will define the traffic to match for the tunnel with the access-list created earlier.

PS_Gateway(config)#crypto map toPartorSup 10 ipsec-isakmp
PS_Gateway(config-crypto-map)#set peer 172.16.5.2
PS_Gateway(config-crypto-map)#set transform-set AH_MD5
PS_Gateway(config-crypto-map)#match address 155

Now we will apply the crypto map just created to the interface the tunnel will be used over.

PS_Gateway(config)#

interface s1/0
PS_Gateway(config-if)#ip address 172.16.5.1 255.255.255.0
PS_Gateway(config-if)#crypto map toPartorSup

The router at the remote site will have a similar configuration.

Entire router configuration can be found in Appendix B.3.

Partner / Services Firewall

Entire firewall configuration can be found in Appendix B.4.

GAIC Perimeter Security Audit

Audit Phases

There will be three phases to the audit of the perimeter security for GIAC.
Phase I:

This consists of sitting down and speaking with the management and technical staff at GIAC Enterprises. In this meeting auditors will discuss the goals of the audit, what will happen, when it will occur, and how. This will allow the auditing team to gather an idea of what management expects, how the technical staff feels about the audit and possibly the technical ability and security awareness of the perimeter security team.

Phase II:

In this phase the actual audit will be conducted. Some parts of the audit will take place during the day, such as port scan, to allow them to see how well logging is working under normal business conditions. Other parts of the audit will be conducted when the least amount of business is taking place, things such as DOS (Denial Of Service) attacks. These functions are split in this fashion in case the network is vulnerable, they do not want to bring the company down during peak business hours.

Phase III:

Here is where the auditors will compile into a report the data received in phase II above. This report will detail findings along with solutions and recommendations. Also, they will sit down with the management and the technical staff once again to be sure they understand the findings and recommendations. Doing a security audit is worthless unless the parties involved understand what was found, the implications of what was found, and how to fix/prevent this in the future.

PHASE I

The first phase of the audit will be a chance for the auditors to speak with the managers and technical staff. It is in this meeting that the auditors will explain they will be checking just the primary firewall security, **NOT** the entire perimeter security architecture. Auditors will explain that they will be checking the integrity of the underlying Operating System that the firewall resides on and they will also verify that the firewall rule-set is working as the administrators expects.

In this meeting the auditors will explain that all audits to the firewall will occur during normal business hours, unless there is a possibility the feature or vulnerability being tested could possibly harm the flow of network traffic. Finally, the auditors will explain to the staff that after the audit is complete a document will be created explaining what was done, how, what was discovered, implications of discovery, and how to fix problems found. In conjunction to giving this document to the company the auditors will verbally explain what is contained in the document and answer any necessary questions.

Before leaving the room auditors will need to have the IP address(es), firewall product manufacturer(plus version+patches), and if applicable Operating System firewall uses(plus version+patches).

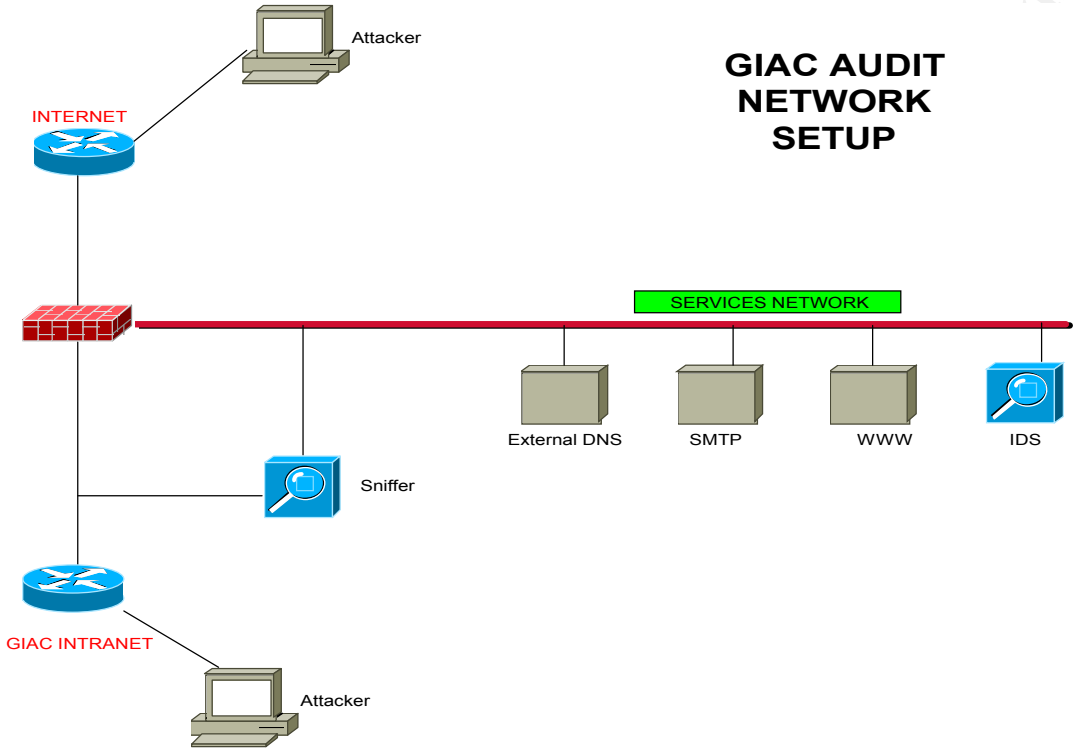
Now that all the rules have been laid out the auditors will begin the technical portion of the audit.

PHASE II

This second phase of the audit is separated into two parts. First, auditing techniques that can be employed during high traffic business hours and non-critical business hours. The reason for the separation is that some tests could make network devices unavailable and the auditors do not want to interrupt any important network traffic.

Business hours audit

- Determine publicly available information
- Audit Firewall box
    - nmap port scanning
- Audit Firewall rule-set
    - nmap port scanning



First auditors will search for publicly available information about the company. This is done to see what kind of information an outsider could gather about the company. The auditors already have the necessary information to perform the audit, but they would like to see if they could gather this without being given this information. Auditors use a simple utility called "whois" to see who owns a particular domain and the addresses assigned to that domain. With this utility they could possibly gain information such as the administrator of the domain and names of machines in the domain. But, as they see we can see below it didn't render quite as much information as it could have.

$whois giac.org

Domain Name: GAIC.ORG
Registrar: NETWORK SOLUTIONS, INC.
Whois Server: whois.networksolutions.com
Referral URL: http://www.networksolutions.com
Name Server: NS1.GIAC.ORG
Name Server: NS.BACKUP.GAIC.ORG
Updated Date: 05-nov-2001

First we will do a scan of the firewall itself. We will do this to see if the firewall has any open ports. To do this we will use "nmap". Network Mapper (better known as nmap) can be used to scan a host (or network of hosts) for running services (ports), determine if ports are being filtered, and OS Fingerprinting.

Complete nmap **syntax** can be found using man pages.

-sS – TCP Stealth, this will only open half of the TCP three-way handshake when scanning
-v – Verbose
-O – Do an OS fingerprint
-P0 – This will tell nmap to NOT ping the host before attempting port scan
-p – Range of ports to be scanned (ex. 1 or 1-10 or 1,5,7-100)
-oN – Create a human readable file with program output

$nmap –sS –v –O –P0 –p 1-65535 –oN giac_audit/fire_os_scan 209.163.X.254

Real Scan Results (note: actual scan not run against primary firewall)

root@greenbox:~# nmap -sS –v -O -P0 -p 1-100 –oN fire_os_scan 207.163.X.253

Starting nmap V. 2.53 by fyodor@insecure.org (www.insecure.org/nmap/)
Initiating SYN half-open stealth scan against (207.163.X.253)
The SYN scan took 271 seconds to scan 100 ports.
Warning: No TCP ports found open on this machine, OS detection will be MUCH less reliable

All 100 scanned ports on (207.163.X.253) are: filtered

Too many fingerprints match this host for me to give an accurate OS guess
TCP/IP fingerprint:
T5(Resp=N)
T6(Resp=N)
T7(Resp=N)
PU(Resp=N)
Nmap run completed -- 1 IP address (1 host up) scanned in 271 seconds

The reason for using the –P0 option in all the nmap commands is that auditors know ICMP is not allowed; therefore if it is not used the nmap scan will fail.
Nmap shows that all ports scanned were "filtered". Ports can have three states in nmap:

open – port is listening (most of the time means, received a SYN/ACK)
closed – port is not listening (received a RST)
filtered – Either nothing was received back or and got an ICMP administratively unreachable

Also as we can see from the output that with no listening ports it was unable to do an OS Fingerprint.

Next auditors will use a few scans to make sure the firewall rule-set is working. To do this auditors will need to use a sniffer. They use a sniffer to see if packets generated while scanning

made it through the firewall. In our example they use "tcpdump" on Linux. On the LAN side of the firewall auditors will setup a sniffer. The following command is used to turn tcpdump on and have it filter on the ip address of the machine auditors are using to conduct the scan. Using a filter will help to insure that only traffic that is related to the audit is captured.

Complete tcpdump **syntax** can be found using man pages.

$tcpdump src host attack.host

Now that the sniffer is listening auditors can begin the scanning process. They will scan all ports for all hosts on the subnet. Then check this output against what should be allowed. Things like web servers should only accept tcp ports 80 and 443, dns udp port 53, etc…

$nmap –sS –v –P0 –p 1-65535 –oN giac_audit/fire_tcpS_scan 209.163.X.0/24

Now that auditors have tried to get through sending SYN packets with the "-sS" option, they will check to make sure the firewall will not let random ACK packets through using the "-sA" option. This will verify that our stateful features are working as we expect.

-sA – ACK scan will send packets with ACK bit set. Mostly to test if a firewall is stateful.

$nmap –sA –v –P0 –p 1-65535–oN giac_audit/fire_tcpA_scan 209.163.X.0/24

Completing this scan the auditors will move on to check the rule-set using udp scans. This will let us know if the firewall will allow UDP packets through.

-sU – Tells nmap to scan udp ports using a packet of length 0 bytes and look for a response

$nmap –sU –v –P0 –p 1-65535–oN giac_audit/fire_udp_scan 209.163.X.0/24

One scan left to complete the nmap scanning. Auditors use the "-g" argument to allow the scan to have specific source ports. They will make sure that the firewall will not let unwanted packets through based on source port number.

-g – Instructs nmap to use number as source port number

$nmap –sS –g80 –v –P0 –p 1-65535 –oN giac_audit/fire_srcprt_scan 209.163.X.0/24

Real Scan Results (note: actual scan not run against primary firewall)

root@greenbox:~# nmap -sU -g3546 -v –P0 -p 53 –oN fire_srcprt_scan 207.163.X.243

Starting nmap V. 2.53 by fyodor@insecure.org (www.insecure.org/nmap/)
Initiating FIN,NULL, UDP, or Xmas stealth scan against (207.163.X.243)
The UDP or stealth FIN/NULL/XMAS scan took 12 seconds to scan 1 ports.
Interesting ports on (207.163.X.243):
Port State Service

53/udp open domain
Nmap run completed -- 1 IP address (1 host up) scanned in 12 second

Source port numbers to be used are 80, 53, 443, 25, and 110. From the output above we can see that the device, DNS Server is accepting connections on UDP port 53.

Using the output from nmap and the traces picked up by tcpdump, auditors will be able to verify that the firewall rule-set is working as expected.

Non-Critical Business hours audit

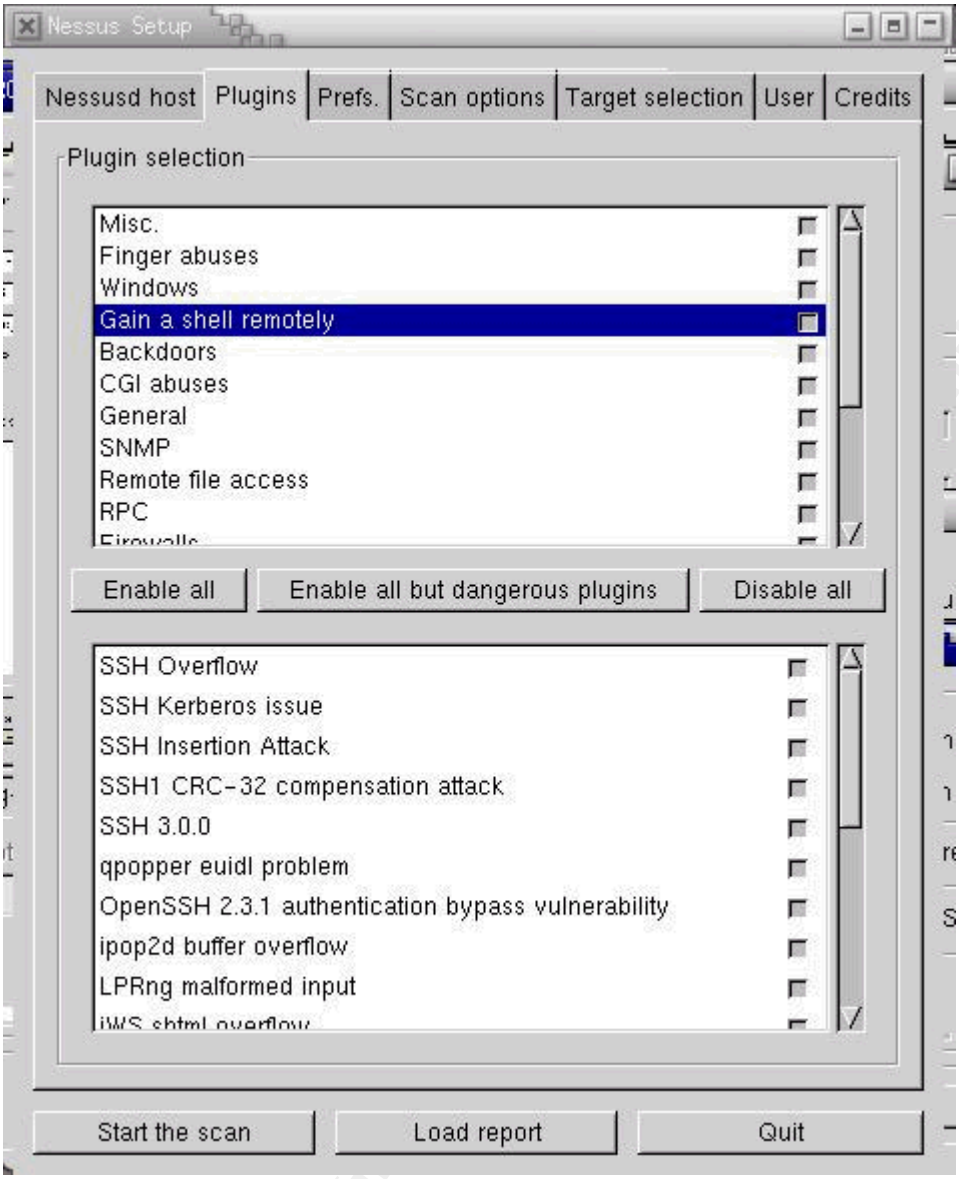
- Audit Firewall box
  - DOS on Slackware and Kernel 2.4.16
- Audit Firewall rule-set
  - DOS on rule-set

In this part of the audit they will try to crash the firewall and/or create situations where legitimate users are unable to gain access to GIAC resources.

First the auditors will use a program called "nessus" to complete some automated vulnerability scans. The biggest benefit to nessus is the ability to use the "plug-in" features. This allows users to add exploits to nessus. Nessus will accept two types of exploits, those written in NASL (Nessus Attack Scripting Language) and those written in C. Therefore nessus is modular and can grow and change as system vulnerabilities arise. Using this tool we are able to test things like SSH, BIND, Windows Vulnerabilities, and etc… We will use this utility to test attacks such as:

Nestea – sends oversized IP fragments
Land.c – sends tcp syn packet with targets ip in source and destination and same source and destination ports
synflood.c – create a syn flood
Any other Slackware 8.0 or Linux 2.4.X vulnerabilities that can be found

Nessus scan plug-in selection window



The scans and attacks performed from the outside of the network should also be performed from inside the network. This will help the administrators understand how vulnerable they are to attack from within. Administrators need to understand the need to watch more that just what is coming from the outside; unfortunately many threats come from within the internal network.

The final part of this phase is to analyze the logs and what happened while the system was under attack. It is necessary to have the administrators compile the logs generated by the firewall during the attacks and scans. These logs in conjunction with what was seen by the sniffer, program output from nmap, and nessus will help to formulate the final report that is provided in phase III.

PHASE III

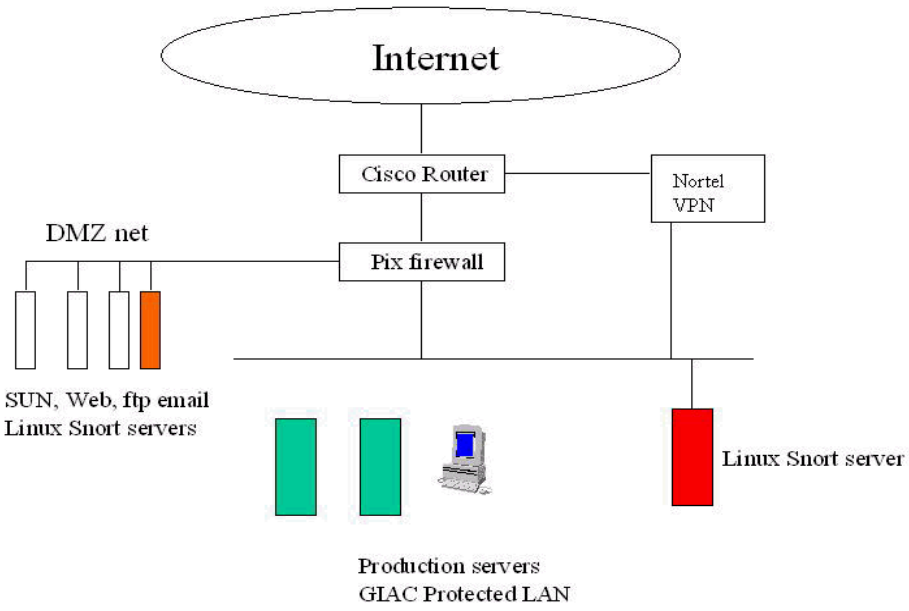
In phase three the auditors will sit down with management and technical staff and review what they found.

The audit team has found that the firewall performs the functions set forth by the GIAC security policy. The team has also found that the underlying operating system the firewall is built upon has been hardened and has no unnecessary services running. The auditing team does have a few network recommendations though:

- Have the firewall provide resets to unwanted packets instead of dropping the packets. This way when being scanned by a tool such as nmap, it will look as though every port on every machine is open.
- Move VPN device out from the LAN, placing it on a second services network.
- Use an Internal Mail server
- Do not allow incoming SSH from anywhere to anywhere within the GIAC LAN
- Use more than one firewall vendor, this will help to disallow one bug from compromising the entire network
- Would recommend using a proxy to also protect the machines in the services network. (web, dns, and mail servers)

Design Under Fire

Chap Wong's design has been chosen for this "design under fire."
Found at: http://www.giac.org/GCFW.php, picture below was taken from Wong's practical.



Firewall Device: Cisco Pix 5.0(3) → not stated will assume PIX 515

Firewall Vulnerabilities

1. Cisco PIX TACACS+ Denial of Service Vulnerability

"A problem with the PIX could allow a denial of service. PIX firewalls using TACACS+ are vulnerable to a resource starvation attack, which results in a denial of service. Upon receiving multiple requests for TACACS+ authentication from an unauthorized user, the firewalls resources can be exhausted. This causes the firewall to crash, requiring power cycling to resume regular service.
This makes it possible for a user from either the public or private side of the PIX to crash the firewall, and deny service to legitimate users.
All PIX Firewalls having configuration lines beginning with the following line are affected: pixfirewall# aaa authentication"
http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=2551

EXPLOIT:

while (true); do (wget http://external.system 2>/dev/null &); done

2. Cisco PIX Firewall SMTP Content Filtering Evasion Vulnerability

"During communication with an smtp server, if the "data" command is sent before the more important information is sent, such as "rcpt to", the smtp server will return error 503, saying that rcpt was required. The firewall, however, thinks everything is alright and will let everything through until recieving "<CR><LF><CR><LF>.<CR><LF>". It is then possible for the attacker to do whatever he wishes on the email server."
http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1698

EXPLOIT

From naif <naif@inet.it>'s Bugtraq post:

Here an example of what i could do exploiting this bug:
helo ciao
mail from: pinco@pallino.it
data (From here pix disable fixup)
expn guest (Now i could enumerate user
vrfy oracle and have access to all command)
help
whatever command i want
quit

3. Cisco Secure PIX Firewall Forged TCP RST Vulnerability

"A connection through a Cisco Secure PIX Firewall can be reset by a third party if the source and destination IP addresses and ports of the connection can be determined or inferred. This can be accomplished by sending a forged TCP Reset (RST) packet to the firewall, containing the same source and destination addresses and ports (in the TCP packet header) as the connection to be disrupted. The attacker would have to possess detailed knowledge of the connection table in the firewall (which is used to track outgoing connections and disallow any connections from the external network that were not initiated by an internal machine) or be able to otherwise determine the required IP address and port information to exploit this."
http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1454

EXPLOIT CODE:
http://downloads.securityfocus.com/vulnerabilities/exploits/pix_reset_state.c

We can use the TACACS+ bug to crash the PIX firewall. This will show how an unhappy or upset internal employee can cause havoc on the network. Given GIAC administrators have enabled AAA authentication on the PIX we can overload this feature. From any machine on the GIAC LAN use these commands to create a simple shell script:

$echo #!/bin/sh > attack.sh
$echo while (true); do (wget http://external.system 2>/dev/null &); done >> attack.sh

Now with this command we can overload the PIX firewall authentication process causing the box to run out of memory and ultimately crash.

$./attack.sh

Now the network is down and no one can enter or leave the GIAC Network. Users cannot even access the Services network.

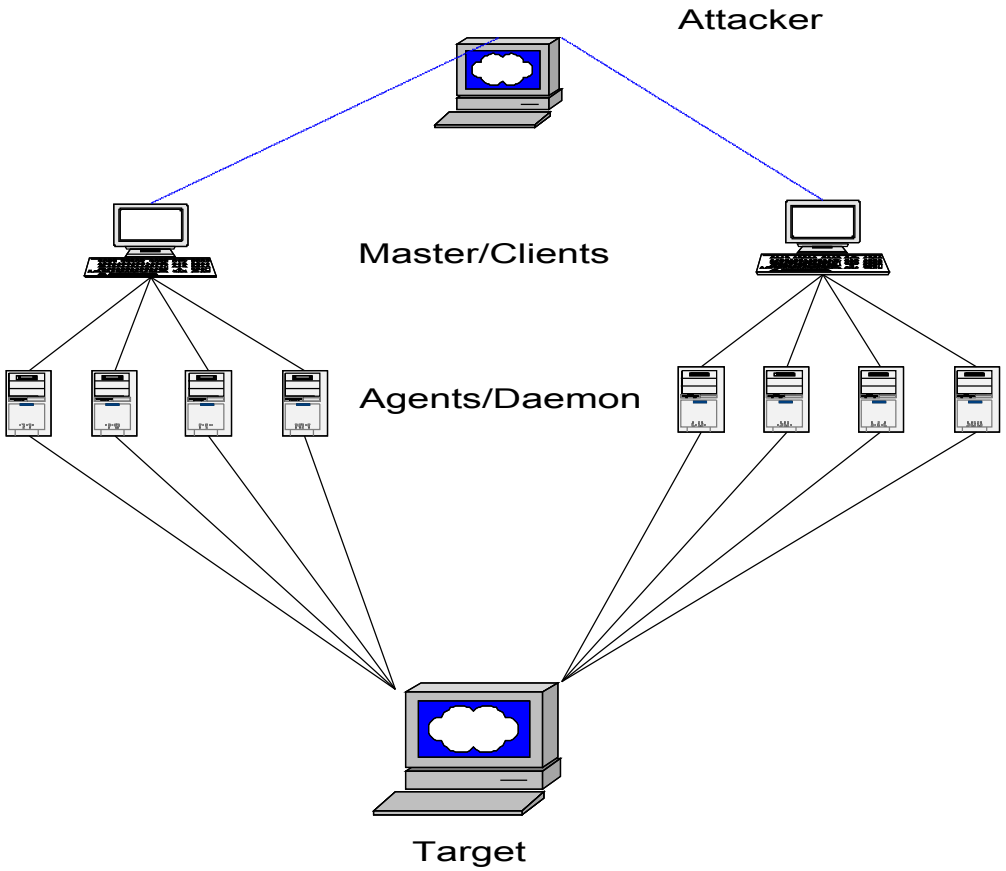
Distributed Denial of Service (DDoS)

To conduct our attack we will use TFN2K (Tribal Flood Network 2000), using this application will allow us to conduct more than one type of DDoS attack from a single program. The types of flooding this software supports are: UDP, ICMP, and TCP SYN.

A SYN flood is a manipulation of how TCP works. TCP uses a three-way handshake

1. Initiator send a SYN →
2. Recipient responds with a SYN/ACK ←
3. Initiator send a ACK →

Now that the TCP 3-way handshake is complete data communications will occur. What a SYN attack will do is send lots and lots of SYN's but will not respond with any ACK's. The Recipient of the SYN has a "half-open" connection and will wait X amount of time for the Initiator to send the ACK, but since this is an attack that ACK will never come. The goal is to make the device reach its limit of half-open connections and deny access to legitimate users or actually crash the box we are attacking.



You should install the client on more than one master machine; this way if one of the machines with the client software goes down or is reloaded you haven't lost communication with all agents. Once you have the client installed it will be necessary to compromise the computers connected to cable networks. Once this is accomplished it will be necessary to install the daemons on these machines. Now that we have the clients installed and the daemons installed we can choose a target(s) for these machines to attack.

According to Ross Oliver a machine connected to a cable network has a theoretical max of producing 200 syn/sec. We have 4 masters that control the other 46 agents, therefore these 46 machines can produce 9200 syn/sec(46x200 = 9200). Also, according to Cisco, the PIX 515-R supports around 50,000 simultaneous connections. Doing a little math with our theoretical 9200 syn/sec we can gather it will take 6sec (50000 / 9200 ~ 6) to reach the capacity of the PIX before it will deny access to legitimate users. Now this is completely theoretical and may take longer. At the point all 50,000 connections are filled the PIX firewall will not accept any more connections, denying legitimate users from making connections. Also, it is possible that a target machine behind the firewall has a TCP/IP stack that incorrectly handles many "half-open" connections. This can also help the DDoS crash the target host.

Prevention of DDoS

- Use of Application layer proxy to help detect fingerprint of TFN2K or other DDoS tools
- Disable ICMP into or out of network

- Only allow specific TCP and UDP ports in and out to limit TCP and UDP traffic
- Filter IANA Reserved and RCF1918 addresses
- Routers can use tcp intercept to aggressively reset half-open connections or to actually handle the connections itself and not pass them on to the destination host until the 3-way handshake is complete.

Compromise an Internal Host

First I would use "whois" to do a lookup on the domain name. This will tell me who owns and manages the domain. If I am lucky we can get the administrators name. If we have the administrators' name, hopefully it is correct, we can use it to do some Social Engineering.

$whois giac.org
Domain Name: GAIC.ORG
Registrar: NETWORK SOLUTIONS, INC.
Whois Server: whois.networksolutions.com
Referral URL: http://www.networksolutions.com
Name Server: NS1.GIAC.ORG
Name Server: NS.BACKUP.GAIC.ORG
Updated Date: 05-nov-2001

The whois didn't yield as much information as we could've hoped, but we know the primary and secondary DNS server names. Using a simple utility, "nslookup", we will be able to get the ip address of those machines as shown below.

$ nslookup giac.org
Name Server: dns.attacker.net
Address: 132.241.X.57

Non-authoritative answer:
Name: ns1.giac.org
Address: 2.2.2.7

Now that we know the IP address of the DNS server for GIAC Enterprises we will try and do a zone transfer. If this works this will give us the name and IP address for the machines contained in the giac.org zone. This information would be highly useful. Having this information will make it easy to create a map of the GIAC network.

$ nslookup
Default Name Server: dns.attacker.net
Address: 132.241.X.57

> server 2.2.2.7

Default Name Server: ns1.giac.org
Address: 2.2.2.7

> set type=any
> ls -d ns1.giac.org. > zone_transfer
[ns1.giac.org]
Received 0 records.
***** Can't list domain ns1.giac.org.: Query refused**

Unfortunately the administrators have disabled this feature. Also we would try and see if we could do a transfer from the secondary dns server(s). While I am still logged onto the primary dns server I will be sure to check the version of BIND it is running(assuming it is running BIND).

>set class=chaos
>set type=txt
>version.bind
Server: ns1.giac.org
Address: 2.2.2.7
VERSION.BIND text = "8.2.1"
>

Now that I know the machine is using BIND, I can get onto the Internet and look for vulnerabilities. Quickly I have more than one hit on this version of BIND. Since we are coming from a remote location there are two great remote buffer overflow vulnerabilities.

Vulnerability VU#196945 ISC BIND 8 contains buffer overflow in transaction signature (TSIG) handling code
http://packetstorm.widexs.nl/0101-exploits/bind-tsig.c

CERT Advisory CA-1999-14 – includes named 8.2/8.2.1 NXT Remote Overflow
http://www.securiteam.com/exploits/3T5Q5QAQ0K.html

If BIND is running as root both of these exploits will give us root access. Both of these vulnerabilities will use UDP 53, therefore most firewalls will not stop these exploits. Once we have root or some other account on this box we can launch off of here to attack other parts of the network exploiting vulnerabilities on other boxes. If other boxes share the Sun Solaris operating system in common with those in the services network we can use the following exploit to gain root access on those systems.

CERT Advisory CA-2001-34 Buffer Overflow in System V Derived Login
http://downloads.securityfocus.com/vulnerabilities/exploits/smash_bin_login.c

Normally we could've used a program like nmap and hping to do more network mapping and OS fingerprinting, but with the "gold mine" we have found in the DNS server those tools were unnecessary in mapping the network from an external location. We can download tools such as

this, along with a rootkit and backdoor once inside. These tools will help us map the network from the inside, while also allowing us access back into the system without having to execute the exploit again. We need to be careful with these tools though because the administrator is running IDS and could possibly already be on to us.

Protection from above vulnerabilities

- Update BIND to patched versions
- Keep IDS signatures up to date
- Update Sun Solaris

NOTE: Constant security updates and vulnerability awareness was addressed in Chap Wong's paper. Therefore, theoretically in an ideal situation the administrators of the network would have been aware of the vulnerabilities and been patched, making this particular attack not possible.

NOTE on OS Fingerprinting:

This is a technique of sending specially crafted packets to a device and seeing what the response or lack of response is. There are specific areas of the TCP/IP Stack implementation that were left up to vendors and by eliciting a response from these areas, we are able to "guess" what OS is running on the remote machine. For instance, Windows NT 4.0 may react to a packet with the SYN and RST bits set differently than the Linux Kernel 2.2.5 or even Windows 2000. By knowing the remote OS we can look for exploits that are applicable to that particular machine.

Appendix A
IPTABLES TUTORIAL

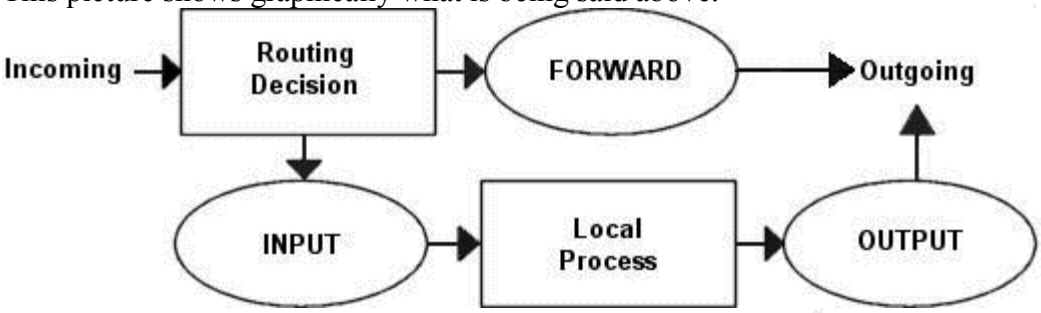
Syntax
iptables [table] <command> <chain name> <match conditions> <jump>

iptables is based off the idea that each packet that enters the computer belongs in a specific table. There are three built-in tables but for this discussion we are only going to talk about the filter table. The filter table is the default in the commands, therefore when entering a command it is not necessary to declare the table as filter. This table is sub-divided into multiple chains. The chains are sets of rules that are evaluated in order. If a packet matches the rule, then the action associated with that rule is executed. The order of the rules is important due to the sequential traversal of each chain. In the filter table all packets fall into one of three chains: INPUT, OUPUT, or FORWARD.

Built-in chains

FORWARD – packets not generated nor destined to this device. Only used if device is acting as a router or firewall
INPUT – packets that are destined to this device
OUTPUT – packets that were generated by this device

This picture shows graphically what is being said above.



Picture taken from http://www.boingworld.com/workshops/linux/iptables-tutorial/iptables-tutorial/iptables-tutorial.html

<command>
The command portion of the iptables syntax is intended to tell iptables what to do with the rest of the line we are giving it. Commands include things like: add, delete, insert, zero and so on to the specified chain. Below we will only talk about a few of the most common commands.

A – this is used to add to the bottom of the chain.
D – this is used to delete a rule from the chain. This can be accomplished by specifying the exact syntax of the rule or by giving the rule number in the chain
F – this is used to flush (remove) all the rules of a specified chain
L – this is used to list the rules of a specified chain, or if left blank will show all rules in table
N – this is used to add a new chain
X – this is used to delete a chain from a table. The chain must be empty.
P – this is used to create a default policy for a chain. If packet meets none of the rules in the chain this will happen. DROP, ACCEPT, or REJECT.

Now that we understand how to add, delete and so on from a chain and table, lets begin to talk about matching specific packets

<match condition>
Match uses specific parts of a packet to determine if we are interested in the packet. Things such as protocol (tcp, udp…), destination and source ports, addresses, and so on are used in this comparison. This section is where we define those parameters to act on.

-p – this stands for protocol, this will allow filtering based on protocol. By default it iptables understands tcp, udp, icmp, and all protocols defined in /etc/protocols. This

will also accept numeric values for the protocol (17 for udp…)
-f – this matches fragments, after the first packet.
-d – this is the destination ip address. can be written using subnet mask or bits used in subnet mask. 192.168.1.0/255.255.255.0 or 192.168.1.0/24
-s – this is the source ip address
-i – this is the interface the packet entered on
-o – this is the interface the packet will exit on

The following command will match all packets destined to this device coming from the eth0 interface that have a protocol of tcp.

iptables –A INPUT –i eth0 –p tcp

There are also more specific matching criteria for predefined protocols such as tcp, udp, and icmp. Both tcp and udp have these extended matching criteria:
--sport *ports* – source port. this can use names if defined in /etc/services or can use just port number. can also search sequential number of port if us colon ":". ex. 135:139
--dport *ports* – destination port

tcp also has:
--tcp-flags – this parameter allows checking of what flags are set. first define the flags we want to look at, then define the flags we want to be high. options SYN, FIN, ACK, URG, PSH, RST, or ALL. example ALL SYN,FIN
--syn – this parameter works the same as "--tcp-flags ALL SYN"
--tcp-options #– follow this with # and if options adds up to this value have a match

icmp has:
--icmp-type # – this allows defining of icmp message type

We can now further extend our earlier command to include port numbers. The command will allow packets coming from a high port number to a destination port of 80(HTTP). This is representative of traffic coming to a web server. Request from a client should have a source port number higher than 1024.

iptables –A INPUT –I eth0 –p tcp –sport 1024:65535 –dport 80

There are some more generic matching conditions used to do things like stateful filtering and rate limiting of packets. To access these features we will use the "-m" option. Following are the necessary options to allowing rate limiting, stateful filtering, and multiport matching. These are not all the possible options.

Rate limiting
This option will allow limiting the number of matches. This can be used to limit the number of log messages generated in a given time period or to combat syn floods. Limit defaults to 3/h and limit-burst defaults to 5/h.
Match on limit with "-m limit"

--limit *#/{s | m | h}* – limiting can be done in the number per second, minute or hour.
--limit-burst #– cause limit above to kick in after it reaches this burst rate
Command will cause a limit of 1 per second to be implemented after it has bursted to 3 per second

"-m limit --limit 1/s --limit-burst 3/s"

Stateful Filtering
This option will cause the firewall to create a state table, allowing the firewall to track connection. Tracking connections is an important feature that helps to make a firewall dynamic. This will allow the administrator to block ports, unless the first packet in the communication channel originated from the internal network.
Match on state with "-m state"

--state – There are four possible states
NEW – Packet that initiates session
ESTABLISHED – Active session packets (packets seen in both directions)
RELATED – Packet associated with connection → used for streams such as ftp and icmp
INVALID – Packets that cannot be associated with a state

Command will match packets that are part of an established or related session.

"-m state --state ESTABLISHED,RELATED"

**Note: It is possible for a packet without the SYN bit set to be considered "NEW" state, therefore don't make the mistake of thinking that NEW == only packets with just SYN bit set. Rule to get rid of non syn NEW packets:

"iptables –A CHAIN –p tcp -- ! syn –m state --state NEW –j DROP"

Multiport
This option will allowing matching a protocol on more than one port or discontinuous ports.
Match on multiport with "-m multiport"

--dport *port[,port…]* – Allows multiple destination port matches
--sport *port[,port…]* – Allows multiple destination port matches

Command will match packets going to ports 80 and 443, typical for web server.

"-m multiport --dport 80,443"

<jump>
Now we will discuss the actions that can be taken when one of these expressions is matched. This part of the command will always start with "-j" and can have take one of three built in options following it or a user defined chain may follow.

Built-in Targets

ACCEPT – Immediately accepts the packet and **quits** traversing chain
DROP – Immediately drops the packet and **quits** traversing chain
REJECT – **Quits** traversing chain and will send type of packet specified with reject options (can only be used from build-in chains)
LOG – Will log the specified packet and execute options, then **returns** to chain

Reject options
--reject-with *packet-type* – many options of packets to send
icmp-net-unreachable
icmp-host-unreachable
icmp-port-unreachable
icmp-proto-unreachable
icmp-net-prohibited
icmp-host-prohibited
echo-reply -- can be used if rule specifies ICMP ping packet
tcp-reset – can be used if rule specifies only TCP protocol

Log options
--log-level *level* -- Level of logging
--log-prefix *prefix* -- Allows prefixing up to 29 characters to log message. Good for parsing of log files for information you are interested in.
--log-tcp-sequence -- Log TCP sequence numbers.
--log-tcp-options -- Log options from the TCP packet header.
--log-ip-options -- Log options from the IP packet header.

Following commands will log any packets that are coming into the box with only the syn bit set. It will place in the log INCOMING SYN so that at a later time we can "grep" on that and get all logs that were incoming syn packets. Once it has logged the packet it will reject the packet with a RST packet.
"iptables –A INPUT –p tcp --syn –j LOG --log-prefix "INCOMING SYN""
"iptables –A INPUT –p tcp --syn –j REJECT--reject-with tcp-reset"

Hopefully this tutorial has given you a basic understanding of how iptables works and how to create simple iptables commands.

APPENDIX B

1.**Complete Border Router Configuration** (actual config is from a Cisco 1602)

```
Border_Router#sh run
Building configuration...

Current configuration:
!
version 12.2
no service pad
service timestamps debug uptime
service timestamps log uptime
service password-encryption
!
```

```
hostname Border_Router
!
aaa new-model
aaa authentication login TELNET group tacacs+ local
enable secret 5 $1$V7UZ$pb4cxmVF4Nfcs8sAH8CqM1
!
username AdMin password 7 03255F260F01287F5A1B18111800
!
ip subnet-zero
no ip source-route
no ip finger
no ip domain-lookup
no ip bootp server
!
interface Ethernet0
 description Connected to FIREWALL
 ip address 207.163.X.254 255.255.255.252
 ip access-group 115 in
 ip access-group 15 out
 no ip redirects
 no ip unreachables
 no ip proxy-arp
!
interface Serial0
 no ip address
 no ip redirects
 no ip unreachables
 no ip proxy-arp
 shutdown
!
interface Serial1
 description Connected to ISP
 ip address 192.168.5.2 255.255.255.0
 ip access-group 5 in
 no ip redirects
 no ip unreachables
 no ip proxy-arp
!
ip classless
ip route 0.0.0.0 0.0.0.0 Serial1
ip route 207.163.X.0 255.255.255.0 Ethernet0
no ip http server
!
access-list 5 deny   10.0.0.0 0.255.255.255
access-list 5 deny   172.16.0.0 0.15.255.255
access-list 5 deny   192.168.0.0 0.0.255.255
access-list 5 deny   127.0.0.0 0.255.255.255
access-list 5 deny   207.163.X.0 0.0.0.255
access-list 5 deny   1.0.0.0 0.255.255.255
access-list 5 deny   2.0.0.0 0.255.255.255
access-list 5 deny   5.0.0.0 0.255.255.255
access-list 5 deny   7.0.0.0 0.255.255.255
access-list 5 deny   23.0.0.0 0.255.255.255
access-list 5 deny   27.0.0.0 0.255.255.255
access-list 5 deny   31.0.0.0 0.255.255.255
access-list 5 deny   36.0.0.0 0.255.255.255
```

```
access-list 5 deny   37.0.0.0 0.255.255.255
access-list 5 deny   39.0.0.0 0.255.255.255
access-list 5 deny   41.0.0.0 0.255.255.255
access-list 5 deny   42.0.0.0 0.255.255.255
access-list 5 deny   58.0.0.0 0.255.255.255
access-list 5 deny   59.0.0.0 0.255.255.255
access-list 5 deny   60.0.0.0 0.255.255.255
access-list 5 deny   69.0.0.0 0.255.255.255
access-list 5 deny   70.0.0.0 0.255.255.255
access-list 5 deny   71.0.0.0 0.255.255.255
access-list 5 deny   72.0.0.0 0.255.255.255
access-list 5 deny   73.0.0.0 0.255.255.255
access-list 5 deny   74.0.0.0 0.255.255.255
access-list 5 deny   75.0.0.0 0.255.255.255
access-list 5 deny   76.0.0.0 0.255.255.255
access-list 5 deny   77.0.0.0 0.255.255.255
access-list 5 deny   78.0.0.0 0.255.255.255
access-list 5 deny   79.0.0.0 0.255.255.255
access-list 5 deny   82.0.0.0 0.255.255.255
access-list 5 deny   83.0.0.0 0.255.255.255
access-list 5 deny   84.0.0.0 0.255.255.255
access-list 5 deny   85.0.0.0 0.255.255.255
access-list 5 deny   86.0.0.0 0.255.255.255
access-list 5 deny   87.0.0.0 0.255.255.255
access-list 5 deny   88.0.0.0 0.255.255.255
access-list 5 deny   89.0.0.0 0.255.255.255
access-list 5 deny   90.0.0.0 0.255.255.255
access-list 5 deny   91.0.0.0 0.255.255.255
access-list 5 deny   92.0.0.0 0.255.255.255
access-list 5 deny   93.0.0.0 0.255.255.255
access-list 5 deny   94.0.0.0 0.255.255.255
access-list 5 deny   95.0.0.0 0.255.255.255
access-list 5 deny   96.0.0.0 0.255.255.255
access-list 5 deny   97.0.0.0 0.255.255.255
access-list 5 deny   98.0.0.0 0.255.255.255
access-list 5 deny   99.0.0.0 0.255.255.255
access-list 5 deny   100.0.0.0 0.255.255.255
access-list 5 deny   101.0.0.0 0.255.255.255
access-list 5 deny   102.0.0.0 0.255.255.255
access-list 5 deny   103.0.0.0 0.255.255.255
access-list 5 deny   104.0.0.0 0.255.255.255
access-list 5 deny   105.0.0.0 0.255.255.255
access-list 5 deny   106.0.0.0 0.255.255.255
access-list 5 deny   107.0.0.0 0.255.255.255
access-list 5 deny   108.0.0.0 0.255.255.255
access-list 5 deny   109.0.0.0 0.255.255.255
access-list 5 deny   110.0.0.0 0.255.255.255
access-list 5 deny   111.0.0.0 0.255.255.255
access-list 5 deny   112.0.0.0 0.255.255.255
access-list 5 deny   113.0.0.0 0.255.255.255
access-list 5 deny   114.0.0.0 0.255.255.255
access-list 5 deny   115.0.0.0 0.255.255.255
access-list 5 deny   116.0.0.0 0.255.255.255
access-list 5 deny   117.0.0.0 0.255.255.255
access-list 5 deny   118.0.0.0 0.255.255.255
access-list 5 deny   119.0.0.0 0.255.255.255
```

```
access-list 5 deny   120.0.0.0 0.255.255.255
access-list 5 deny   121.0.0.0 0.255.255.255
access-list 5 deny   122.0.0.0 0.255.255.255
access-list 5 deny   123.0.0.0 0.255.255.255
access-list 5 deny   124.0.0.0 0.255.255.255
access-list 5 deny   125.0.0.0 0.255.255.255
access-list 5 deny   126.0.0.0 0.255.255.255
access-list 5 deny   197.0.0.0 0.255.255.255
access-list 5 deny   221.0.0.0 0.255.255.255
access-list 5 deny   222.0.0.0 0.255.255.255
access-list 5 deny   223.0.0.0 0.255.255.255
access-list 5 deny   240.0.0.0 0.255.255.255
access-list 5 deny   241.0.0.0 0.255.255.255
access-list 5 deny   242.0.0.0 0.255.255.255
access-list 5 deny   243.0.0.0 0.255.255.255
access-list 5 deny   244.0.0.0 0.255.255.255
access-list 5 deny   245.0.0.0 0.255.255.255
access-list 5 deny   246.0.0.0 0.255.255.255
access-list 5 deny   247.0.0.0 0.255.255.255
access-list 5 deny   248.0.0.0 0.255.255.255
access-list 5 deny   249.0.0.0 0.255.255.255
access-list 5 deny   250.0.0.0 0.255.255.255
access-list 5 deny   251.0.0.0 0.255.255.255
access-list 5 deny   252.0.0.0 0.255.255.255
access-list 5 deny   253.0.0.0 0.255.255.255
access-list 5 deny   254.0.0.0 0.255.255.255
access-list 5 deny   255.0.0.0 0.255.255.255
access-list 5 permit any
access-list 10 permit 207.163.X.12
access-list 10 permit 207.163.X.15
access-list 11 deny   any
access-list 15 permit 207.163.X.0 0.0.0.255
access-list 15 deny   any log
access-list 115 permit ip any 207.163.X.0 0.0.0.255
access-list 150 permit tcp any 0.0.0.0 255.255.255.0
no cdp run
tacacs-server host 207.163.X.16
tacacs-server key IAMSECRET
banner motd ^C
UNAUTHORIZED ACCESS TO THIS DEVICE IS PROHIBITED
^C
!
line con 0
 exec-timeout 5 30
 transport input none
 password donthurtme
line vty 0 4
 access-class 10 in
 access-class 11 out
 exec-timeout 5 30
 login authentication TELNET
!
end
```

2.**Complete Primary Firewall Configuration**

```
INET_FACE = eth1
LAN_FACE = eth0
SER_FACE = eth2

BRD_ROUTER = 207.163.X.1
MAIL_SERVER = 207.163.X.244
WWW_SERVER = 207.163.X.242
DNS_SERVER =207.163.X.243
INTERNAL_DNS = 207.163.X.15
TAC_SERVER = 207.163.X.16
VPN_SERVER = 207.163.X.17
DBASE = 207.163.X.18

INTERNAL_NET = 207.163.X.0/24
NOT_INTERNAL = 207.163.X.240/28

IANA_PRIVATE="
        0.0.0.0/8 1.0.0.0/8 2.0.0.0/8 \
        5.0.0.0/8 \
        7.0.0.0/8 \
        23.0.0.0/8 \
        27.0.0.0/8 \
        31.0.0.0/8 \
        36.0.0.0/8 37.0.0.0/8 \
        39.0.0.0/8 \
        41.0.0.0/8 42.0.0.0/8 \
        58.0.0.0/8 59.0.0.0/8 60.0.0.0/8 \
        67.0.0.0/8 68.0.0.0/8 69.0.0.0/8 70.0.0.0/8 71.0.0.0/8 72.0.0.0/8 73.0.0.0/8 \
        74.0.0.0/8 75.0.0.0/8 76.0.0.0/8 77.0.0.0/8 78.0.0.0/8 79.0.0.0/8 80.0.0.0/8 \
        81.0.0.0/8 82.0.0.0/8 83.0.0.0/8 84.0.0.0/8 85.0.0.0/8 86.0.0.0/8 87.0.0.0/8 \
        88.0.0.0/8 89.0.0.0/8 90.0.0.0/8 91.0.0.0/8 92.0.0.0/8 93.0.0.0/8 94.0.0.0/8 \
        95.0.0.0/8 96.0.0.0/8 97.0.0.0/8 98.0.0.0/8 99.0.0.0/8 100.0.0.0/8 101.0.0.0/8 \
        102.0.0.0/8 103.0.0.0/8 104.0.0.0/8 105.0.0.0/8 106.0.0.0/8 107.0.0.0/8 \
        108.0.0.0/8 109.0.0.0/8 110.0.0.0/8 111.0.0.0/8 112.0.0.0/8 113.0.0.0/8 \
        114.0.0.0/8 115.0.0.0/8 116.0.0.0/8 117.0.0.0/8 118.0.0.0/8 119.0.0.0/8 \
        120.0.0.0/8 121.0.0.0/8 122.0.0.0/8 123.0.0.0/8 124.0.0.0/8 125.0.0.0/8 \
        126.0.0.0/8 127.0.0.0/8 \
        197.0.0.0/8 \
        201.0.0.0/8 \
        219.0.0.0/8 220.0.0.0/8 221.0.0.0/8 222.0.0.0/8 223.0.0.0/8 \
        240.0.0.0/8 241.0.0.0/8 242.0.0.0/8 243.0.0.0/8 244.0.0.0/8 245.0.0.0/8 \
        246.0.0.0/8 247.0.0.0/8 248.0.0.0/8 249.0.0.0/8 250.0.0.0/8 251.0.0.0/8 \
        252.0.0.0/8 253.0.0.0/8 254.0.0.0/8 255.0.0.0/8
        10.0.0.0/8 172.16.0.0/12 192.168.0.0/16 224.0.0.0/4 240.0.0.0/5"


################## CLEAN/CREATE TABLES #################

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

```
iptables -P FORWARD DROP

iptables -N LANOUT
iptables -F LANOUT
iptables -N LANIN
iptables -F LANIN
iptables -N SERVICEIN
iptables -F SERVICEIN
iptables -N SERVICEOUT
iptables -F SERVICEOUT
iptables -N CHECK_SCAN
iptables -F CHECK_SCAN
iptables -N SRC_CHECK
iptables -F SRC_CHECK
iptables -N DST_CHECK
iptables -F DST_CHECK


################## LOOPBACK TRAFFIC #################
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

################## SEND TRAFFIC TO USR TABLES ################
#
#NOTE: Traffic from LAN to SER is filtered by LANOUT
#     Traffic from SER to LAN is filtered by SEROUT
#     Traffic from NET to SER is filtered by SERIN
#     Traffic from SER to NET is filtered by SEROUT
#     Traffic from NET to LAN is filtered by LANIN
#     Traffic from LAN to NET is filtered by LANOUT
#
iptables -A FORWARD -p tcp --syn -j SYN_FLOOD
iptables -A FORWARD -p tcp -j CHECK_SCAN
iptables -A FORWARD -i $LAN_FACE -j LANOUT
iptables -A FORWARD -i $INET_FACE -o $LAN_FACE -j LANIN
iptables -A FORWARD -i $INET_FACE -o $SER_FACE -j SERIN
iptables -A FORWARD -i $SER_FACE -j SEROUT


################## LANIN #################

iptables -A LANIN -j SRC_CHECK
iptables -A LANIN -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A LANIN -p tcp --dport 49 -s $BRD_ROUTER -d $TAC_SERVER \
        -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A LANIN -p udp --dport 500 -d $VPN_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A LANIN -p 50 -d $VPN_SERVER -m state --state \
        NEW,ESTABLISHED -j ACCEPT
iptables -A LANIN -p tcp,udp --dport ! 1024:65535 -j DROP
iptables -A LANIN -m state --state INVALID -j DROP
iptables -A LANIN -m state --state ESTABLISHED -j ACCEPT
iptables -A LANIN -m limit --limit 5/minute -j LOG \
        --log-prefix "LANIN nondrop"
iptables -A LANIN -j DROP
```

```
################## LANOUT ##################

iptables -A LANOUT -s ! $INTERNAL_NET -j LOG --log-prefix "SPOOFING"
iptables -A LANOUT -s ! $INTERNAL_NET -j DROP
iptables -A LANOUT -j DST_CHECK
iptables -A LANOUT -p tcp --dport 22 -m state --state \
        NEW,ESTABLISHED -j ACCEPT
iptables -A LANOUT -p tcp --dport 80 -m state --state \
        NEW,ESTABLISHED -j ACCEPT
iptables -A LANOUT -p tcp --dport 443 -m state --state \
        NEW,ESTABLISHED -j ACCEPT
iptables -A LANOUT -p tcp --dport 110 -d $MAIL_SERVER  -m state --state \
        NEW,ESTABLISHED -j ACCEPT
iptables -A LANOUT -p tcp --sport 49 -d $BRD_ROUTER -s $TAC_SERVER \
        -m state --state ESTABLISHED -j ACCEPT
iptables -A LANOUT -p tcp --sport 1521 -s $DBASE -d $WWW_SERVER -j ACCEPT
iptables -A LANOUT -p udp --dport 53 -s $INTERNAL_DNS -j ACCEPT
iptables -A LANOUT -p udp --sport 500 -s $VPN_SERVER -m state --state \
        ESTABLISHED -j ACCEPT
iptables -A LANOUT -p 50 -s $VPN_SERVER -m state --state ESTABLISHED -j ACCEPT
iptables -A LANOUT -m limit --limit 5/minute -j LOG \
        --log-prefix "LANOUT nondrop"
iptables -A LANOUT -j DROP



################## SERIN ##################

iptables -A SERIN SRC_CHECK
iptables -A SERIN -p tcp --dport 25 -d $MAIL_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A SERIN -p tcp --dport 80 -d $WWW_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A SERIN -p tcp --dport 443 -d $WWW_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A SERIN -p udp --dport 53 -d $DNS_SERVER -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A SERIN -m limit --limit 5/minute -j LOG \
        --log-prefix "SERIN nondrop"
iptables -A SERIN -j DROP


################## SEROUT ##################

iptables -A SEROUT -s ! NOT_INTERNAL -m limit --limit 5/minute \
        -j LOG --log-prefix "SERVICE SPOOFING"
iptables -A SEROUT -s ! NOT_INTERNAL -j DROP
iptables -A SEROUT -j DST_CHECK
iptables -A SEROUT -p tcp --sport 80 --dport 1024:65535 -s $WWW_SERVER \
        -m state --state ESTABLISHED -j ACCEPT
iptables -A SEROUT -p tcp --sport 443 --dport 1024:65535 -s $WWW_SERVER \
        -m state --state ESTABLISHED -j ACCEPT
iptables -A SEROUT -p tcp --sport 25 -s $MAIL_SERVER -m state \
        --state ESTABLISHED -j ACCEPT
```

```
iptables -A SEROUT -p tcp --sport 22 -d $INTERNAL_NET -m state \
        --state ESTABLISHED -j ACCEPT
iptables -A SEROUT -p tcp --sport 110 -d $INTERNAL_NET -s MAIL_SERVER -m state \
        --state ESTABLISHED -j ACCEPT
iptables -A SEROUT -p tcp --dport 1521 -s $WWW_SERVER -d $DBASE -j ACCEPT
iptables -A SEROUT -p udp --sport 53 --dport 1024:65535 -s $DNS_SERVER \
        -m state --state ESTABLISHED -j ACCEPT
#NEW connections out
iptables -A SEROUT -m state --state NEW -m limit --limit \
        5/minute -j LOG --log-prefix "SERVICE SYN"
iptables -A SEROUT -m state --state NEW -j DROP
iptables -A SEROUT -m limit --limit 5/minute -j LOG \
        --log-prefix "SEROUT nondrop"
iptables -A SEROUT -j DROP



################## CHECK_SCAN via TIM FOREMAN #################

#NMAP SCAN
iptables -A CHECK_SCAN -p tcp --tcp-flags ALL FIN,URG,PSH -m limit \
        --limit 5/minute -j LOG --log-prefix "NMAP_SCAN"
iptables -A CHECK_SCAN -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP

#SYN/RST
iptables -A CHECK_SCAN -p tcp --tcp-flags SYN,RST SYN,RST -m limit \
        --limit 5/minute -j LOG --log-prefix "SYN/RST"
iptables -A CHECK_SCAN -p tcp --tcp-flags SYN,RST SYN,RST -j DROP

#SYN/FIN
iptables -A CHECK_SCAN -p tcp --tcp-flags SYN,FIN SYN,FIN -m limit \
        --limit 5/minute -j LOG --log-prefix "SYN/FIN"
iptables -A CHECK_SCAN -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP

#ALL ALL
iptables -A CHECK_SCAN -p tcp --tcp-flags ALL ALL -m limit \
        --limit 5/minute -j LOG --log-prefix "ALL/ALL"
iptables -A CHECK_SCAN -p tcp --tcp-flags ALL ALL -j DROP

#ALL NONE
iptables -A CHECK_SCAN -p tcp --tcp-flags ALL NONE -m limit \
        --limit 5/minute -j LOG --log-prefix "ALL/NONE"
iptables -A CHECK_SCAN -p tcp --tcp-flags ALL NONE -j DROP

#DISALLOWS NON-SYN PACKETS FROM CREATING STATE ENTRY
iptables -A CHECK_SCAN -p tcp ! --syn -m state --state NEW -j DROP
#REJECT IDENT ALLOWING QUICKER RESPONSE TIMES
iptables -A CHECK_SCAN -p tcp --dport 113 -j REJECT --reject-with tcp-reset



################## SYN_FLOOD #################

iptables -A SYN_FLOOD -i $INET -m limit --limit 200/s -j RETURN
iptables -A SYN_FLOOD -i $INET -j LOG --log-prefix "INET-SYN-FLOOD"
iptables -A SYN_FLOOD -o $LAN -m limit --limit 300/s -j RETURN
```

```
iptables -A SYN_FLOOD -o $INET -j LOG --log-prefix "INTERNAL-SYN-FLOOD"
iptables -A SYN_FLOOD -j DROP

################## SRC_CHECK ################

for NET $IANA_PRIVATE;do
        iptables -A SRC_CHECK -s $NET -j DROP
done


################## DST_CHECK ################

for NET $IANA_PRIVATE;do
        iptables -A DST_CHECK -d $NET -j DROP
done
```

4. **Partner / Suppliers Border Router** (actual config is from a Cisco 1602)

```
PS_Gateway#sh run
Building configuration...

Current configuration:
!
version 12.2
no service pad
service timestamps debug uptime
service timestamps log uptime
service password-encryption
!
hostname PS_Gateway
!
aaa new-model
aaa authentication login TELNET tacacs+ local
enable secret 5 $1$V7UZ$pb4cxmVF4Nfcs8sAH8CqM1
!
ip subnet-zero
no ip source-route
no ip finger
no ip bootp server
no ip domain-lookup
!
!
crypto isakmp policy 10
 authentication pre-share
crypto isakmp key I_AM_KEY address 172.16.1.2
!
!
crypto ipsec transform-set AH_MD5 ah-md5-hmac
!
 !
 crypto map toPartorSup 10 ipsec-isakmp
 set peer 172.16.1.2
 set transform-set AH_MD5
 match address 155
!
```

```
!
process-max-time 200
!
interface Ethernet0
 description Connected to P/S Network
 ip address 172.21.5.1 255.255.255.0
 ip access-group 101 in
 no ip unreachables
 no ip proxy-arp
 shutdown
!
interface Serial0
 no ip address
 shutdown
!
interface Serial1
 ip address 172.16.1.1 255.255.255.252
 no ip redirects
 no ip unreachables
 no ip proxy-arp
 bandwidth 1024
 crypto map toPartorSup
!
ip classless
ip route 0.0.0.0 0.0.0.0 Null0
ip route 10.10.10.0 255.255.255.0 172.16.1.2
no ip http server
!
access-list 10 permit 172.21.5.10
access-list 10 permit 172.21.5.115
access-list 11 deny   any
access-list 101 permit ip 172.21.5.0 0.0.0.255 10.10.10.0 0.0.0.255
access-list 101 permit ip 172.21.5.0 0.0.0.255 110.10.10.0 0.0.0.255
access-list 101 permit ip 172.21.5.0 0.0.0.255 210.10.10.0 0.0.0.255
access-list 101 permit ip 172.21.5.0 0.0.0.255 120.10.10.0 0.0.0.255
access-list 101 permit ip 172.21.5.0 0.0.0.255 60.10.10.0 0.0.0.255
access-list 101 permit ip 172.2.5.0 0.0.0.255 30.10.10.0 0.0.0.255
access-list 101 deny   ip any any log
access-list 155 permit ip 172.21.5.0 0.0.0.255 10.10.10.0 0.0.0.255
no cdp run
tacacs-server host 172.21.5.30
tacacs-server key IAMSECRET2
banner motd ^CC
UNAUTHORIZED ACCESS TO THIS DEVICE IS PROHIBITED
^C
!
line con 0
 exec-timeout 5 30
 transport input none
line vty 0 4
 access-class 10 in
 access-class 11 out
 exec-timeout 5 30
 login authentication TELNET
!
end
```

4. Partner / Suppliers Firewall

```
LAN_FACE = eth0
PS_FACE = eth1

TRANSFER_HOST = 207.163.X.20
ADMIN_HOST = 207.163.X.12
PS_NET = 172.21.5.0/24
################## CLEAN/CREATE TABLES #################

iptables -F INPUT
iptables -P INPUT DROP
iptables -F OUTPUT
iptables -P OUTPUT DROP
iptables -F FORWARD
iptables -P FORWARD DROP

iptables -N LANOUT
iptables -F LANOUT
iptables -N LANIN
iptables -F LANIN
iptables -N CHECK_SCAN
iptables -F CHECK_SCAN


################## LOOPBACK TRAFFIC ################
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT


################## SEND TRAFFIC TO USR TABLES #################

iptables -A FORWARD -i $LAN_FACE -o $PS_FACE -j LANOUT
iptables -A FORWARD -i $PS_FACE -o $LAN_FACE -j LANIN


################## LANIN #################

iptables -A LANIN -s ! $PS_NET -m limit --limit 5/minute -j \
        LOG --log-prefix "PS_NET BAD_SRC"
iptables -A LANIN -s ! $PS_NET -j DROP
iptables -A LANIN -d ! $TRANSFER_HOST -m limit --limit \
        5/minute -j LOG --log-prefix "NONHOST PACKETS IN"
iptables -A LANIN -d ! $TRANSFER_HOST -j DROP
iptables -A LANIN -m multiport -sport ! 22,1521 -j DROP
iptables -A LANIN -m state --state INVALID -j DROP
iptables -A LANIN -m state --state ESTABLISHED -j ACCEPT
iptables -A LANIN -m limit --limit 5/minute -j LOG \
        --log-prefix "LANIN nondrop"
iptables -A LANIN -j DROP


################## LANOUT #################
```

```
iptables -A LANOUT -p tcp --dport 22 -s $ADMIN_HOST -d $PS_NET -m state \
        --state NEW,ESTABLISHED -j ACCEPT
iptables -A LANOUT -s ! $TRANSFER_HOST -m limit --limit \
        5/minute -j LOG --log-prefix "NONHOST PACKETS OUT"
iptables -A LANOUT -s ! $TRANSFER_HOST -j DROP
iptables -A LANOUT -d ! $PS_NET -j DROP
iptables -A LANOUT -p tcp --dport 1521 -m state --state \
        NEW,ESTABLISHED -j ACCEPT
iptables -A LANOUT -m limit --limit 5/minute -j LOG \
        --log-prefix "LANOUT nondrop"
iptables -A LANOUT -j DROP
```

BIBLIOGRAPHY

(Web links listed in text are not restated here)

Security Policy (mostly iptables)

http://www.iana.org/assignments/ipv4-address-space
http://www.sns.ias.edu/~jns/security/iptables/index.html#CONFIG
http://www.linuxsecurity.com/resource_files/firewalls/IPTables-Tutorial/iptables-tutorial/iptables-tutorial.html#AEN194

http://packetstorm.widexs.nl/papers/firewall/iptables.txt
http://www.telematik.informatik.uni-karlsruhe.de/lehre/seminare/LinuxSem/downloads/netfilter/iptables-HOWTO-6.html

http://monmotha.mplug.org/~monmotha/firewall/firewall/2.3/rc.firewall-2.3.8-pre2
http://timf.anansi-web.com/misc/iptables.html

Mason, Andrew G and Newcomb, Mark J "Cisco Secure Internet Security Solutions". Indianapolis: Cisco Press, 2001. pgs 36-68

Audit Architecture

http://www.insecure.org/nmap/
http://www.enteract.com/~lspitz/audit.html
http://hackpalace.com/hacking/unix/c/synflood.c
http://www.freebsd.org/index.html

Design under fire

http://www.usenix.org/events/sec01/invitedtalks/oliver.pdf → slide 24
http://www.zdnet.com/products/stories/reviews/0,4161,2448193,00.html
http://rr.sans.org/malicious/DNS_exploit.php
http://packetstorm.widexs.nl/distributed/TFN2k_Analysis.htm

http://www.cisco.com/warp/public/cc/pd/fw/sqfw500/prodlit/pix51_ds.htm

Exploit
http://rr.sans.org/unix/BIND8.php
McClure, Stuart and Scambray, Joel and Kurtz, George, "Hacking Exposed: Network Security Secrets and Solutions". Berkley: Osborne/McGraw-Hill, 1999.

Honors paper used as reference
http://www.giac.org/GCFW.php → Michael_Vars
http://www.giac.org/GCFW.php → Kofi_Arthiabah
http://www.giac.org/GCFW.php → Asad_Alsader