Global Information Assurance Certification Paper

Table of Contents 1
Robert_A_Smith_GCFW.doc 2

Robert A Smith

Rocky Mountain SANS JUNE 28 - JULY 3, 2001

GCFW Practical Version 1.5e

"An Extensible Corporate Security Architecture"

ASSIGNMENT 1- SECURITY ARCHITECTURE

- General Hardware and Software Specifications

GIAC Enterprises requires extensible security architecture due to its long-term predicted growth. This goal will be accomplished through the use of components, which are themselves extensible.

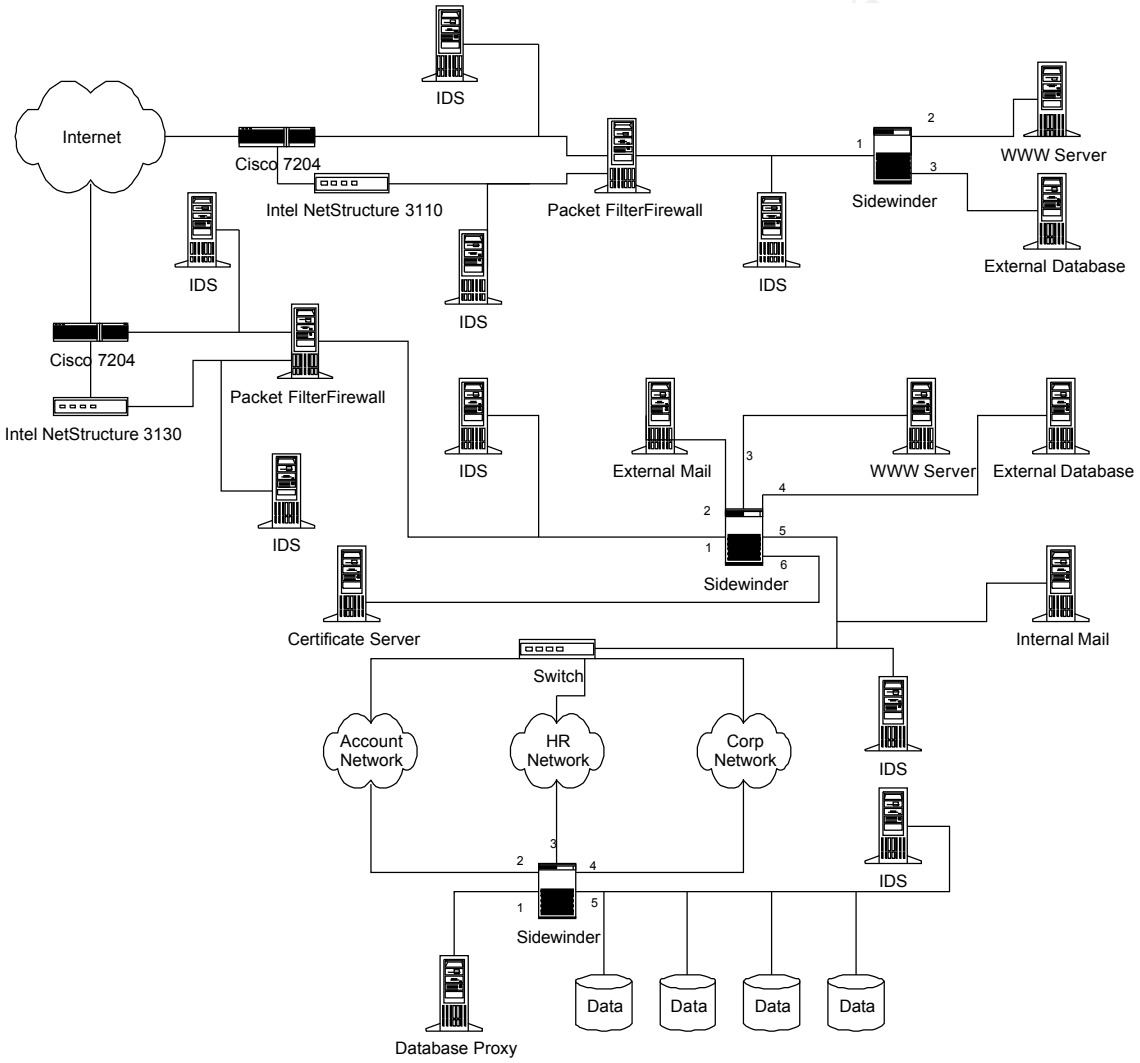
The first components of the architecture are the border routers, which are Cisco 7204 series routers with an eight port T1/E1 interface adapter in one slot and a two port Fast Ethernet adapter in another slot. This leaves two more slots for expandability if future needs arise.

The second components of the architecture are the packet filtering firewalls which are Pentium 1.0 Ghz personal computers with 512 MB RAM and three 100 Mbps Fast Ethernet cards. These run Red Hat 6.2 Linux as an operating system and use ipchains to filter packets. These systems have been configured as bastion hosts with minimal services running. Using good quality but common components such as these, and open source operating systems provides the opportunity to have spares available at low cost.

The third components of the architecture are the Sidewinder 5.1 firewalls. These are to be run on Dell PowerEdge 6400 servers with quad 900Mhz processors, 4 GB RAM, and four double port NICs. The Sidewinder firewalls will have split DNS enabled. Using Sidewinder for a proxy firewall gives us the opportunity to concentrate on getting the network up and running, each proxy firewall has three to six interfaces and it has the available NIC drivers (for dual NICs) and many configuration options. Sidewinder can run single or split DNS and we will be running split DNS at GIAC. The Sidewinder Firewall acts as the master DNS server with slave servers located where we want to place them.

The fourth components of the architecture are the Intel NetStructure VPN appliances. The architecture calls for using two different models depending upon the traffic expected at each location. The Intel NetStructure 3130 will be used in high traffic areas and the Intel NetStructure 3110 will be used in low traffic areas. The Intel NetStructure VPN client software will be loaded onto users computers who require remote access. All of the VPN tunnels will use Shiva Smart Tunneling (SST), not IPSEC, although the Intel NetStructure VPN Gateways will support IPSEC, and this feature will be available if a new trading

partner has existing IPSEC equipment. SST tunnels are easier to configure and more secure than IPSEC, since SST tunnels encrypt the complete packet including the IP headers. An Intel Certificate Authority will be used for authentication certificates and will be placed inside the corporate network.
The fifth components of the architecture are the Intrusion Detection Systems, which will use RedHat Linux 6.2 bastion hosts for the sensors. These will be Pentium 1.0 Ghz personal computers with 512 MB RAM and one 100 Mbps Fast Ethernet card. They will have Snort installed, logging to the local machine in two files, one of which is on a cdrom.



EXPLANATION OF DATA FLOW AND USER ACCESS

GIAC Enterprises has two WAN connections to the Internet. This decision was made to more easily segregate the users of the GIAC network, with whom we have different security concerns. The first of these connections is to provide access to the corporate e-commerce web site. The second connection is for the

corporate network, and provides the connection point for customers with corporate accounts, suppliers, partners, and remote users.

Connection 1

This connection is for the e-commerce web site, which needs to be publicly accessible. This is the web site users access for information about the company, orders for fortunes can be made by first time buyers and low volume customers who do not wish to open a corporate account. The data flowing in this network will be TCP/IP; protocols will be DNS, HTTP, HTTPS, and ICMP.

Starting at the border router, from the internet connection to the packet filter connection, ACLs will be applied to the inbound interface to filter out reserved IP ranges, loop back address, multicast address, 0.0.0.0 address and broadcast address, telnet, FTP, all login services (Rlogin, etc), Sun RPC and NFS, Net BIOS, Xwindows, Netbus, Backorifice , and most ICMP. All of these will be implemented with logging. On the outbound interface egress ACLs will be applied to prevent unwanted traffic from exiting the network. Again, reserved IP ranges, loop back address, multicast address, 0.0.0.0 and broadcast address, telnet, FTP, all login services (Rlogin, etc), Sun RPC and NFS, Net BIOS, Xwindows, Netbus, Backorifice, and most ICMP will be blocked, with logging performed. The third connection will have ACLs applied to block everything inbound and outbound except TCP ports 10027-10028 and UDP ports 2233,10025-10027**,** which carry encrypted traffic to the VPN.

The VPN appliance connected at the border router will be an Intel NetStructure 3110 whose only purpose is to provide a testing gateway. With the VPN situated here it will be apparent which part of the network is down if trouble arises. A remote client coming over the Internet via a separate dial in ISP connection will access the VPN gateway, if the gateway can be contacted then the WAN link and router to that point are up and operating. ICMP will be allowed only from the VPN gateway past the packet filter and Sidewinder to the various other points on the network for further testing.

The packet filtering router will block the same traffic as the border router and in addition will block other well known threats such as LDAP, SMTP, POP, IMAP, SSL, TCP and UDP ports below 20, NNTP, and other miscellaneous protocols on both the internet connection and the VPN connection. These will be blocked with logging enabled so the ACLs on the router can be confirmed and checked against the IDS log files.

The Sidewinder firewalls interface one will respond to exterior DNS queries, proxy HTTP and HTTPS to and from the web server (interface two), and proxy ICMP to and from our diagnostic address. Interface two will proxy HTTP and HTTPS communication between the Web server and interface one, and database communication between the Web server and interface three. Interface three will only proxy database communication between interfaces two and three.

Connection 2

This connection is for the corporate network, corporate clients, suppliers and partners. This is the connection corporate users access the Internet and send-receive email with. Corporate clients, suppliers, and partners are given a secure web site to connect to for ordering and supply purposes. The data flowing in this network will be TCP/IP; protocols will be DNS, HTTP, HTTPS, SMTP and ICMP.

Starting at the border router, from the internet connection to the packet filter connection, ACLs will be applied to the inbound interface to filter out reserved IP ranges, loop back address, multicast address, 0.0.0.0 address and broadcast address, telnet, FTP, all login services (Rlogin, etc), Sun RPC and NFS, Net BIOS, Xwindows, Netbus, Backorifice, and most ICMP. All of these will be implemented with logging. On the outbound interface egress ACLs will be applied to prevent unwanted traffic from exiting the network. Again, reserved IP ranges, loop back address, multicast address, 0.0.0.0 and broadcast address, telnet, FTP, all login services (Rlogin, etc), Sun RPC and NFS, Net BIOS, Xwindows, Netbus, Backorifice, and most ICMP will be blocked, with logging performed. The third connection will have ACLs applied to block everything inbound and outbound except TCP ports 10027-10028 and UDP ports 2233,10025-10027**,** which carry encrypted traffic to the VPN.

The VPN appliance connected at the border router will be an Intel NetStructure 3130 whose purpose is to provide VPNs to partners, secure remote access, and a testing gateway. The VPN will be used for testing in the same manner as described above on this network. ICMP will be allowed only from the VPN gateway past the packet filter and Sidewinder to the various other points on the network for further testing. Intel NetStructure 3110s will be installed at the partner sites to establish site-to-site tunnels with the corporate network. VPN client software will be installed on designated supplier computers and remote user computers.

The packet filtering router will block the same traffic as the border router and in addition will block other well known threats such as LDAP, SMTP, POP, IMAP, SSL, TCP and UDP ports below 20, NNTP, and other miscellaneous protocols on both the internet connection and the VPN connection. These will be blocked with logging enabled so the ACLs on the router can be confirmed and checked against the IDS log files.

The Sidewinder firewalls interface one will respond to exterior DNS queries, proxy HTTPS to and from the web server (interface three), proxy HTTP to and from interface five, and proxy ICMP to and from our diagnostic address. Interface two will only proxy SMTP communication between interfaces one, two, and five. Interface three will only proxy Web communication between interfaces one and three, and database communication between four and three. Interface four will only proxy database communication between interfaces four and three. Interface five will proxy connections to and from the corporate network; exterior DNS

queries, HTTP, HTTPS, and SMTP.

The third Sidewinder is located between the three parts of the corporate network and the working copies of the databases. This not only protects the databases from outside threats but also from inside threats. All connections to the database are made through the master proxy on interface one. Only users in the correct department can connect to their respective database. A copy of each master database is kept on separate servers, which are not connected to any network or to each other. The two different databases for the web servers are synchronized at intervals through tape drives, as are the different corporate databases. These physically different databases are not shown on the diagram.

ASSIGNMENT 2- SECURITY POLICY

Security Policies will be defined for GIAC Enterprises number One connection.

- Border Router Security Policy

The border routers will be setup to enforce the policy of least access. All traffic not expressly allowed will be prohibited, either with a specific rule or by default. The first order of business is to restrict access to the routers configuration interface. The routers will be configured to allow access only on the console port, which will be connected to a management laptop in the secured server room only when necessary.

Configuration of Cisco routers is done via the command line interface (CLI). A laptop or PC is connected to the console port by a console cable supplied by Cisco to an adaptor, which is connected to a COM port. The COM port settings would be:

1. Bits per Second: 9600
2. Data Bits: 8
3. Parity: none
4. Stop Bits: 1
5. Flow Control: none

A telnet session is opened with the router and the login screen appears. After being authenticated by a user name and password, the user is in the user executive mode where only basic router information is available.

Configuration commands apply to two different areas of the router s operation. There are global commands that apply to the router as a whole and there are interface commands that apply to a specific interface. To get to the privileged executive mode where more detail about the routers configuration is available, the user types “enable” <ENTER> at the router prompt. To perform configuration changes the user must type “configuration terminal” <ENTER> and he is at the global configuration prompt. This is where passwords, Access lists, and other items pertaining to the operation of the router can be entered and changed. For example: In order to turn off telnet access to the router the user would type “login VTY no password”, followed by CTRL^Z and “write memory” to write the configuration change to running memory, in order to save the configuration change to the start up configuration you must use the “copy running configuration to start up configuration” command. To configure changes to a specific interface, while in global configuration mode the user would specify which interface he wants to configure by typing “interface ‘whatever’”. For example “interface Ethernet 0”. This is where access lists are applied to interfaces by the command "access-group ‘access list number’ ‘direction of packet travel’”. For example: “access-group 101 in”. When an access list is written, there is an implicit deny all at the end of the list. If an empty access list is applied to an interface this has the effect of permitting all traffic. Packets entering or exiting the router are examined and compared to the access lists

applied to that interface in order from top to bottom, so order of the rules matter. There are two general ways to write access lists; allow only what traffic you want to come through, or allow all traffic and deny that traffic you don't want. These two methods can be combined to achieve certain objectives. For instance if you want to know how much of a certain type of traffic you are getting you could deny that traffic and log the instances of rejected packets.

Access list usage syntax is:
access-list *access-list-number* [dynamic *list-name* [timeout *value*]] {deny | permit} tcp *source source-wildcard* [operator port [port]] *destination destination-wildcard* [operator port [port]] [established] [precedence precedence] [tos tos] [log | log-input]

To apply an Access List to an interface:
ip access-group {*access-list-number* | *name*} {in | out}

In GIAC's case we will use a combination of the two methods and enumerate what will be accepted and what we explicitly want rejected.

The general protections that will be applied to the routers will be to prohibit:

1. Directed Broadcasts. These can be used to perform a denial of service attack such as Smurf. Interface Command: "no ip directed-broadcast". This is the default on Cisco IOS 12.0 and above.
2. IP Source Routing. This can be used along with a spoofed address to establish a session with an untrusted host. Global Command: "no ip source-route".
3. ICMP Redirects. There is no reason to dictate ICMP paths; we will disable this to prevent its use in an attack. Global Command: "no icmp redirects".
4. TCP and UDP "small services". These are echo, chargen, and discard and are rarely used for legitimate purposes but can be used by attackers to launch denial of service and other attacks. Global Command: "no service tcp-small-servers" and "no service udp-small-servers". These services are disabled by default in Cisco IOS 12.0 and above.
5. Finger. This service is not needed in a properly secured network, you should always know who is logged into the router. Global Command: "no service finger".
6. NTP. Network Time Protocol is not needed in our network and we will remove it to prevent any exploitation. Interface Command: "no ntp enable".
7. CDP. Cisco Discovery Protocol allows any system on a directly connected segment to learn that the router is a Cisco device and determine its IOS software version which could be used to implement other attacks. Global Command: "no cdp running". Interface Command: "no cdp enable".
8. HTTP management access. The router will be managed with the command line through the console port; we do not need HTTP management or the potential risk it brings. Global Command: "no ip http server".
9. BOOTP service access. This will be disabled to prevent denial of service attacks. Global Command: "no ip bootp server".

10. Proxy ARP. There is no need for this service on our network, this can open up our router to a spoofing attack and so will be disabled. Global Command: “no ip proxy arp”.

Following is a tutorial on how to write and apply an Ingress access list for the border router to be used at the GIAC Internet connection one.

Ingress access list

The Access list we will apply to the border gateway routers Serial 0 interface, connected to the Internet, will filter:

1. Spoofed Addresses and Private address ranges. These should not be coming in or going out of our network, coming in they must be “crafted” packets, built to gain information or worse. These will be protected against so no one can pretend to be a trusted source.
2. Net BIOS, Win NT and 2000 ports. These have no need to be allowed in or out, the only use is internal to the network. Prevents information gathering and denial of service attacks.
3. Loopback Address. Prevents a “Land Attack” to crash machines.
4. Multicast Address. By returning packets to a multicast address we could be participating in a denial of service attack.
5. Broadcast Address. This takes up bandwidth, can be used for denial of service, and to be a good neighbor we want it to be blocked on the egress filter.
6. Netbus and Back Orifice. Block remote control of machines.
7. Login services. Telnet transmits passwords in the clear, hackers use telnet with special termcap settings as a back door; we are not running a telnet server. We don’t need any remote login services’, letting these protocols through gives an attacker a “door” to bang on.
8. RPC. RPC is used to allow remote execution of programs, not something we want anyone to be able to do.
9. NFS. We are not running an NFS server; this is just another service an attacker can try to exploit.
10. X Windows. We don’t want any attackers trying to connect to an internal x-server. If successful he would gain root access.
11. ICMP. ICMP will be allowed only to our Router and Sidewinder Firewall. ICMP is an information-gathering tool.

We will allow HTTP, HTTPS, and ICMP through to the Sidewinder, and ICMP to the router, certain tcp and udp ports will be allowed to the VPN.

Note: All real ip addresses have been changed to the 192.168.x.x network.

Router External Address 192.168.90.3

Test Address is 192.168.90.1

VPN appliance external 192.168.90.2

ISP Address is 192.168.90.4

Firewall External Address 1 is 192.168.92.5

Firewall External Address 2 is 192.168.92.6

Firewall Internal Address is 192.168.92.7

Sidewinder External Address is 192.168.92.8

Router Internal Address 192.168.92.1

VPN appliance internal 192.168.92.2

Web Server 192.168.92.50

DNS Server 192.168.92.51

So to configure the router, after logging in, the router prompt will look like this:
RouterName>
Type in “enable” <ENTER>, and the prompt will change to:
RouterName#
Type in “config t” <ENTER>, and the prompt will change to:
RouterName(config)#
The router is now in global configuration mode and global configuration commands can be entered.
Starting with the general commands to be entered, press <ENTER> after each command:

no ip directed-broadcast
no ip source-route
no icmp redirects
no service tcp-small-servers
no service finger
no cdp running
no ip http server

The Access List to be entered will look like this:

Permit Statements

1. Web Server Access
access-list 101 permit tcp 0.0.0.0 0.0.0.0 host 192.168.92.50 80
access-list 101 permit tcp 0.0.0.0 0.0.0.0 host 192.168.92.50 443
2. ICMP to Router and Firewall
access-list 101 permit icmp 0.0.0.0 0.0.0.0 host 192.168.90.3
access-list 101 permit icmp 0.0.0.0 0.0.0.0 host 192.168.92.5
3. Encrypted traffic to VPN from test address

access-list 101 permit udp host 192.168.90.1 host 192.168.90.2 2233
access-list 101 permit udp host 192.168.90.1 host 192.168.90.2 range 10025 10027
access-list 101 permit tcp host 192.168.90.1 host 192.168.90.2 range 10027 10028

4. DNS traffic to DNS server
access-list 101 permit tcp any host 192.168.92.51 eq domain
access-list 101 permit udp any host 192.168.92.51 eq domain

Deny Statements

1. Spoofed Addresses
access-list 101 deny ip 192.168.92.2 0.0.0.0 any log
access-list 101 deny ip 168.192.92.0 0.0.0.255 any log
access-list 101 deny ip host 0.0.0.0 any log
access-list 101 deny ip 10.0.0.0 0.255.255.255 any log
access-list 101 deny ip 172.16.0.0 0.15.255.255 any log
access-list 101 deny ip 192.168.0.0 0.0.255.255 any log

2. Net BIOS, Win NT, Win 2K
access-list 101 deny tcp any any 135 log
access-list 101 deny udp any any 135 log
access-list 101 deny udp any any range 137 138 log
access-list 101 deny tcp any any eq 139 log
access-list 101 deny tcp any any eq 445 log
access-list 101 deny udp any any eq 445 log

3. Loopback Address
access-list 101 deny ip 127.0.0.0 0.255.255.255 any log

4. Multicast Address
access-list 101 deny ip 224.0.0.0 7.255.255.255 any log

5. Broadcast Address
access-list 101 deny ip 255.0.0.0 0.255.255.255 any log

6. Netbus and Back Orifice
access-list 101 deny udp any any eq 31337 log
access-list 101 deny tcp any any range 12345 12346 log

7. Login Services
access-list 101 deny tcp any any range ftp telnet log
access-list 101 deny any any range exec lpd log

8. RPC
access-list 101 deny udp any any eq sunrpc log

access-list 101 deny tcp any any eq sunrpc log

9. NFS
access-list 101 deny udp any any eq 2049 log
access-list 101 deny tcp any any eq 2049 log
access-list 101 deny udp any any eq 4045 log
access-list 101 deny tcp any any eq 4045 log

10. X Windows
access-list 101 deny tcp any any range 6000 6255 log

11. ICMP
access-list 101 deny icmp any any

To disable NTP on the Serial 0 interface, at the global configuration prompt:
RouterName#
Type: “interface serial 0” <ENTER>
And the prompt will change to:
RouterName(config-if)#
Now serial interface configuration commands can be entered.
Type: “no ntp enable” <ENTER>
Access lists are applied at the interface configuration prompt.
To apply the access list to this interface:
Type: “ip access-group 101 in” <ENTER>
Type: “^ z” <ENTER>
Prompt changes to: RouterName#
Type: “write memory” <ENTER>
Type “copy run config start config” <ENTER>
Type “reboot” <ENTER>

Now the configuration changes have been saved to the start up configuration file, and when the
router reboots it will have the access list applied.

The Access list we will apply to the border gateway routers eth01 interface, connected to the packet filter firewall eth01interface, will look like this:

Permit Statements: Allow established TCP connections and UDP for DNS
access-list 110 permit tcp host 192.168.92.5 any established
access-list 110 permit udp host 192.168.92.5 any

access-list 101 permit icmp host 192.168.92.5 0.0.0.0 0.0.0.0

Deny Statements

1. Spoofed Addresses
access-list 101 deny ip 192.168.92.2 0.0.0.0 any log
access-list 101 deny ip 168.192.92.0 0.0.0.255 any log
access-list 101 deny ip host 0.0.0.0 any log
access-list 101 deny ip 10.0.0.0 0.255.255.255 any log
access-list 101 deny ip 172.16.0.0 0.15.255.255 any log
access-list 101 deny ip 192.168.0.0 0.0.255.255 any log

2. Net BIOS, Win NT, Win 2K
access-list 101 deny tcp any any 135 log
access-list 101 deny udp any any 135 log
access-list 101 deny udp any any range 137 138 log
access-list 101 deny tcp any any eq 139 log
access-list 101 deny tcp any any eq 445 log
access-list 101 deny udp any any eq 445 log

3. Loopback Address
access-list 101 deny ip 127.0.0.0 0.255.255.255 any log

4. Multicast Address
access-list 101 deny ip 224.0.0.0 7.255.255.255 any log

5. Broadcast Address
access-list 101 deny ip 255.0.0.0 0.255.255.255 any log

6. Netbus and Back Orifice
access-list 101 deny udp any any eq 31337 log
access-list 101 deny tcp any any range 12345 12346 log

7. Login Services
access-list 101 deny tcp any any range ftp telnet log
access-list 101 deny any any range exec lpd log

8. RPC
access-list 101 deny udp any any eq sunrpc log
access-list 101 deny tcp any any eq sunrpc log

9. NFS
access-list 101 deny udp any any eq 2049 log
access-list 101 deny tcp any any eq 2049 log
access-list 101 deny udp any any eq 4045 log
access-list 101 deny tcp any any eq 4045 log

10. X Windows
access-list 101 deny tcp any any range 6000 6255 log

11. ICMP
access-list 101 deny icmp any any

The Access list we will apply to the border gateway routers eth02 interface, connected to the packet filter firewall eth02 interface, will look like this:

Permit Statements: Allow only encrypted data ports.
access-list 101 permit udp host 192.168.90.2 host 192.168.92.2 2233
access-list 101 permit udp host 192.168.90.2 host 192.168.92.2 range 10025 10027
access-list 101 permit tcp host 192.168.90.2 host 192.168.92.2 range 10027 10028

Deny Statements: Everything else is denied and logged
access-list 101 tcp deny any any log
access-list 101 udp deny any any log

Egress Access List

The Access list applied to the border gateway routers Serial 0 interface, outbound, will filter:

1. Spoofed Addresses and Private address ranges. These should not be going out of our network.
2. Net BIOS, Win NT and 2000 ports. These should not be allowed out, the only use is internal to the network. Prevents information gathering and denial of service attacks.
3. Loopback Address. Prevents a "Land Attack" to crash machines.
4. Multicast Address. By returning packets to a multicast address we could be participating in a denial of service attack.
5. Broadcast Address. This is just "noise" to be filtered out to be a "good neighbor".

We will allow HTTP, HTTPS, some ICMP, some UDP, and some DNS.

Permit Statements

1. Web Server Access
access-list 102 permit tcp host 192.168.92.50 eq HTTP 0.0.0.0 0.0.0.0
access-list 102 permit tcp host 192.168.92.50 eq HTTPS 0.0.0.0 0.0.0.0
2. ICMP to Router and Firewall
access-list 102 permit icmp host 192.168.90.3 0.0.0.0 0.0.0.0
access-list 102 permit icmp host 192.168.92.5 0.0.0.0 0.0.0.0
3. Encrypted traffic to VPN from test address
access-list 102 permit udp host 192.168.90.2 2233 host 192.168.90.1
access-list 102 permit udp host 192.168.90.2 range 10025 10027 host 192.168.90.1
access-list 102 permit tcp host 192.168.90.2 range 10027 10028 host

192.168.90.1
4. DNS traffic to DNS server
access-list 102 permit tcp host 192.168.92.51 eq domain any
access-list 102 permit udp host 192.168.92.51 eq domain any

Deny Statements

1. Spoofed Addresses
access-list 101 deny ip 192.168.92.2 0.0.0.0 any log
access-list 101 deny ip 168.192.92.0 0.0.0.255 any log
access-list 101 deny ip host 0.0.0.0 any log
access-list 101 deny ip 10.0.0.0 0.255.255.255 any log
access-list 101 deny ip 172.16.0.0 0.15.255.255 any log
access-list 101 deny ip 192.168.0.0 0.0.255.255 any log

2. Net BIOS, Win NT, Win 2K
access-list 101 deny tcp any any 135 log
access-list 101 deny udp any any 135 log
access-list 101 deny udp any any range 137 138 log
access-list 101 deny tcp any any eq 139 log
access-list 101 deny tcp any any eq 445 log
access-list 101 deny udp any any eq 445 log

3. Loopback Address
access-list 101 deny ip 127.0.0.0 0.255.255.255 any log

4. Multicast Address
access-list 101 deny ip 224.0.0.0 7.255.255.255 any log

5. Broadcast Address
access-list 101 deny ip 255.0.0.0 0.255.255.255 any log

- Packet Filter Security Policy

Ipchains is implemented as a kernel level service in Linux, so the kernel must be compiled with ipchains enabled. To use ipchains you write rules with the command: /sbin/ipchains “rule set” and the ipchains command applies those rules to the kernels packet filtering section. To make the changes permanent run the command (as root):”ipchains-save > (file you want to save as, such as: /etc/ipchains.rules)”. Then run a script on boot up to run this script.
The kernel reads each packet’s header information and compares it to the rule set to determine what to do with the packet. Ipchains has three built in chains, input, output, and forward, which you can’t delete. You may create chains of your own, if desired. For the GIAC packet filtering firewall we will be using a sub set of the ipchains command syntax, the basic syntax is as

follows:

ipchains *command rule-specification options*
The commands used for the GIAC firewall will be:

ipchains –F, to flush the rules from all chains.

ipchains –X, to delete the empty chains.

Ipchains –P, to establish the policies for the new chains.

Ipchains –A, to add a rule to a chain.

The policies for the three default chains will be set with the commands: ipchains –P input DENY; this denies all packets entering any interface by default, ipchains –P forward ACCEPT; this forwards all packets from one interface to another by default, and ipchains –P output DENY; this denies all packets exiting any interface by default.

These are the options for an ipchains rule:

-d destn. IP address and port num.

-f packet fragment
-p protocol type (tcp, udp, icmp)
-s source IP address and port num.
-i interface name (lo, ppp0, eth0)
-j target (ACCEPT, DENY, REJECT, MASQ)
-l log this packet
-y syn packet (tcp only)
--dport destn. port
--sport source port
--icmp-type
! logical not, anything but this address

For example, to add a rule to a chain the following command is used: ipchains -A input -i eth0 -s 192.168.1.0/24 -j ACCEPT - l

This command adds a rule to the input chain on the eth0 interface that will accept packets from the 192.168.1.0 network and log each matching packet.

The packet filter firewall rule set that will be used at GIAC will filter the following:

We will accept HTTP, HTTPS and DNS to the Sidewinder. We will accept ICMP from the ISP address, our test address and the VPN interface. All other traffic will be denied, sometimes explicitly with logging, and the rest with an explicit deny with no logging. The traffic we will deny will the same traffic that was denied at the border router, for explanations, see border router section. We will log all other denied packets to audit what the firewall is stopping, so we can adjust the rules if necessary.

6. Spoofed Addresses and Private address ranges.
7. Net BIOS, Win NT and 2000 ports.
8. Loopback Address.
9. Multicast Address.
10. Broadcast Address.
11. Netbus and Back Orifice.
12. Login services.
13. RPC.
14. NFS.

10. X Windows.
11. ICMP

This is the shell script we will run at boot time, less the explanatory remarks , which are underlined.

```
#!/bin/sh

/sbin/ipchains -F
/sbin/ipchains -X


Establish Policies for the three default chains
/sbin/ipchains -P input DENY
/sbin/ipchains -P output DENY
/sbin/ipchains -P forward ACCEPT

Enable IP Forwarding
 echo 1 > /proc/sys/net/ipv4/ip_forward

Enable TCP SYN Cookie Protection
echo 1 > /proc/sys/net/ipv4/tcp_syncookies

Always defrag packets
echo 1 > /proc/sys/net/ipv4/ip_always_defrag

Ignore echo broadcasts
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

Ignore bad error messages
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses

Protect from IP spoofing with Source Address Verification
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
 echo 1 > $f
```

done

Don't accept ICMP Redirects

```
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
   echo 0 > $f
done
```

Don't accept Source Routed Packets

```
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
   echo 0 > $f
done
```

Spoofed Packets, Source Routed Packets, Redirect Packets are logged

```
for f in /proc/sys/net/ipv4/conf/*/log_martians; do
   echo 1 > $f
done
```

Accept Internal Loopback

```
/sbin/ipchains -A input -i lo -j ACCEPT
/sbin/ipchains -A output -i lo -j ACCEPT
```

Deny TCP connections initiating from the inside

```
/sbin/ipchains -A input -i 192.168.92.6 -p tcp -s 192.168.92.0 255.255.255.240 -y -
j Deny -l
```

Deny packets from private addresses on ext NIC

```
/sbin/ipchains -A input -i eth0 -s 10.0.0.0/8 -j DENY -l
/sbin/ipchains -A input -i eth0 -d 10.0.0.0/8 -j DENY -l
/sbin/ipchains -A output -i eth0 -s 10.0.0.0/8 -j DENY -l
/sbin/ipchains -A output -i eth0 -d 10.0.0.0/8 -j DENY -l
/sbin/ipchains -A input -i eth0 -s 172.16.0.0/12 -j DENY -l
/sbin/ipchains -A input -i eth0 -d 172.16.0.0/12 -j DENY -l
/sbin/ipchains -A output -i eth0 -s 172.16.0.0/12 -j DENY -l
/sbin/ipchains -A output -i eth0 -d 172.16.0.0/12 -j DENY -l
/sbin/ipchains -A input -i eth0 -s 192.168.0.0/16 -j DENY -l
/sbin/ipchains -A input -i eth0 -d 192.168.0.0/16 -j DENY -l
/sbin/ipchains -A output -i eth0 -s 192.168.0.0/16 -j DENY -l
/sbin/ipchains -A output -i eth0 -d 192.168.0.0/16 -j DENY -l
```

Deny External Loopback

```
/sbin/ipchains -A input -i eth0 -s 127.0.0.0/8 -j DENY -l
```

/sbin/ipchains -A output -i eth0 -s 127.0.0.0/8 -j DENY -l

Deny Broadcasts on External

/sbin/ipchains -A input -i eth0 -s 255.255.255.255 -j DENY -l
/sbin/ipchains -A input -i eth0 -d 0.0.0.0 -j DENY -l

Deny reserved addresses

/sbin/ipchains -A input -i eth0 -s 1.0.0.0/8 -j DENY -l
/sbin/ipchains -A input -i eth0 -s 2.0.0.0/8 -j DENY -l
/sbin/ipchains -A input -i eth0 -s 5.0.0.0/8 -j DENY -l
/sbin/ipchains -A input -i eth0 -s 7.0.0.0/8 -j DENY -l
/sbin/ipchains -A input -i eth0 -s 23.0.0.0/8 -j DENY -l
/sbin/ipchains -A input -i eth0 -s 27.0.0.0/8 -j DENY -l
/sbin/ipchains -A input -i eth0 -s 31.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 37.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 39.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 41.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 42.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 58.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 60.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 65.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 66.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 67.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 68.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 69.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 70.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 71.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 72.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 73.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 74.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 75.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 76.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 77.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 78.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 79.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 80.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 96.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 112.0.0.0/8 -j DENY - l

/sbin/ipchains -A input -i eth0 -s 113.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 114.0.0.0/8 -j DENY – l
/sbin/ipchains -A input -i eth0 -s 115.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 116.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 117.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 118.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 119.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 120.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 121.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 122.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 123.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 124.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 125.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 126.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 217.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 218.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 219.0.0.0/8 -j DENY - l
/sbin/ipchains -A input -i eth0 -s 220.0.0.0/8 -j DENY - l

Accept ICMP from ISP and Test address

/sbin/ipchains -A input -i eth0 -p icmp --icmp-type echo-request -d 192.168.92.7 -j ACCEPT
/sbin/ipchains -A output -i eth0 -p icmp --icmp-type echo-reply -d 192.168.90.4 -j ACCEPT
/sbin/ipchains -A input -i eth2 -p icmp --icmp-type echo-request -d 192.168.92.6 -j ACCEPT
/sbin/ipchains -A output -i eth2 -p icmp --icmp-type echo-reply -d 192.168.92.2 -j ACCEPT
/sbin/ipchains -A input -i eth2 -p icmp --icmp-type echo-request -d 192.168.92.7 -j ACCEPT
/sbin/ipchains -A input -i eth0 -p icmp --icmp-type echo-request -d 192.168.92.7 -j ACCEPT
/sbin/ipchains -A output -i eth2 -p icmp --icmp-type echo-reply -s 192.168.92.7 -j ACCEPT
/sbin/ipchains -A input -i eth2 -p icmp --icmp-type echo-request -d 192.168.92.6 -j ACCEPT
/sbin/ipchains -A input -i eth1 -p icmp --icmp-type echo-request -s 192.168.92.6 -j ACCEPT
/sbin/ipchains -A output -i eth2 -p icmp --icmp-type echo-reply -s 192.168.92.6 -j ACCEPT
/sbin/ipchains -A input -i eth2 -p icmp --icmp-type echo-request -d 192.168.92.50 -j ACCEPT

/sbin/ipchains -A input -i eth1 -p icmp --icmp-type echo-request -d 192.168.92.50 -j ACCEPT
/sbin/ipchains -A output -i eth1 -p icmp --icmp-type echo-request -d 192.168.92.50 -j ACCEPT
/sbin/ipchains -A input -i eth1 -p icmp --icmp-type echo-reply -s 192.168.92.50 -j ACCEPT
/sbin/ipchains -A output -i eth2 -p icmp --icmp-type echo-reply -d 192.168.92.2 -j ACCEPT

Accept HTTP to Sidewinder

/sbin/ipchains -A input -i eth0 -s ! 192.168.92.8 1024:65535 -d 0.0.0.0/0.0.0.0 80 -p tcp -j ACCEPT

/sbin/ipchains -A output -i eth0 -s 0.0.0.0/0.0.0.0 80 -d ! 192.168.92.8 1024:65535 ! -y -p tcp -j ACCEPT

/sbin/ipchains -A input -i eth1 -s 0.0.0.0/0.0.0.0 80 -d ! 192.168.92.81024:65535 ! -y -p tcp -j ACCEPT

/sbin/ipchains -A output -i eth1 -s ! 192.168.92.8 1024:65535 -d 0.0.0.0/0.0.0.0 80 -p tcp -j ACCEPT

/sbin/ipchains -A forward -i eth1 -s ! 192.168.92.8 1024:65535 -d 0.0.0.0/0.0.0.0 80 -p tcp -j ACCEPT

/sbin/ipchains -A forward -i eth0 -s 0.0.0.0/0.0.0.0 80 -d ! 192.168.92.8 1024:65535 ! -y -p tcp -j ACCEPT

Accept DNS to Sidewinder

/sbin/ipchains -A input -i eth0 -s ! 192.168.92.50 1024:65535 -d 0.0.0.0/0.0.0.0 53 -p udp -j ACCEPT

/sbin/ipchains -A output -i eth0 -s 0.0.0.0/0.0.0.0 53 -d ! 192.168.92.50 1024:65535 -p udp -j ACCEPT

/sbin/ipchains -A input -i eth1 -s 0.0.0.0/0.0.0.0 53 -d ! 192.168.92.50 1024:65535 -p udp -j ACCEPT

/sbin/ipchains -A output -i eth1 -s ! 192.168.92.50 1024:65535 -d 0.0.0.0/0.0.0.0 53 -p udp -j ACCEPT

/sbin/ipchains -A forward -i eth1 -s ! 192.168.92.50 1024:65535 -d 0.0.0.0/0.0.0.0 53 -p udp -j ACCEPT

/sbin/ipchains -A forward -i eth0 -s 0.0.0.0/0.0.0.0 53 -d ! 192.168.92.50 1024:65535 -p udp -j ACCEPT

Accept HTTPS to Sidewinder

/sbin/ipchains -A input -i eth0 -s ! 192.168.92.50 1024:65535 -d 0.0.0.0/0.0.0.0 443 -p tcp -j ACCEPT

/sbin/ipchains -A output -i eth0 -s 0.0.0.0/0.0.0.0 443 -d ! 192.168.92.50 1024:65535 ! -y -p tcp -j ACCEPT

/sbin/ipchains -A input -i eth1 -s 0.0.0.0/0.0.0.0443 -d ! 192.168.92.50 1024:65535 -p tcp -j ACCEPT

/sbin/ipchains -A output -i eth1 -s ! 192.168.92.50 1024:65535 -d 0.0.0.0/0.0.0.0 443 -p tcp -j ACCEPT

/sbin/ipchains -A forward -i eth1 -s ! 192.168.92.50 1024:65535 -d 0.0.0.0/0.0.0.0 443 -p tcp -j ACCEPT

/sbin/ipchains -A forward -i eth0 -s 0.0.0.0/0.0.0.0 443 -d ! 192.168.92.50 1024:65535 ! -y -p tcp -j ACCEPT

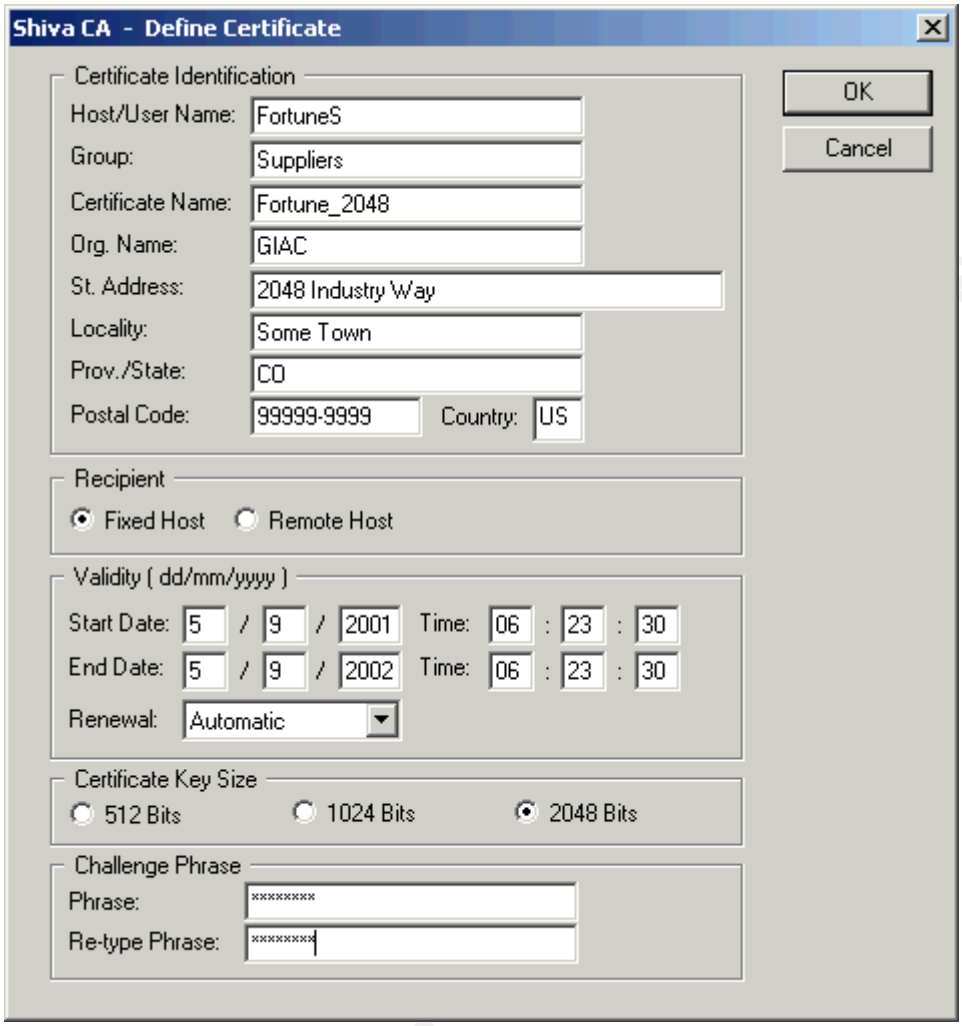
Log some denied packets

ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp -j DENY -l

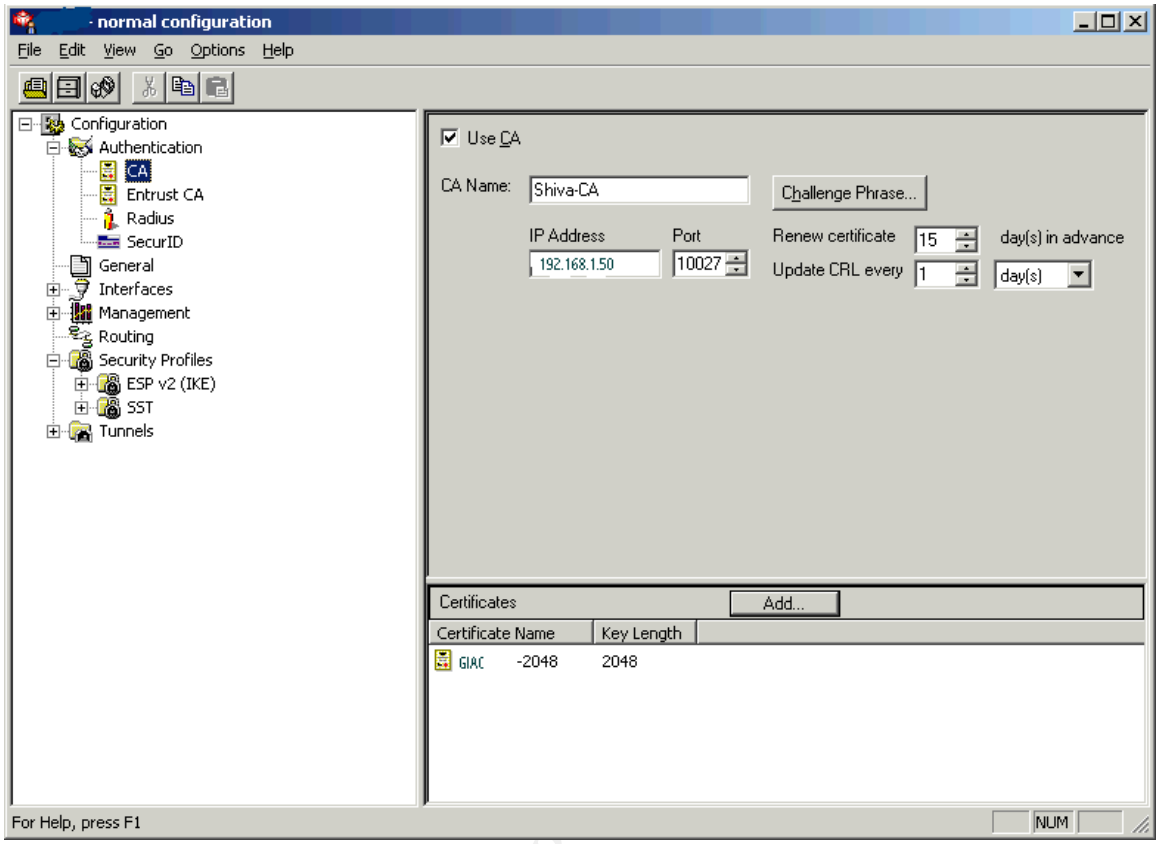
- VPN Security Policy

The VPN appliances used at GIAC will be Intel NetStructure 3110's and 3130's. The differences between the two being the bandwidth they can handle and the number of simultaneous tunnels they are able to support.

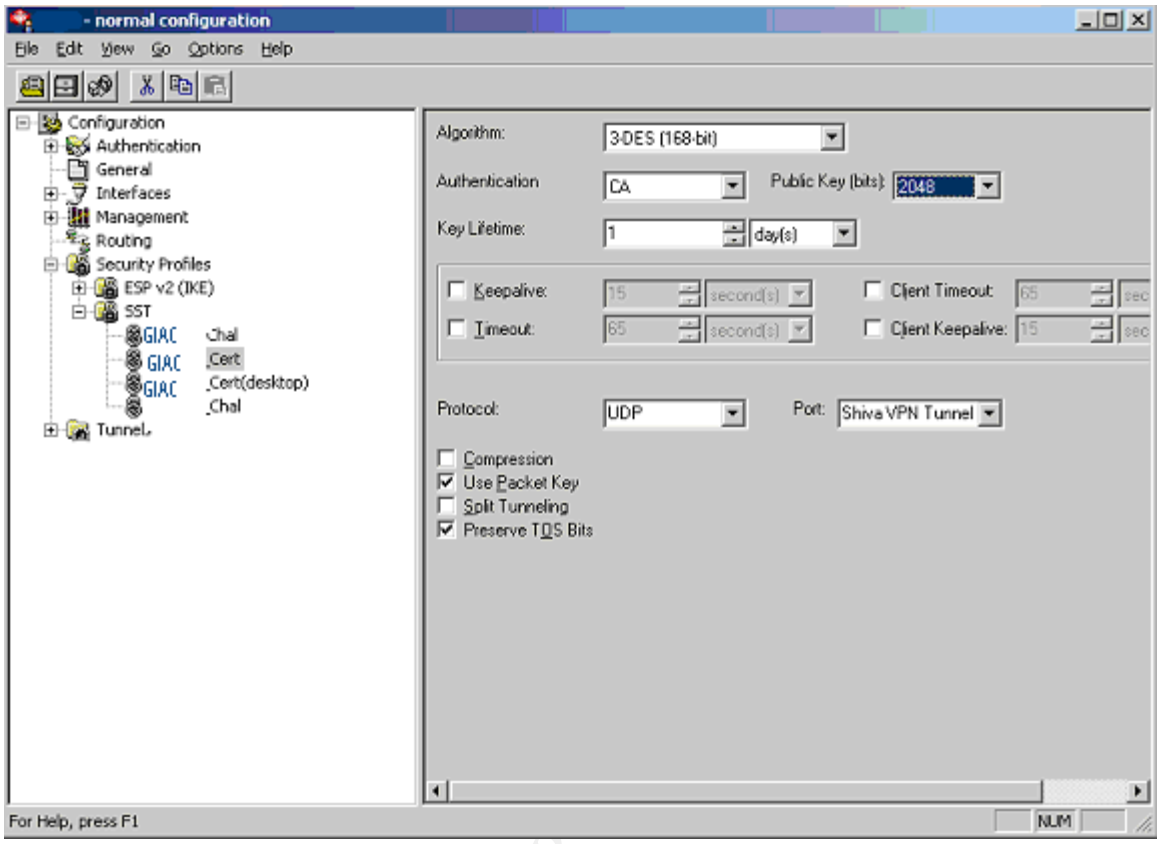
GIAC will supply a 3110 to those partners and suppliers as is deemed necessary by security requirements. GIAC Enterprises will use an Intel NetStructure 3130, in a hub and spoke arrangement with the deployed 3110's. This will provide for any growth requirements. The tunnels established will be SST tunnels (Shiva Smart Tunneling). SST tunnels encrypt each packet with a different packet key and then the packet key is encrypted with a session key. Validation of each Intel NetStructure will be accomplished with a Shiva Certificate Authority installed at GIAC headquarters. SST tunnels are initiated by the VPN appliances interfaces "seeing" each other over the Internet, so unless the device is configured with a partners device IP address, no connection is allowed. To setup an SST tunnel to use Certificates, the Certificate must first be created in the following manner on the Certificate server:



To establish a site-to-site tunnel between two VPN appliances the configuration on each NetStructure would be performed as follows:
First the Authentication method for the device needs to be set. This is accomplished by setting the Certificate Authority name, Challenge Phrase (password), IP address, Port and Certificate name and key length.



Second, the Security Association of the tunnel would be established. This is done by specifying the Algorithm, Authentication method, key length, key lifetime, connection keep alive and timeout, protocol to use (UDP or 99), which port to use (default is 2233). The other four options are whether to use compression, a packet key (encrypt each packet individually), split tunneling (allow unencrypted traffic to network), and whether to keep the Type of Service bits in the new IP header.



Client tunnels are established in much the same manner, except the certificate is fulfilled to a floppy disk, which is mailed, or the certificate is emailed with the client software to the user and the configuration done.

ASSIGNMENT 3

- Auditing the Security Architecture

A perimeter security audit would not focus strictly on the network portion of the architecture, but would include other aspects of security as well, such as:
Company Security Policy
Physical security of site
Physical security of network devices
Network device access
Password Policy
Security education of Users
All of these areas would be examined in depth before any work on the network would be undertaken.

Primary Firewall

The approach I would take to audit the primary firewall would be to look at the firewall from two different angles, inside and outside. The inside view would consist of looking at the packet filtering rules, and determining whether they allow and deny the proper traffic. After that determination the log files of the firewall would be examined to see what it had been denying.

The packet filtering rules have already been examined. The kernel log files list the packets that have been denied during the various tests that have been performed on the firewall, a sample of which follows with explanations:
Aug 11 15:47:44 localhost kernel: Packet log: input DENY eth0 PROTO=6 192.168.1.213:36937 192.168.92.7:252 L=40 S=0x00 I=5447 F=0x0000 T=51 SYN (#26)

The first section of the log, Aug 11 15:22:07 localhost kernel: Packet log:, consists of the date stamp, and the origin of the log. The second section, input DENY eth0, tells us that a packet was denied coming into interface ethernet 0. The next section, PROTO=6 192.168.1.213:36448 192.168.1.15:6145, tells us the protocol is TCP (1=ICMP, 6=TCP, 17=UDP), the originating ip address, the originating port, the destination ip address, and the destination port. The last section, L=40 S=0x00 I=14124 F=0x0000 T=53 SYN (#26), consists of the ip header fields in the packet, L= length of Datagram in octets, S= type of service, I= identifying value for reassembly of fragmented packets, F= various control flags, T= time to live, and the SYN indicates that this is a synchronization packet.

This was a packet generated by an Nmap TCP SYN scan done to the packet filters Internet address.

The outside view would be obtained by monitoring the traffic on all interfaces of the firewall to determine what traffic is actually there. Then a scanner, such as Nmap, would be employed to determine what it saw as the open ports of the firewall from all interfaces. Finally, traffic that is to be allowed or denied would be directed at the firewall interfaces with monitoring being done by TCP Dump to see what was actually allowed or denied.

There is no real traffic to be monitored since this is a dummy network, a sample TCPdump of an Nmap TCP SYN generated scan follows with explanations:

13:21:20.507643 eth0 < 192.168.90.5.51443 > 192.168.92.7.366: S 1949304813:1949304813(0) win 3072

The first section of the log, 13:21:20.507643 eth0 ,consists of the date stamp, and the interface the packet was received on, ethernet 0. The next section, < 192.168.90.5.51443 > 192.168.92.7.366: S, tells us the originating ip address, the destination ip address, and the type of packet, which is a SYN packet. The last section, 1949304813:1949304813(0) win 3072 , tells us the beginning and ending sequence numbers and the window size.

Nmap scans can be initialized from the command line on Windows NT, and from a GUI or command line on Unix. The following examples are from a Windows NT machine.

Nmap scans can take many forms and require tailoring to the situation. To audit the packet filter firewall at GIAC, four different types of scans were used: a TCP SYN scan in which only SYN packets are sent, an "Xmas Tree" scan which uses packets with the FIN, URG, and PUSH flags turned on, a UDP scan, and a ACK scan. These scans should give a good indication of the packet filters vulnerabilities.

All of the scans were performed with a TCPdump session logging packets at the packet filter, and kernel log files were examined for the same time frame. The Nmap SYN scan session looked as follows:

Nmap command screen dump

Nmap (V. nmap) scan initiated 2.53 as: nmapnt -sS -vv -p 1-65535 -oN C:\nmap1.txt 192.168.92.7
All 65535 scanned ports on (192.168.92.7) are: filtered
Nmap run completed at Thu Aug 23 13:31:11 2001 -- 1 IP address (1 host up) scanned in 280 seconds

The scan was initiated with the –sS option, which indicate a SYN scan. Nmap sends out SYN packets to all the ports and waits for the response of a SYN/ACK packet to determine that the port is open. A RST response indicates that the port is closed. The –vv gives verbose messages, the –p tells Nmap to scan ports 1 through 65535, and the –oN sends the Nmap output to the file of your choice, in this case C:\nmap1.txt. The results of this scan indicate that Nmap determined that all 1523 ports that were scanned are in the filtered state which means that something (firewall, filter, other network obstacle) is covering those ports and not allowing Nmap to determine whether the port is open.

The TCPdump traffic generated by this scan is lengthy, so only selected packets will be examined.

13:21:40.937118 eth0 < 192.168.90.5.51444 > 192.168.92.7.880: S 1223753908:1223753908(0) win 3072
13:21:40.937169 eth0 < 192.168.90.5.51444 > 192.168.92.7.6141: S 1223753908:1223753908(0) win 3072
13:21:40.937293 eth0 < 192.168.90.5.51444 > 192.168.92.7.930: S

1223753908:1223753908(0) win 3072
These three samples show that Nmap is sending out SYN packets from a high port to different destination ports on the packet filter.

Matching kernel Log entries from SYN scan:

Aug 24 17:12:35 localhost kernel: Packet log: input DENY eth0 PROTO=6 192.168.90.5: 51444 192.168.92.7:880 L=40 S=0x00 I=9225 F=0x0000 T=47 (#26)
Aug 24 17:12:35 localhost kernel: Packet log: input DENY eth0 PROTO=6 192.168.90.5:63370 192.168.92.7:48 L=40 S=0x00 I=61523 F=0x0000 T=47 (#26)

Results: All Ports are filtered, not open for use.

The Nmap UDP scan session looked as follows:
Nmap command screen dump

Nmap (V. nmap) scan initiated 2.53 as: nmapnt -sU -vv -p 1-65535 -oN C:\sUscan.txt 192.168.92.7
All 65535 scanned ports on (192.168.92.7) are: filtered
Nmap run completed at Sat Aug 25 18:21:42 2001 -- 1 IP address (1 host up) scanned in 4259 seconds

Selected TcpDump Output
17:18:37.297382 eth0 < 192.168.90.5.51743 > 192.168.92.7.3131: udp 0
17:18:37.300903 eth0 < 192.168.90.5.51743 > 192.168.92.7.22502: udp 0
17:18:37.304347 eth0 < 192.168.90.5.51743 > 192.168.92.7.16050: udp 0

Matching Kernel Log entries:
Aug 25 17:18:41 localhost kernel: Packet log: input DENY eth0 PROTO=17 192.168.90.5:51743 192.168.92.7:3131 L=28 S=0x00 I=13879 F=0x0000 T=39 (#27)
Aug 25 17:18:41 localhost kernel: Packet log: input DENY eth0 PROTO=17 192.168.90.5:51743 192.168.92.7:22502 L=28 S=0x00 I=36160 F=0x0000 T=39 (#27)
Aug 25 17:18:41 localhost kernel: Packet log: input DENY eth0 PROTO=17 192.168.90.5:51743 192.168.92.7:16050 L=28 S=0x00 I=20863 F=0x0000 T=39 (#27)

Results: All Ports are filtered, not open for use.

The Nmap ACK scan session looked as follows:
Nmap command screen dump
Nmap (V. nmap) scan initiated 2.53 as: nmapnt -sA -vv -p 1-65535 -oN

C:\sAscan.txt 192.168.92.7
All 65535 scanned ports on (192.168.92.7) are: filtered
Nmap run completed at Sat Aug 25 20:58:17 2001 -- 1 IP address (1 host up) scanned in 2166 seconds

Selected TcpDump Output:
20:21:42.734097 eth0 < 192.168.90.5.35450 > 192.168.92.7.13911: . 2087393126:2087393126(0) ack 1727088108 win 4096
20:21:42.734414 eth0 < 192.168.90.5.35450 > 192.168.92.7.39867: . 2087393126:2087393126(0) ack 1727088108 win 4096
20:21:42.734734 eth0 < 192.168.90.5.35450 > 192.168.92.7.5343: . 2087393126:2087393126(0) ack 1727088108 win 4096

Matching Kernel Log entries:
Aug 25 20:21:42 localhost kernel: Packet log: input DENY eth0 PROTO=6 192.168.90.5:35450 192.168.92.7:13911 L=40 S=0x00 I=591 F=0x0000 T=47 (#26)
Aug 25 20:21:42 localhost kernel: Packet log: input DENY eth0 PROTO=6 192.168.90.5:35450 192.168.92.7:39867 L=40 S=0x00 I=12320 F=0x0000 T=47 (#26)
Aug 25 20:21:42 localhost kernel: Packet log: input DENY eth0 PROTO=6 192.168.90.5:35450 192.168.92.7:5343 L=40 S=0x00 I=5233 F=0x0000 T=47 (#26)

Results: All Ports are filtered, not open for use.

The Nmap X-mas Tree scan session looked as follows:
Nmap command screen dump

Nmap (V. nmap) scan initiated 2.53 as: nmapnt -sX -vv -p 1-65535 -oN C:\SXscan.txt 192.168.92.7
All 65535 scanned ports on (192.168.92.7) are: filtered
Nmap run completed at Mon Sep 03 15:09:03 2001 -- 1 IP address (1 host up) scanned in 5110 seconds

Selected TcpDump Output:
13:43:17.081881 eth0 < 192.168.90.5.59955 > 192.168.92.7.13887: FP 0:0(0) win 1024 urg 0
13:43:17.082226 eth0 < 192.168.90.5.59955 > 192.168.92.7.6277: FP 0:0(0) win 1024 urg 0
13:43:17.082509 eth0 < 192.168.90.5.59955 > 192.168.92.7.16291: FP 0:0(0) win 1024 urg 0

Matching kernel Log entries:

Sep 3 13:43:17 localhost kernel: Packet log: input DENY eth0 PROTO=6 192.168.90.5:59955 192.168.92.7:13887 L=40 S=0x00 I=61976 F=0x0000 T=48 (#26)
Sep 3 13:43:17 localhost kernel: Packet log: input DENY eth0 PROTO=6 192.168.90.5:59955 192.168.92.7:6277 L=40 S=0x00 I=29782 F=0x0000 T=48 (#26)
Sep 3 13:43:17 localhost kernel: Packet log: input DENY eth0 PROTO=6 192.168.90.5:59955 192.168.92.7:16291 L=40 S=0x00 I=65027 F=0x0000 T=48 (#26)

Audit Results: All Ports are filtered, not open for use.

TCP dump results from HTTP and DNS traffic directed at web server and DNS server behind packet filter:

Constructing a web server on Microsoft IIS and connecting to it through the packet filter tested HTTP and HTTPS traffic.
Selected TcpDump output:
17:30:36.895255 eth1 > 192.168.90.5.1034 > 192.168.92.50.www: S 58905:58905(0) win 8192 <mss 1460> (DF)
17:30:36.895484 eth1 < 192.168.92.50.www > 192.168.90.5.1034: S 56779:56779(0) ack 58906 win 8760 <mss 1460> (DF)

These two lines show the SYN request and the ACK reply.

The packet filter firewall rules were changed to log HTTP traffic allowed to see the connections going through also:
Sep 4 17:29:26 localhost kernel: Packet log: input ACCEPT eth0 PROTO=6 192.168.90.5:1031 192.168.92.50:80 L=44 S=0x00 I=38144 F=0x4000 T=128 SYN (#17)
Sep 4 17:29:26 localhost kernel: Packet log: forward ACCEPT eth1 PROTO=6 192.168.90.5:1031 192.168.92.50:80 L=44 S=0x00 I=38144 F=0x4000 T=127 SYN (#1)
Sep 4 17:29:26 localhost kernel: Packet log: output ACCEPT eth1 PROTO=6 192.168.90.5:1031 192.168.92.50:80 L=44 S=0x00 I=38144 F=0x4000 T=127 SYN (#15)
Sep 4 17:29:26 localhost kernel: Packet log: input ACCEPT eth1 PROTO=6 192.168.92.50:80 192.168.90.5:1031 L=44 S=0x00 I=48384 F=0x4000 T=128 (#19)
Sep 4 17:29:26 localhost kernel: Packet log: forward ACCEPT eth0 PROTO=6 192.168.92.50:80 192.168.90.5:1031 L=44 S=0x00 I=48384 F=0x4000 T=127 (#2)
Sep 4 17:29:26 localhost kernel: Packet log: output ACCEPT eth0 PROTO=6

192.168.92.50:80 192.168.90.5:1031 L=44 S=0x00 I=48384 F=0x4000 T=127 (#13)
The first three lines show the SYN packets passing through the packet filter from the web browser to the web site and the next three show SYN packets passing through the opposite way.

These packet log entries show HTTPS traffic:
Sep 5 12:44:30 localhost kernel: Packet log: input ACCEPT eth0 PROTO=6 192.168.90.5:1056 192.168.92.50:443 L=44 S=0x00 I=64772 F=0x4000 T=128 SYN (#24)
Sep 5 12:44:30 localhost kernel: Packet log: forward ACCEPT eth1 PROTO=6 192.168.90.5:1056 192.168.92.50:443 L=44 S=0x00 I=64772 F=0x4000 T=127 SYN (#5)
Sep 5 12:44:30 localhost kernel: Packet log: output ACCEPT eth1 PROTO=6 192.168.90.5:1056 192.168.92.50:443 L=44 S=0x00 I=64772 F=0x4000 T=127 SYN (#22)
Sep 5 12:44:30 localhost kernel: Packet log: input ACCEPT eth1 PROTO=6 192.168.92.50:443 192.168.90.5:1056 L=44 S=0x00 I=17567 F=0x4000 T=128 (#25)
Sep 5 12:44:30 localhost kernel: Packet log: forward ACCEPT eth0 PROTO=6 192.168.92.50:443 192.168.90.5:1056 L=44 S=0x00 I=17567 F=0x4000 T=127 (#6)
Sep 5 12:44:30 localhost kernel: Packet log: output ACCEPT eth0 PROTO=6 192.168.92.50:443 192.168.90.5:1056 L=44 S=0x00 I=17567 F=0x4000 T=127 (#21)
The first three lines show the packets passing through the packet filter from the web browser to the web site and the next three show the packets passing through the opposite way.

Here is the DNS traffic passing through:
Sep 5 15:02:20 localhost kernel: Packet log: input ACCEPT eth0 PROTO=17 192.168.90.5:1063 192.168.92.50:53 L=54 S=0x00 I=19974 F=0x0000 T=128 (#20)
Sep 5 15:02:20 localhost kernel: Packet log: forward ACCEPT eth1 PROTO=17 192.168.90.5:1063 192.168.92.50:53 L=54 S=0x00 I=19974 F=0x0000 T=127 (#3)
Sep 5 15:02:20 localhost kernel: Packet log: output ACCEPT eth1 PROTO=17 192.168.90.5:1063 192.168.92.50:53 L=54 S=0x00 I=19974 F=0x0000 T=127 (#18)
Sep 5 15:02:20 localhost kernel: Packet log: input ACCEPT eth1 PROTO=17 192.168.92.50:53 192.168.90.5:1063 L=54 S=0x00 I=7330 F=0x0000 T=128 (#21)
Sep 5 15:02:20 localhost kernel: Packet log: forward ACCEPT eth0 PROTO=17 192.168.92.50:53 192.168.90.5:1063 L=54 S=0x00 I=7330 F=0x0000 T=127 (#4)
Sep 5 15:02:20 localhost kernel: Packet log: output ACCEPT eth0 PROTO=17 192.168.92.50:53 192.168.90.5:1063 L=54 S=0x00 I=7330 F=0x0000 T=127

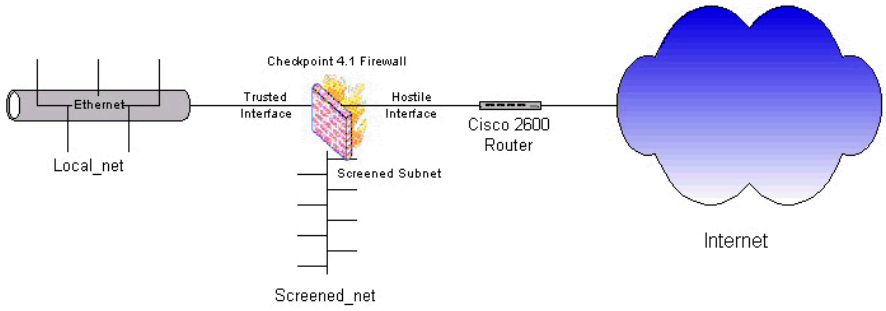
(#17)
The first three entries show the initial DNS request passing to the DNS server and the next three show packets passing back through to the requester.

Final Audit Results

The results of the four different scans performed show that the packet filter firewall is securing the network from attacks, but allows the traffic for HTTP, HTTPS, and DNS to the appropriate IP addresses.

ASSIGNMENT 4

For this assignment the practical at:
http://www.sans.org/y2k/practical/Rick_Dreger.doc will be used. This design features a Cisco 2600 router (IOS 12.x) with a CheckPoint 4.1 firewall.



1. Firewall Attack

The firewall in this design is specified as a Checkpoint 4.x. This firewall has several vulnerabilities reported at: http://ciac.llnl.gov/ciac/bulletins/k-073.shtml
The specific vulnerabilities are enumerated here.
1. One-way Connection Enforcement Bypass
2. Improper stderr Handling for RSH/REXEC
3. FTP Connection Enforcement Bypass
4. Retransmission of Encapsulated Packets
5. FWA1 Authentication Mechanism Hole
6. OPSEC Authentication Spoof
7. S/Key Password Authentication Brute Force Vulnerability
8. GetKey Buffer Overflow

The vulnerability that will be exploited will be number 5, FWA1 Authentication Mechanism Hole. A Firewall-1 that allows control connections from untrusted sources can allow an attacker to flood the authentication mechanism causing a denial of service attack. The exploit code for this attack is available from http://www.dataprotect.com/bh2000/ , and a detailed explanation is available at http://p.ulh.as/xploitsdb/Others/fw-20.html. A common misconfiguration of the Firewall-1 is to include the source address 127.0.0.1: */none as a peer address of a management module to make local administration easier. In conjunction with this misconfiguration, the Firewall-1 has a weakness in the way that authentication is done that allows an attacker to re-use the values sent in a Diffie-Hellman key exchange enabling the attacker to imitate a trusted management module. After successful authentication, a policy unload command can be given enabling the attacker to bypass the firewall.

2. DOS Attack

For a DOS attack a DDOS tool named blitznet will be used available at: http://anticode.antionline.com/download.php?dcategory=distributed-attack-tools&sortby=. This tool launches spoofed SYN flood attacks from many computers at once. To use the attack, the tool is loaded on the compromised systems and started with the command "./blitzd <port> <stealth> &" where <port> is the port you want the daemon to listen on, and <stealth> is the one word string used to mask the process in the process table. When this part is finished on the compromised computers, a file named shell.list is built on the control computer to enumerate the attacking computers IP addresses and the ports each is listening on, such as the following:
192.23.68.4 1589
165.39.36.89 2598
.
.
189.56.36.91 1259
And so on. The command to start the attack is "./rush.tcl <source address> <destination address> <start port> <end port> <# of duplicate threads of slice to run> <duration in seconds>".
Ex. ./ rush.tcl 45.45.45.45 192.168.1.255 1 600 10 400
This attack would overwhelm the network for a period of 400 seconds.
To prevent this attack from breaking the network, there is an access list addition that can be made to the Cisco 2600 that limits the TCP SYN rate detailed at http://www.cisco.com/warp/public/707/newsflash.html. Also the Router's IOS could be upgraded to a Cisco IOS Firewall feature set, and be configured for Denial of Service detection and prevention.
http://www.cisco.com/univercd/cc/td/doc/pcat/iofwfts1.htm

3. Compromise internal system.

To compromise an internal system in this design, the first attack would be to attempt to exploit another Firewall-1 vulnerability, known as TCP fast mode. The first step would be to scan the firewall to find the open ports and the to scan the network through these ports with Nmap and the –g and –sF options. The Nmap –g option sets the source port used in the scan and the –sF option uses bare FIN packets. After the hosts have been identified an operating system fingerprint would be done with a Nmap –O –g <port> <IP address range> to identify those hosts most susceptible to compromise.
Looking at the firewall ruleset it appears that only the screened subnet DNS server, the internal DNS, the screened subnet Web server, and the screened subnet Mail server would be identified.
Since BIND is commonly used on the Internet the attack to be used will target

a vulnerability in BIND version 8.2 named the TSIG bug that allows root access. http://www.securiteam.com/securitynews/Multiple_vulnerabilities_in_BIND__version_4_and_8_.html

The code for this bug can be found at:
http://www.securiteam.com/exploits/BIND_TSIG_exploit_code_released.html
This exploit gives the attacker root access, enabling him to compromise other systems, using this system as a base of operations. After compromising this system the attacker could use the same exploit to compromise the internal DNS server and "own" the internal network.
To start this process it would only be necessary to perform a name query for the authoritative name server and launch the TSIG bug attack against it. This attack works with UDP and TCP, so restricting the protocol at the firewall will not protect against it.

REFERENCES

Port Number Assignments
http://www.iana.org/assignments/port-numbers

Cisco 7204 Information
http://www.cisco.com/univercd/cc/td/doc/product/core/7204/index.htm

Cisco Security Overview
http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113ed_cr/secur_c/scoverv.htm

Cisco Configuring IP Services
http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/np1_c/1cprt2/1cip.htm - xtocid1002117

Cisco Configuration Guide Master Reference
http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/cbkixol.htm

CERT Security Improvement Modules
http://www.cert.org/security-improvement/

Information from CERT
http://www.cert.org/ftp/tech_tips/packet_filtering

IPCHAINS
http://www.linuxselfhelp.com/howtos/IPCHAINS/IPCHAINS-HOWTO.html
http://www.linuxplanet.com/linuxplanet/tutorials/1241/1

Example Script:
http://uber.chorn.com/ipchains.html

Firewall and Proxy How-To
http://www.ibiblio.org/mdw/HOWTO/Firewall-HOWTO.html

Ipchains command syntax
http://www.caldera.com/LDP/LDP/nag2/x-087-2-firewall.fwchains.html

Using IP chains to make a packet filtering firewall
http://www.syrlug.org/tutorials/ipchains.html

Intel NetStructure Information
http://support.intel.com/support/netstructure/vpn/gateway/index.htm

NMAP Home Page
http://www.insecure.org/nmap/

Syn flood
http://www.niksula.cs.hut.fi/~dforsber/synflood/result.html

Network Design for Design Under Fire
http://www.sans.org/y2k/practical/Rick_Dreger.doc

CheckPoint 4.x Vulnerabilities
http://ciac.llnl.gov/ciac/bulletins/k-073.shtml

Exploit Code and Explanation for CheckPoint Attack
http://www.dataprotect.com/bh2000/
http://p.ulh.as/xploitsdb/Others/fw-20.html

Blitznet DDOS tool
http://anticode.antionline.com/download.php?dcategory=distributed-attack-tools&sortby

Cisco 2600 TCP SYN rate limiting
http://www.cisco.com/warp/public/707/newsflash.html

BIND version 8.2 TSIG bug info
http://www.securiteam.com/securitynews/Multiple_vulnerabilities_in_BIND__version_4_and_8_.html

http://www.securiteam.com/exploits/BIND_TSIG_exploit_code_released.html