Global Information Assurance Certification Paper

SANS GIAC Certified Firewall Analyst Practical

Lonestar SANS conference, June 2001,
Dallas, Texas

Ben A. Laws, Jr.
October 11, 2001

GIAC Enterprises

www.fortunet.com

"A fortunate choice"

1 Security Architecture.

Assignment 1 - Security Architecture (15 Points)

Define a security architecture for GIAC Enterprises, an e-business which deals in the online sale of fortune cookie sayings. Your architecture must include the following components:

filtering routers;
firewalls;
VPNs to business partners;
secure remote access; and
internal firewalls.

Your architecture must consider access requirements (and restrictions) for:

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

Include a diagram or set of diagrams that shows the layout of GIAC Enterprises' network and the location of each component listed above. Provide the specific brand and version of each perimeter defense component used in your design. Finally, include an explanation that describes the purpose of each component, the security function or role it carries out, and how the placement of each component on the network allows it to fulfill this role.

1.1 Security Diagram.

The system architecture focuses upon the needs for the GIAC Enterprises business to support its fast growing fortune cookie business. Despite the fact that this business is one of the few fast growing Internet businesses, market conditions have made the startup environment less that ideal. Rather than waiting for the ideal level of financing, the business has started with the limited resources that it has to protect and apply its valuable technology. The "family jewels" of the business are three-fold; the data regarding the customers and suppliers of the fortunes, and the fortunes themselves. It is also necessary to provide a protected environment to defend against the large number of attackers who only view the computing resources as resources for attacking others, or their own nefarious purposes.

A primary tenet in the system architecture is that of "defense in depth," whereby more than one layer of security is used to block intruders. For example, router access lists begin the defense, and internally two types of firewalls are used, one as a main site firewall and a second (of a different type) at functional servers. The overworked analogy can be made to a medieval castle in a large plain, surrounded by a moat, with stone walls and iron gates that can be securely locked. (We do stop short of pouring boiling oil over the walls!) A

second tenet is that outside services are silently denied unless specifically allowed and only those services essential to the business purpose are permitted. A third tenet is that of detection and response, for each layer of security really only buys time against an intrusion. An intrusion detection system gathers data for access by a separate security server via encrypted connections. A fourth tenet is that encrypted services via ssh are used internally rather than unencrypted services such as telnet and ftp to prevent local sniffing of user identities and passwords; internal systems are also hardened to withstand attack. A final tenet is that good practices are used to prevent the site from being used as an attack platform by keeping abreast of vulnerabilities, application of appropriate vendor patches and by proper application of egress filtering. It is important for all Internet sites to be "good neighbors" in preventing unauthorized use of their own site in attacking other sites.

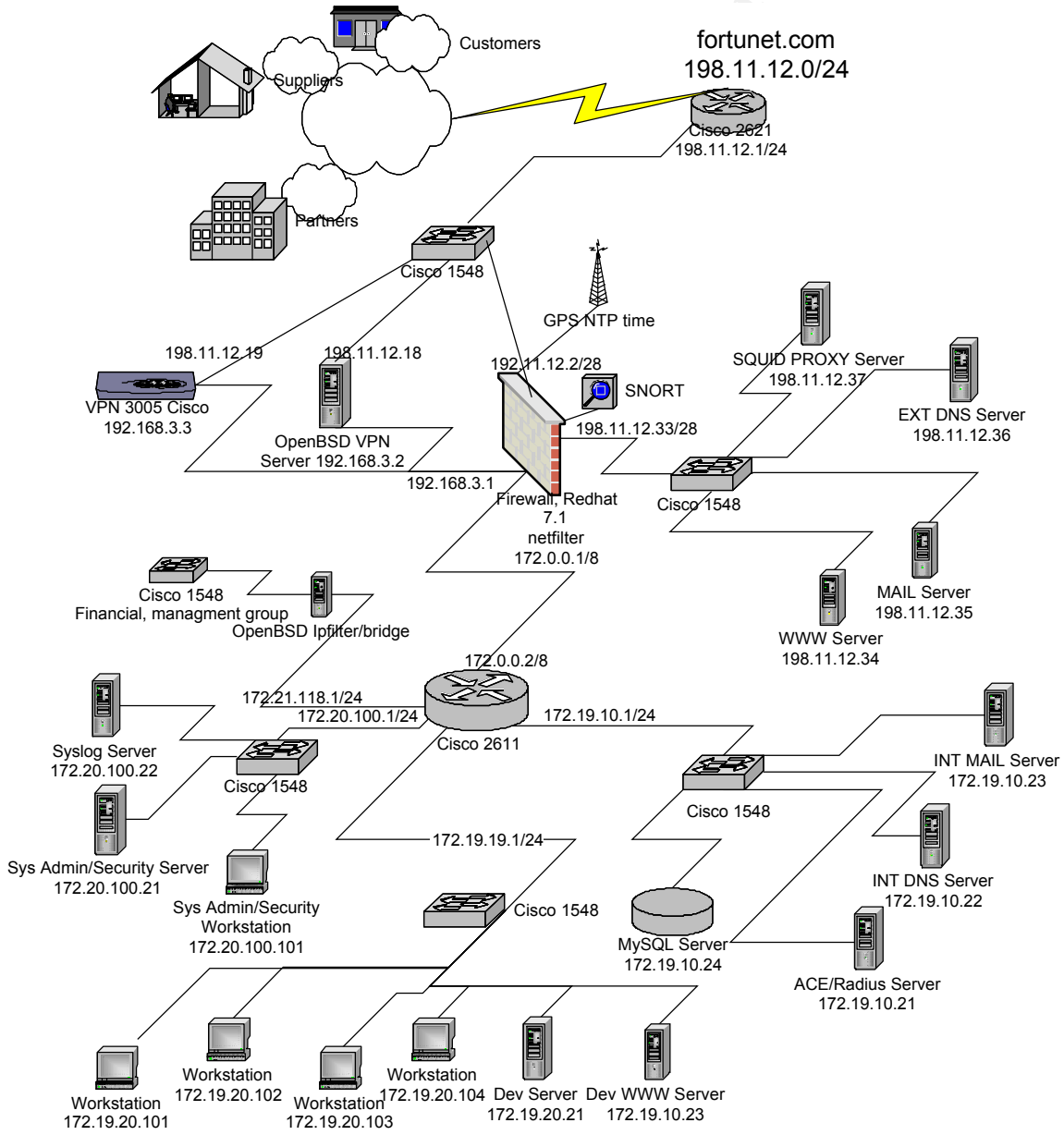


Figure 1-1, System Architecture.

1.2 System Overview

Figure 1-1 shows the System Architecture. The key features are the border router, the main firewall, the VPN service network, which allows partner and secure remote access, the protected services network and the protected development network. Each service system is further protected with its own IPFilter firewall. The filtering border router, placed between all internal components and the internet feed, is a Cisco 2611 with one high speed serial port and two high speed Ethernet ports. The next level of defense is provided by the main firewall, a Redhat Linux 7.1 system running the 2.4 kernel with NetFilter/IPTables for the firewall. This server is up to current patch levels and the "Bastille Linux" (http://www.bastille-linux.org/)scripts are used to harden it. It is configured with a RAID disk drive using SNORT and Tripwire for intrusion detection. The Tripwire database should be backed up on read-only media once it is complete, for use later in auditing the installation. This system screens all access to internal networks, except for the VPN systems' external interfaces. The VPN systems are an OpenBSD server running the IPFilter firewall, with IPSEC VPN support for the partner connections, and a Cisco 3005 VPN appliance for the customers and the suppliers running Windows applications. The protected service network is used to provide the Web server, the outside DNS service, the outside Mail server and the Squid outbound proxy server (all are hardened Solaris 8 running Tripwire) for internal WWW users. All protected net systems use IPFilter and TCP wrappers to control and log access. All servers allow for SSH connections from the internal Sys Admin server; TELNET and FTP accesses are specifically dropped and possibly logged. Time service is provided directly by a GPS receiver to allow the Firewall to serve as an NTP Stratum 1 time server for the site. Each functional group in the company is given a private IP range from the 172.xx.xx.yy address range; a Cisco 2611 router with quad Ethernet adapter (for a total of six fast Ethernet interfaces) is used to connect the segments.

In the future, connections for customers will be made via a VPN connection using the Cisco (Altiga) 3005 concentrator; current fiscal limitations have prevented this solution. This provides a very straightforward connection for Windows based systems using SecurID for two factor authentication via the RADIUS server. In the meantime, access for customers will be provided via the NAI PGP desktop, which includes IPSEC VPN and the OpenBSD VPN server. Customers are allowed to access the SQL server in order to obtain the download batches of fortunes for their site.

Partner connections are made to the OpenBSD server via IPSEC connections. These connections are established with shared secrets which are changed monthly by certified courier hard copy exchange. As the number of partner sites is limited, there is no reason to use a PKI based certificate system for the foreseeable future. Partner VPN connections are limited to the development server network.

Finally, the supplier contributions are made via a secure web page using SSL support.

This allows for collaboration between the suppliers and the internal reviewers, all using SSL encrypted Web browser connections.

Hardened Solaris 8 servers provide external system services for DNS, Mail and Web service. IPfilter and TCP wrappers access lists protect each system. Cisco 1548 eight-port switches are used to provide a "star" network to make sniffing more difficult.

All syslog data are directed from the VPN and Protected Services Networks to an internal syslog server within the Sysadmin/Security internal network via port forwarding to via the appropriate firewall IP address/port syslog(513). These ports are forwarded by the firewall to the internal syslog server. Outside Mail is received in the Protected Services network Email server. Access from the internal network is via an SSH poll, every two minutes. The external (protected services) systems are backed up via rsync and SSH during daily polls.

2 Security Policy.

Assignment 2 - Security Policy (35 Points)

Part 1 – Define Your Security Policy (25 points)

Based on the security architecture that you defined in Assignment 1, provide a security policy for AT LEAST the following three components:

Border Router
Primary Firewall
VPN

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

By `security policy' we mean the specific ACLs, firewall ruleset, IPSec policy, etc. (as appropriate) for the specific component used in your architecture. For each component, be sure to consider internal business operations, customers, suppliers and partners. Keep in mind you are an E-Business with customers, suppliers, and partners - you MAY NOT simply block everything!

You must include the complete policy (ACLs, ruleset, IPSec policy) in your paper. It is not enough to simply state "I would include ingress and egress filtering..." etc. The policies may be included in an Appendix if doing so will help the "flow" of the paper.

(Special note VPNs: since IPSec VPNs are still a bit flaky when it comes to implementation, that component will be graded more loosely than the border router and primary firewall. However, be sure to define whether split-horizon is implemented, key exchange parameters, the choice of AH or ESP and why. PPP-based VPNs are also fully acceptable as long as they are well defined.)

Part 2 - Security Policy Tutorial (10 points)
Select one of the three security policies defined above and write a tutorial on how to implement the policy. Use screen shots, network traffic traces, firewall log information, and/or URLs to find further information as appropriate. Be certain to include the following:

1. A general explanation of the syntax or format of the ACL, filter, or rule for your device.
2. A general description of each of the parts of the ACL, filter, or rule.
3. An general explanation of how to apply a given ACL, filter, or rule.
4. For each ACL, filter, or rule in your security policy, describe:
   - the service or protocol addressed by the rule, and the reason this service might be considered a vulnerability.
   - Any relevant information about the behavior of the service or protocol on the network.
   - If the order of the rules is important, include an explanation of why certain rules must come before (or after) other rules.
5. Select three sample rules from your policy and explain how you would test each rule to make sure it has been applied and is working properly.

Be certain to point out any tips, tricks, or potential problems ("gotchas").

2.1 Border Router Configuration and Testing

First the border router configuration will be defined and tested in an isolated lab setting. Before any "live" (i.e. Internet connected) testing is done, written permission should be obtained from the organization being tested. These tests should be arranged so as not to interfere with the business activities of the organization.

2.1.1 Border Router Access Control List (ACL) definitions

The border router ACL has two access control lists, one for inbound addresses and one for outbound (egress filtering). Standard access lists are applied to reduce processing load on the router wherever possible, while extended access lists are used for fine grained control. The router configuration is shown in Figure 2-1. See [Brenton01], and [Brenton02] for more detailed explanation of the Cisco configuration and commands, or on the web site, www.cisco.com.

```
# config t
# hostname starburst
! First disable all non-essential services
# no service tcp-small-servers
# no service udp-small-servers
# no service finger
# no ip http server
! Disable the Router's discovery protocols, use static
! routing.
# no cdp run
! NTP (Network Time Protocol) is not needed, we have
! our own NTP GPS clock.
# ntp disable
! Save DNS accesses
# no ip domain-lookup
! Don't allow source routed packets at all.
# no ip source-route
#
!
! Now establish timestamps and logging.
# service timestamps debug uptime
# service timestamps log uptime
# service password-encryption
# logging buffered 4096 debugging
# logging console emergencies
# logging 192.11.12.14
! Setup encrypted passwords
# enable secret 5 $1$LCro$vhTpC4dllE.DCZwHF.CW3/
# enable password 7 14381745A6361E6A
! Define the external (ISP) serial interface
# interface s0
# encapsulation ppp
! < possibly other commands related to the ISP >
! Allow the internal router port to be used as the
! visible Internet address
# ip unnumbered 0
^Z
! Now setup the internal port
# interface FastEthernet0/0
```

```
! Prevent inside SMURF attacks
# no ip directed-broadcast
! Static routing is used internally, no redirects allowed.
# no ip redirects
! Disable both SNMP and Router Discover Protocols
! (we'll use static route tables)
# no snmp
# no cdp enable
! Allow forwarding for supernets
# ip classless
# ip subnet-zero
#
```

Figure 2-1, Border Router basic configuration

Next, we restrict the number of services that are available from our site using Access Control Lists (ACLs). This list is modeled after the one from [Brenton01], pp. 213-224. The ACLs make use of both the Standard access list for their speed, and Extended Access Lists for finer grain control. Our policy for the border router is to allow all outbound connections including TELNET and FTP, and to allow only inbound access for SMTP, HTTP and HTTPS, and DNS. In addition we allow the IPSEC ESP (Encapsulation Payload protocol 50) and the IKE (Internet Key Exchange, port udp 500). The AH (Authentication Header, protocol 51) capability of IPSEC is not used. The ACLs for our border router are shown in Figure 3.

Standard Access List syntax is ([Brenton01], p. 217):

```
access-list {list # or name} permit/deny {source} {mask}

list # 1-99 designate static lists
source is a standard IP address
mask is a "wild card" match, essential an inverted IP subnet mask.
```

Extended Access List syntax is ([Brenton01], pp. 219):

```
access-list {list # or name} permit/deny {protocol} {source} {mask}
{destination} { mask} {operator <lt, gt, eq, neq>} {port}
[established]
list # 101-199 designate extended lists
protocol is tcp, esp, icmp, etc.
source, destination are standard IP addresses
mask is a "wild card" match, essential an inverted IP subnet mask.
operator is selected from:
    lt for "less than"
    gt for "greater than"
    eq for "equal to"
    neq for "not equal to"
port is the tcp or udp service port number
established is an optional keyword meaning established connections
(SYN is zero)
```

Rules are applied in the order given, so the most frequently used rules should probably be applied first, realizing that the order may allow patterns that appear to be blocked later! Figure 2-2 shows the access controls which are applied to the Serial0 interface. It may be

a good idea to do the initial definition of these lists via the serial console port, since it is easy to cut off TCP/IP access if a mistake is made during their definition. An access-list remark is used to remind ourselves that the default policy is to deny everything that's not permitted in the ACL!

```
# config t
! First, don't allow any source claiming to be from inside!
# access-list 101 deny ip 198.11.12.0 0.0.0.255 any log
! Also disallow any private or unassigned sources
# 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
# access-list 101 deny ip 127.0.0.0 0.255.255.255. any log
# access-list 101 deny ip 0.0.0.0 any log
# access-list 101 deny ip 240.0.0.0 15.255.255.255 any log
!Now protocol specific
! Allow established connection replies (SYN == 0)
# access-list 101 permit tcp any 198.11.12.0 0.0.0.255 gt 1023 est
! Allow IPSEC/IKE traffic
# access-list 101 permit tcp 198.11.12.18 any eq 22
# access-list 101 permit udp 198.11.12.18 any eq 500
# access-list 101 permit udp 198.11.12.19 any eq 500
# access-list 101 permit esp 198.11.12.18 any
# access-list 101 permit esp 198.11.12.19 any
! Suggested as an "early warning" for a port scan
# access-list 101 deny tcp any any eq 19 log
! Now allow the Mail server to receive connections for SMTP
# access-list 101 permit tcp any 198.11.12.35 eq 25
! Likewise for the DNS traffic to the external DNS
! Restrict TCP access to our ISP
# access-list 101 permit tcp 66.66.66.90 198.11.12.36 eq 53
# access-list 101 permit tcp 66.66.66.100 198.11.12.36 eq 53
# access-list 101 permit udp any 198.11.12.36 eq 53
! Also allow the Web server to receive connections
# access-list 101 permit tcp any 198.11.12.34 eq 80
# access-list 101 permit tcp any 198.11.12.34 eq 443
! Allow inside icmp echo requests and traceroutes
# access-list 101 permit icmp any any echo-reply
# access-list 101 permit icmp any any time-exceeded
# access-list 101 permit icmp any any unreachable
! Don't forget that now the default is to deny!
# access-list 101 remark remember default deny ip any any
!
! Egress anti-spoof filter:
!
# access-list 111 permit ip 198.11.12.0 0.0.0.255 any
! Block exiting traceroute from others
# access-list 111 deny icmp any any time-exceeded
! Don't forget that now the default is to deny!
# access-list 111 remark remember default deny ip any any
!
! Now apply the access list to the interface
!
# interface serial0
# ip access-group 101 in
# ip access-group 111 out
^Z
```

Figure 2-2, Access control for Serial0

Additional Access controls are now defined for the internal Fast Ethernet interface, as shown in Figure 2-3.

```
! Allow replies from the web/mail/DNS servers TODO
```

```
# access-list 102 permit tcp 198.11.12.34 any gt 1023 est
# access-list 102 permit tcp 198.11.12.35 any gt 1023 est
# access-list 102 permit tcp 198.11.12.37 any gt 1023 est
! Allow replies from the DNS servers
# access-list 102 permit udp 198.11.12.36 any eq 53
# access-list 102 permit tcp 198.11.12.36 66.66.66.90  eq 53
# access-list 102 permit tcp 198.11.12.36 66.66.66.100  eq 53
! Allow IKE traffic to the VPN servers
# access-list 102 permit tcp 192.11.12.18 any eq 22
# access-list 102 permit udp 198.11.12.18 any eq 500
# access-list 102 permit udp 198.11.12.19 any eq 500
# access-list 102 permit esp 198.11.12.18 any
# access-list 102 permit esp 198.11.12.19 any
! Block all other UDP traffic
# access-list 102 deny udp 198.11.12.0 0.0.0.255 any
! Allow connections from the internal Sysadmin network
### Upgrade this to SSH if the router supports it!
# access-list 102 permit tcp host 172.20.100.100 host 198.11.12.1 eq 23
! Block all other Telnet traffic to the router
# access-list 102 deny tcp any 198.11.12.1 0.0.0.0 eq 23
! Allow all other ip traffic to enter
# access-list 102 permit ip 198.11.12.0 0.0.0.255 any
! Don't forget that now the default is to deny!
# access-list 102 remark remember default deny ip any any
# interface FastEthernet0/0
# ip access-group 102 in
^Z
```

Figure 2-3, Access control for FastEthernet0/0

2.1.2 Testing the Border Router

The first tests are done from the router console port. Verify that ping, telnet and traceroute work both to internal and external addresses. Figure 2-4 shows the expected results.

```
#ping 192.81.77.5
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.81.77.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 52/59/68 ms
#
#traceroute 192.81.77.5
Type escape sequence to abort.
Tracing the route to 192.81.77.5

  1 124.17.7.201 28 msec 16 msec 36 msec
  2 152.63.100.218 36 msec 56 msec 36 msec
  3 152.63.101.253 12 msec 12 msec 56 msec
  4 152.63.0.193 28 msec 20 msec 36 msec
  5 152.63.2.42 72 msec 88 msec 84 msec
  6 152.63.9.69 84 msec 428 msec 92 msec
  7 152.63.91.205 120 msec 88 msec 84 msec
  8  *  *  *
  9 192.81.77.5 56 msec *  88 msec
#
!verify connections out
#telnet 192.81.77.5
#telnet 192.81.77.5 25
#telnet 192.81.77.5 80
#telnet 192.81.77.5 . . .
#
```

Figure 2-4, Testing from the border router console

Next, from an outside Internet address, scan the router port with the excellent port scanning tool **nmap**, available from http://www.insecure.org. Be sure that you have written permission from the organization being scanned before proceeding. Figure 2-5 shows the scan and typical results. There are two options used, -P0 tells nmap not to use icmp ping and –sS says to do a "stealth" SYN scan. Note that our route does not have any ports open—exactly what we want.

```
[root@benzdiamond /root]# nmap -P0 -sS 198.11.12.1

Starting nmap V. 2.54BETA26 ( www.insecure.org/nmap/ )
All 1548 scanned ports on 198.11.12.1 (198.11.12.1) are: filtered


Nmap run completed -- 1 IP address (1 host up) scanned in 2263 seconds
[root@benzdiamond /root]#
```

Figure 2-5, Scanning the Border router with nmap

2.2 Iptables firewall configuration

A very useful description about the new netfilter subsystem for Linux 2.4 kernels is found in http://www.gnumonks.org/papers/netfilter-lk2000/presentation.html which describes the order of processing for packets in the kernel. A great resource for starting an Iptables configuration (netfilter) can be found at http://www.cs.princeton.edu/~jns/security/iptables/index.html which was used as a starting point for this ruleset. The rules for the main firewall are shown in Figure 2-6.

```
# First take care of global definitions:
INET_IFACE=eth0;export INET_IFACE
IPADDR=198.11.12.2/32
BROADCAST="198.11.12.255"

VPN_IFACE=eth2;export VPN_IFACE
VPN_IPADDR=192.168.3.3;export VPN_IPADD

SVC_IFACE=eth3;export SVC_IFACD
SVC_IPADDR=198.11.12.33;export SVC_IPADD

LAN_IFACE=eth1;LAN_IFACE
LAN_IP_RANGE="172.0.0.0/24"
LAN_IP="172.0.0.1/32"

IPTABLES=/sbin/iptables

# Load the required Linux netfilter modules
modprobe ip_tables
modprobe ip_conntrack
modprobe ip_conntrack_ftp

# Now clear out any existing rules so that we can start wit a "clean
slate..;"
$IPTABLES -F
$IPTABLES -X
$IPTABLES -Z

# Now make the default policy to be (silently) DROP:
$IPTABLES -P INPUT DROP
```

```
$IPTABLES -P FORWARD DROP
$IPTABLES -P OUTPUT DROP
```

Figure 2-6, Main Firewall definitions

In addition, Figure 2-7 shows a number of kernel settings for the 2.4 kernel that enhance security.

```
# Kernel setup for main firewall.
# Don't respond to "ping."
/bin/echo "0" > /proc/sys/net/ipv4/icmp_echo_ignore_all

# Disable response to icmp broadcasts, used in "smurf" attacks
/bin/echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# Disable source routing
/bin/echo "0" > /proc/sys/net/ipv4/conf/all/accept_source_route

# Disable ICMP redirects, we use static routes.
/bin/echo "0" > /proc/sys/net/ipv4/conf/all/accept_redirects

# Enable bad error message protection.
/bin/echo "1" > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses

# Enable reverse path filtering.

for interface in /proc/sys/net/ipv4/conf/*/rp_filter; do
   /bin/echo "1" > ${interface}
done

# Log spoofed packets, source routed packets, redirect packets.
/bin/echo "1" > /proc/sys/net/ipv4/conf/all/log_martians

# Enable IP forwarding since this is a multi-homed host.
/bin/echo "1" > /proc/sys/net/ipv4/ip_forward
```

Figure 2-7, Kernel settings for the Linux 2.4 kernel.

Before proceeding, it's helpful to review the Netfilter processing model (http://www.linuxguruz.org/iptables/howto/). From this document, we see that the model basic for packet filtering is as shown in Figure 2-8. There are also PREROUTING and POSTROUTING decision points available in the model, not shown here.

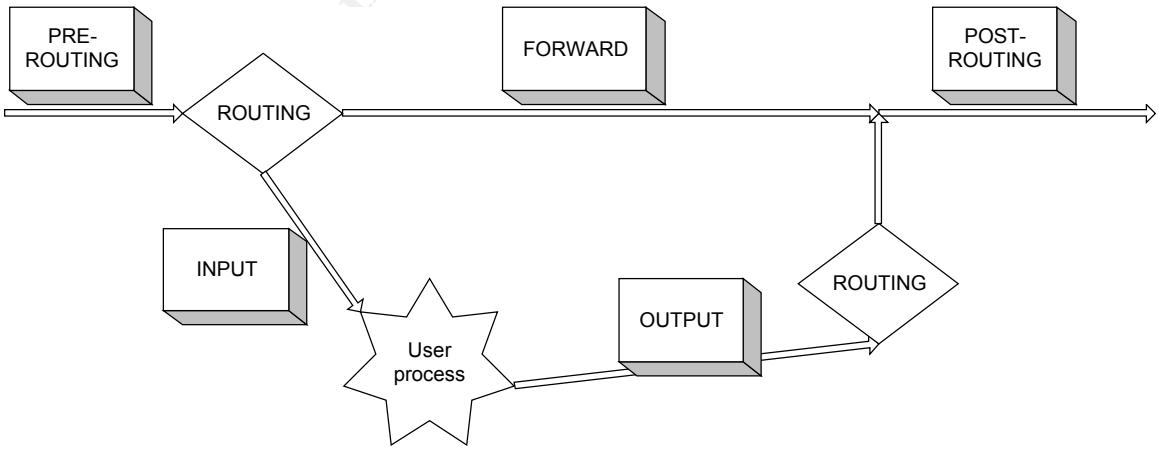


Figure 2-8, IPTables processing model.

2.3 NetFilter/IPTables firewall configuration

The Netfilter/IPTables firewall provides more than simple IP filtering, it also allows the tracking of connection state through loadable modules. The main Netfilter web page is http://netfilter.samba.org/. Other great sources of information about NetFilter/IPTables can be found at http://www.boingworld.com/workshops/linux/iptables-tutorial/index.html and http://www.knowplace.org/netfilter/. There is also a graphical tool, fwbuilder, that is extremely useful in setting up a network of this complexity (http://www.fwbuilder.org). The output from this tool, in annotated form, is in the Appendix due to its length. Please note that this tool is under active development, and the output should be carefully checked before it is applied to a real firewall. Even considering this, the tool provides a very logical framework for developing the iptables commands. First, the interfaces must be defined for the main firewall as shown in Figure 2-9.

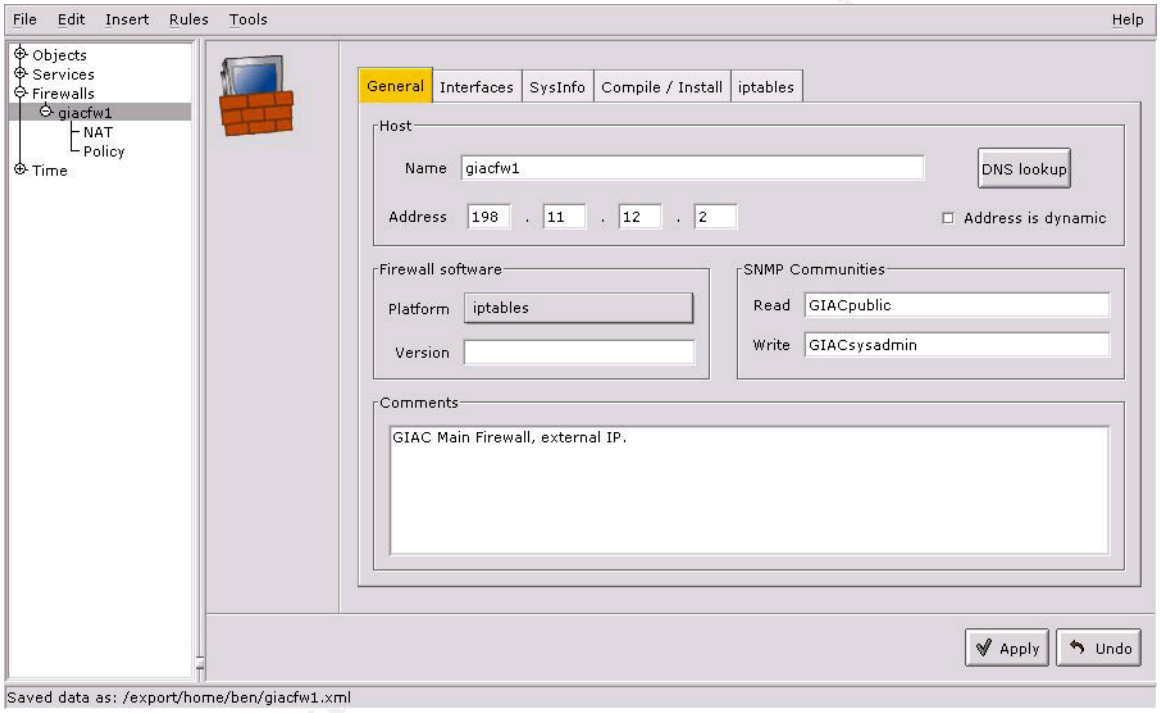


Figure 2-9, Main firewall external interface definition

Secondly, we define the four Ethernet interfaces and the internal loopback interface as shown in Figure 2-10. The details of the definitions of the various interfaces are not shown graphically, and correspond to the System Architecture. There are two rules associated directly with this interface, defined by the "Help me firewall build policy" feature of this program. The first rule prevents spoofing of any of our protected addresses and the second prevents any of the protected addresses from being exposed on the external interface. Both of these are logged, and options are used to disable state tracking. Two rules were added to the loopback interface to allow unlimited local traffic; the other interfaces don't have any special rules.

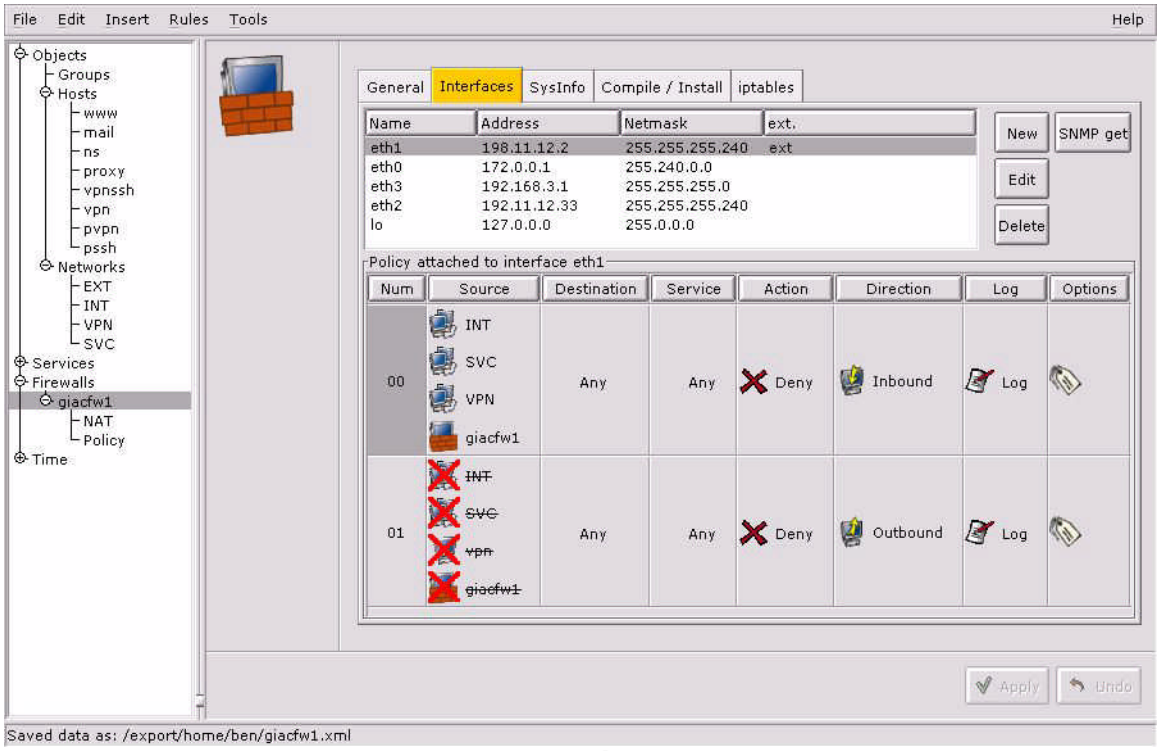


Figure 2-10, Main Firewall Interface definitions.

To save space, we have pre-defined the various networks required where EXT is eth1, (198.11.12.2), INT is eth0 (172.0.0.1), SVC is eth2 (192.11.12.33) and VPN is eth3 (192.168.3.1).

The FW policy is shown in Figure 2-11. The rules shown as "block fragments", "'masquerading' rule" and "'catch all' rule" are defined by the "help me build firewall policy" that is built in to the program. Since the rules are processed in order of appearance, it's important to have the "'catch all' rule" last so that the final action is to deny access. The "block fragments" rule simply denies any packets that have extremely small IP fragmentation, since this has been a source of so many problems in other firewalls. Rules "Reduce visibility of Firewall" are defined to reduce the visibility of the firewall to icmp probes. The group Useful_ICMP contains "Time exceeded (Type 11, Code 0)," "Time exceeded in transit (Type 11, Code 1)," "ping reply (Type 0, Code 0)," and "all icmp unreachables (Type 3, Code any)." The "'masquerading' rule" allows the internal systems to be externally masqueraded. Since DNS access is required by nearly every system, DNS udp access is allowed complete access by the next rule. The "Syslog server access" rule allows access to the internal syslog server from the VPN and SVC networks and the firewall itself. The next rule allows the WWW server access to the MySQL server. Next, we allow the VPN servers to access the internal network VPNDEV (172.19.0.0/16) for product development. Only the internal (pvpn, 192.168.3.2 and pssh, 192.168.3.3) interfaces are allowed access. The external address for these systems is connected directly to the border router. Finally, the default policy for the main firewall is "'catch all'", to drop and log all other packets. These settings for logging may be

voluminous, and can be controlled by options available if so desired.

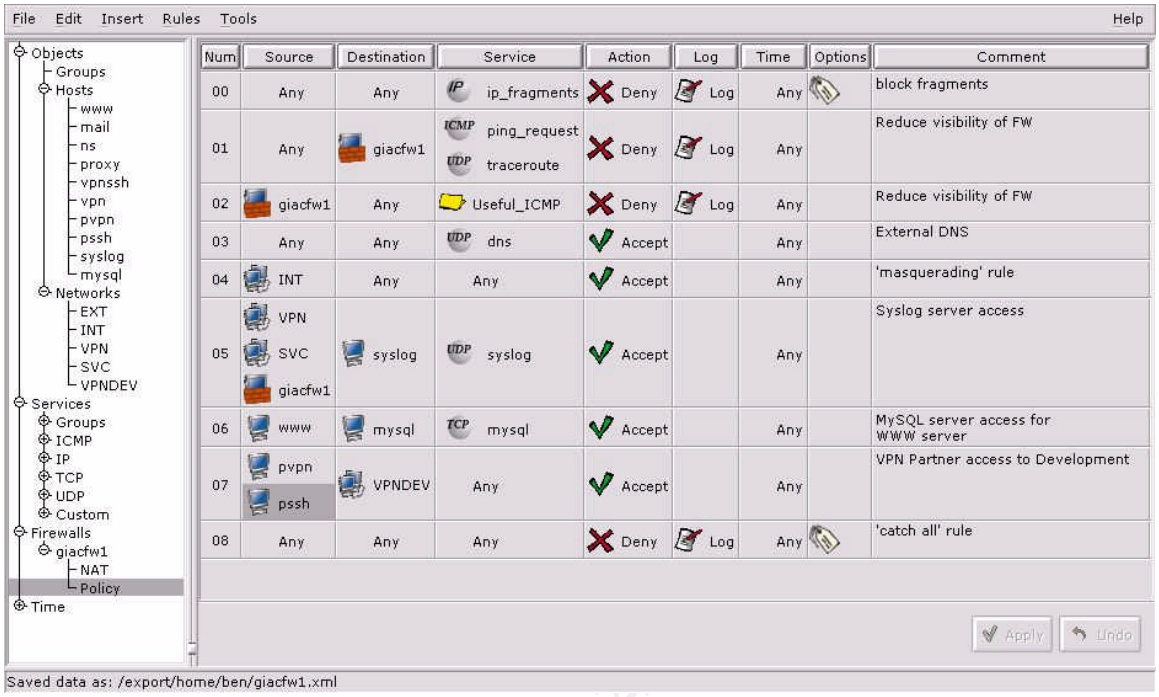


Figure 2-11, Main firewall policy.

Finally, we define the Address Translation for our internal users to be able to access the external Internet in Figure Figure 2-12.

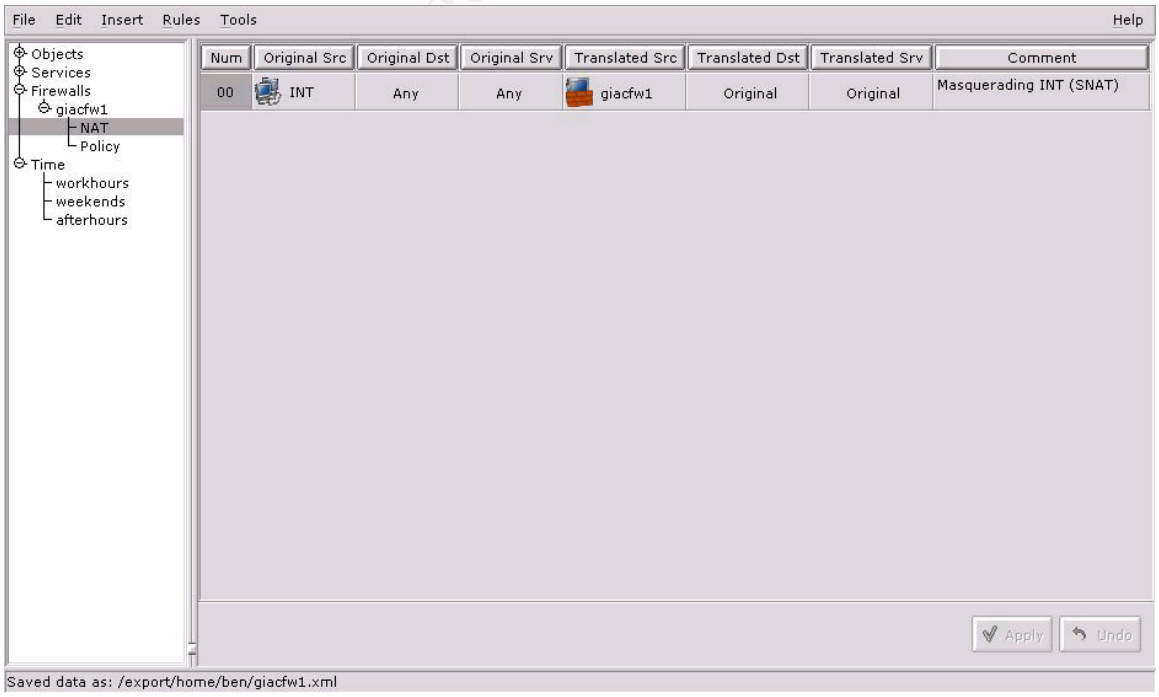


Figure 2-12, Address Translation

2.4 VPN configuration

The VPN IPSEC configuration is shown in the Appendix in order to save space here. This configuration shows the basic setup to allow a partner site running a matching OpenBSD server to access the local development network. This can be combined to allow customer site access via the excellent NAI PGP desktop which includes VPN support. There are numerous excellent websites http://www.linuxsecurity.com/resource_files/cryptography/ipsec-howto/HOWTO.html , http://kubarb.phsx.ukans.edu/~tbird/vpn/vpn-howto.html , http://www.allard.nu/openbsd/. The last website provided files that can be directly used in setting up the VPN connection to PGP desktop. The excellent OpenBSD documentation provides sample configuration files that can be used as a starting point in establishing a secure connection to the partner sites. In order to setup a VPN under OpenBSD, three files must be defined. The first, /etc/isakmp/isakmpd.policy, defines the acceptable policies available for our VPN. The /etc/isakmp/isakmpd.conf file defines the specific details of these policies. Finally, additional firewall rules are supplied to to control access by the remote networks. Initially, the setup is fairly non-restrictive in order to test the VPN; finally, the policies and firewall configuration are tightened to allow only the desired internal access.

2.5 VPN Firewall configuration

The VPN system uses its own firewall configuration in order to be more secure. It also prohibits outbound connections to the Internet, but allows access to those internal resources needed by partners. The older version of fwbuilder (0.8.7) can be used to generate an acceptable policy as shown in Figure 2-13. The details are so similar to the WWW server that only those actual commands will be shown below in Figure 2-15.

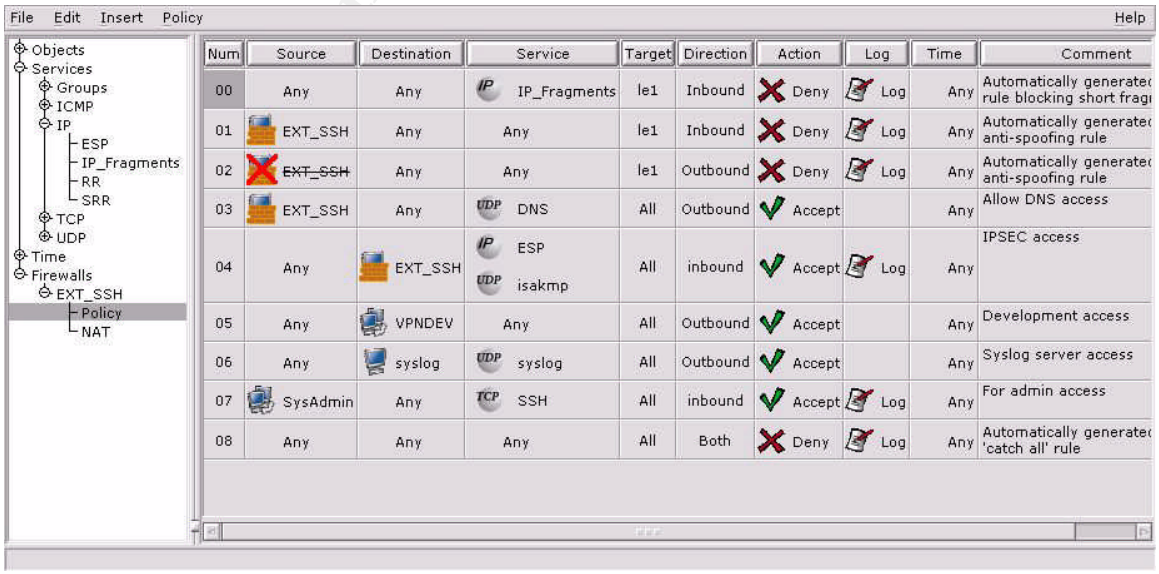


Figure 2-13, VPN ipfilter Firewall Configuration

2.6 WWW Firewall configuration

The WWW server also uses its own ipfilter (ipf) OpenBSD firewall configuration in order to be more secure (Figure 2-14). It also prohibits outbound connections to the Internet, but allows access to those internal resources needed by partners. The older version of fwbuilder (0.8.7) can be used to generate an acceptable policy. The name EXT_WWW (192.11.12.34) was predefined as the local firewall host IP address on le1.The detailed ipfilter configuration is shown in Figure 2-15.

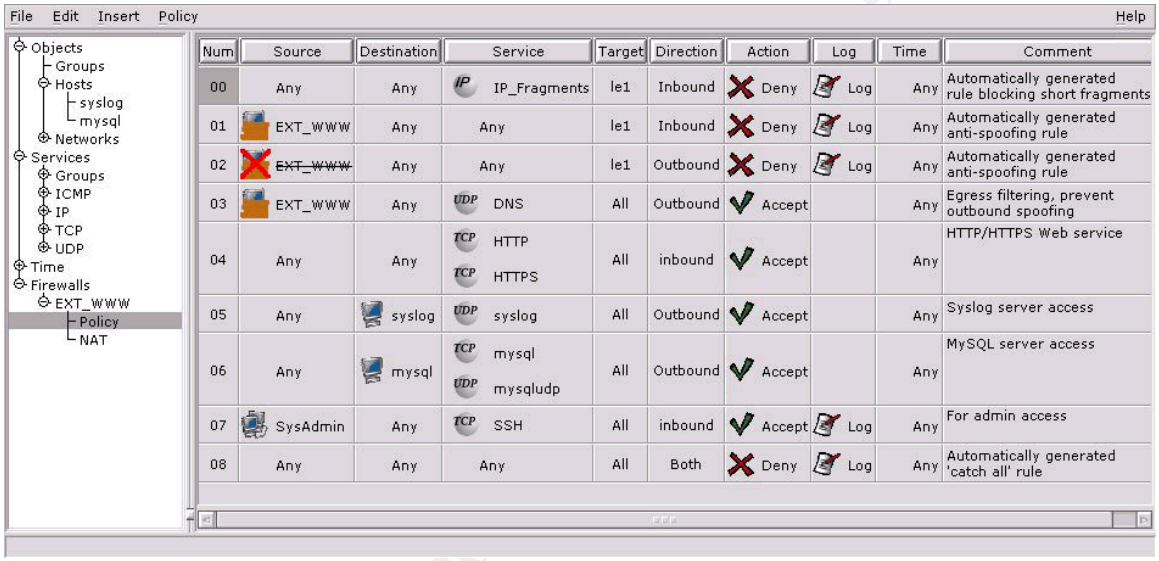


Figure 2-14, WWW ipfilter Configuration

```
# Interface le1, incoming packets: group 100
block in on le1 all head 100
# Interface le1, outgoing packets:  group 105
block out on le1 all head 105
#    Rule #0
#    Automatically generated rule blocking short fragments
#    subrule  #0
block in log quick  proto ip from 0/0 to 0/0 with short  group 100
#    Rule #1
#    Automatically generated anti-spoofing rule
#    subrule  #0
block in log quick  from 198.11.12.34/32 to 0/0  group 100
#    Rule #2
#    Automatically generated anti-spoofing rule
#    subrule  #0
skip 1 out from 198.11.12.34/32 to 0/0  group 105
### The next rule seems to conflict with the final policy rule under
### OpenBSD. The "quick" modifier was dropped on that rule.
block out log quick  all  group 105
#    Rule #3
#    Allow DNS access
#    subrule  #0
pass out  quick  proto udp from 198.11.12.34/32  to 0/0 port = 53 keep
state  group 105
#    Rule #4
#    HTTP/HTTPS Web service
```

```
pass in  quick  proto tcp from 0/0  to 0/0 port = 80 flags S keep state
group 100
pass in  quick  proto tcp from 0/0  to 0/0 port = 443 flags S keep state
group 100
#
#    Rule #5
#    Syslog server access
pass out  quick  proto udp from 0/0  to 172.20.100.22/32 port = 514 keep
state  group 105
#
#    Rule #6
#    MySQL server access
pass out  quick  proto tcp from 0/0  to 172.10.19.10.24/32 port = 3306
flags S keep state  group 105
pass out  quick  proto udp from 0/0  to 172.10.19.10.24/32 port = 3306 keep
state  group 105
#
#    Rule #7
#
#    For admin access
pass in log quick  proto tcp from 172.20.100.0 mask 255.255.255.0  to 0/0
port = 22 flags S keep state  group 100
#
#    Rule #8
#    Automatically generated 'catch all' rule
block in log quick  from 0/0 to 0/0  group 100
### Modified to prevent conflict with the "skip" rule above:
####block out log quick  from 0/0 to 0/0  group 105
block out log from 0/0 to 0/0  group 105
```

Figure 2-15, WWW Detailed ipfilter Configuration

3 Security Audit.

Assignment 3 - Audit Your Security Architecture (25 Points)

You have been asked to provide a technical audit of the **primary firewall** (described in Assignments 1 and 2) for GIAC Enterprises. In order to conduct the audit, you will need to:

1. Plan the audit. Describe the technical approach you recommend to assess the firewall. Be certain to include considerations such as what shift or day you would do the assessment. Estimate costs and level of effort. Identify risks and considerations.
2. Conduct the audit. Using the approach you described, validate that the Primary Firewall is actually implementing the GIAC Enterprises' security policy. Be certain to state exactly how you do this, including the tools and commands used. Include screen shots in your report if possible.
3. Evaluate the audit. Based on your assessment (and referring to data from your assessment), analyze the perimeter defense and make recommendations for improvements or alternate architectures. Diagrams are strongly recommended for this part of the assignment.

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

3.1 Security Audit plan for the main firewall.

The Security Audit for the main firewall will be done in three parts. The first part will check the basic system installation for current patches and general configuration. The second part of the audit will be done from a third-party ISP to allow external functional testing. Next the inside configuration will be verified. Finally, VPN connections will be checked to make sure that the access is restricted as desired.

Before this plan can be implemented, we will obtain written acknowledgment of this plan, which includes permission to proceed with the audit, by the company management.

The audit will take approximately three days to complete, with two people working together; most of the work will be performed outside of normal working hours so as not to disrupt daily activity.

3.2 Conducting the audit.

The system patches can be checked by going to the RedHat website (http://www.redhat.com/) and viewing the security advisories for RedHat 7.1. The best way to keep up to date is to subscribe to the RedHat network and run the "up2date" package as shown in Figure 3-1.

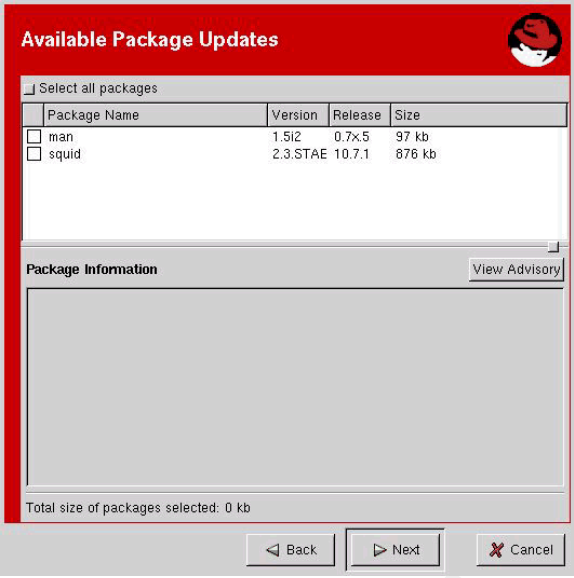


Figure 3-1, RedHat Network update

In particular, there was a critical patch for the netfilter proxy module that checks FTP connections, an exploit was found that allows arbitrary ports to be opened on the firewall without it (http://www.tempest.com.br/advisories/01-2001.html). Additional searching can be done at the Security Focus (http://www.securitfocus.com) site which has the BugTraq and Vuln-dev email lists. These lists can be searched for "iptables" or "netfilter" to track known bugs and vulnerabilities. Since we are not running an FTP server, this is not of immediate concern, but fits with the overall philosophy of "defense in depth." Checking the RedHat site, we find that the patch for this is installed correctly. The SMTP service is provided by the "qmail" program, hence recent "sendmail" exploits are of no concern. Another source of vulnerabilities is the DNS server (bind); the version running will be checked for any known issues. The Apache webserver is also up to date, and there are no extra cgi-bin scripts available to an attacker. The Tripwire system allows us to verify that the basic installation is sound; the database file was saved after the installation on read-only media.

Using publicly available information, we begin to build information about this site. Information is obtained from the "whois" database, the "ARIN" database, the company's own web server. A search of various free and for-fee informational services can also provide information that an attacker might find useful.

The "nslookup" and "dig" commands can be used to verify that our DNS is operating correctly and not allowing "zone" transfers to external sites. The excellent port scanning tool "nmap" is invaluable in verifying the port and network status of our system.

3.3 Evaluation of the audit.

Our audit has show that externally, our network appears as we desire. Only the required

ports and protocols for Web service, DNS, SMTP and VPN service are available as desired, they are assigned to the desired IP addresses. This was determined with an "nmap" scan:

```
### Scanned from vpnssh, 192.168.3.3
# nmap -sS -P0 192.168.3.1

Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
All 1542 scanned ports on  (192.168.3.1) are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 2729
seconds


### no Ping from vpnssh to the firewall, 192.168.1.1

# ping 192.168.1.1
PING 192.168.1.1 (192.168.2.31): 56 data bytes
--- 192.168.1.1 ping statistics ---
40 packets transmitted, 0 packets received, 100% packet loss

### Verify that the syslog port is open
# nmap -P0 -sU -p514 172.20.100.22

Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
Interesting ports on sunblade.benlaws.com (172.20.100.22):
Port       State       Service
514/udp    open        syslog


Nmap run completed -- 1 IP address (1 host up) scanned in 12 seconds



### no Ping from vpnssh to extwww

# ping 198.11.12.34
PING 198.11.12.34: 56 data bytes
--- 198.11.12.34 ping statistics ---
40 packets transmitted, 0 packets received, 100% packet loss

### nmap scan from vpnssh to service net
# nmap -sS -P0 198.11.12.0/28

Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
All 1086 scanned ports on  (198.11.12.1) are: filtered
All 1086 scanned ports on  (198.11.12.2) are: filtered
. . .
All 1542 scanned ports on  (198.11.12.15) are: filtered

Nmap run completed -- 16 IP address (16 host up) scanned in 2729
seconds


### no Ping from vpnssh to the firewall, 192.168.3.1
```

Next we scan the main firewall to verify that no ports are open.

```
### Main fw scan from the outside...

# nmap –P0 -sS -O 198.11.12.1

All 1548 scanned ports on 198.11.12.1 are: filtered

Nmap run completed -- 1 IP address (1 host up) scanned in 2315
seconds
```

In checking the VPN, it's difficult to be sure that there are no holes, but we can certainly verify that there's no access to our known internal IP addresses using ping and nmap. Using nmap from a remote site to verify that we cannot access the SysAdmin IP addresses, we see something like this (this takes a long time!):

```
# nmap -P0 -sS 172.19.100.0/24

Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
All 1086 scanned ports on  (172.19.100.1) are: filtered
All 1086 scanned ports on  (172.19.100.3) are: filtered
...
All 1086 scanned ports on  (172.19.100.255) are: filtered

Nmap run completed -- 16 IP address (256 host up) scanned in 26830
seconds
#
```

From the internal perspective, one improvement that might be made is to restrict access by IP subnet to the main firewall. This is done by adding two entries in the firewall definition as show below Figure 3-2.

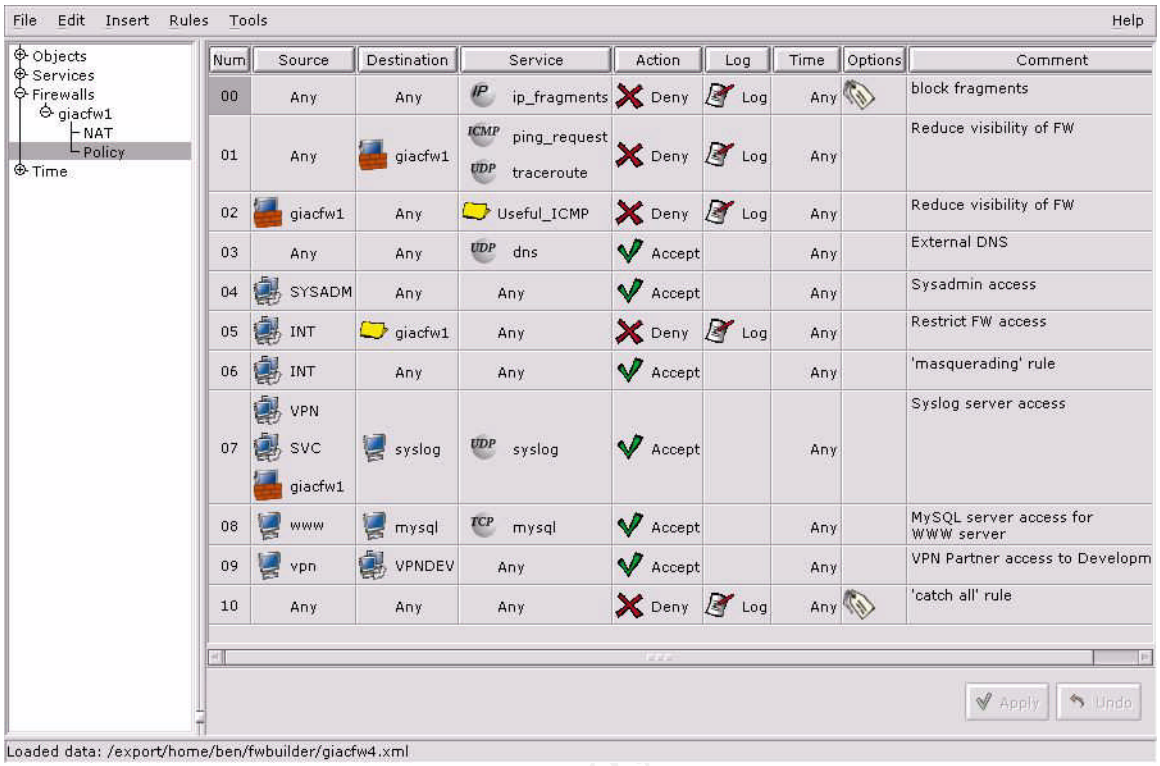


Figure 3-2, Recommended additions to main firewall.

Rule 04, "Sysadmin access" must be inserted first, to allow only those systems in the System Administration address space access to "Any." Next, a specific rule is used to deny all INT (internal) users access to the group "giacfw1" that includes all of the firewall interface IP addresses.

In addition, it was found that the VPN ipf rules could be improved to allow IPSEC connections only from approved sites. There may be conflicts by using the private IP for the "inside" VPN as time goes on, a better solution is to use the IP address space above 198.11.12.128 for private VPN areas. In this way there will be no conflicts with other sites.

4 Design Under Fire

Assignment 4 - Design Under Fire (25 points)

The purpose of this exercise is to help you think about threats to your network and therefore develop a more robust design. Keep in mind that the next certification group will be attacking your architecture!

Select a network design from any previously posted GCFW practical http://www.sans.org/giactc/gcfw.htm) and paste the graphic into your submission. Be certain to list the URL of the practical you are using. Design the following three attacks against the architecture:

1.An attack against the firewall itself. Research and describe at least three vulnerabilities that have been found for the type of firewall chosen for the design. Choose one of the vulnerabilities, design an attack based on the vulnerability, and explain the results of running that attack against the firewall.

2. A denial of service attack. Subject the design to a theoretical attack from 50 compromised cable modem/DSL systems using TCP SYN, UDP, or ICMP floods. Describe the countermeasures that can be put into place to mitigate the attack that you chose.

3.An attack plan to compromise an internal system through the perimeter system. Select a target, explain your reasons for choosing that target, and describe the process to compromise the target.

In designing your attacks, keep the following in mind:

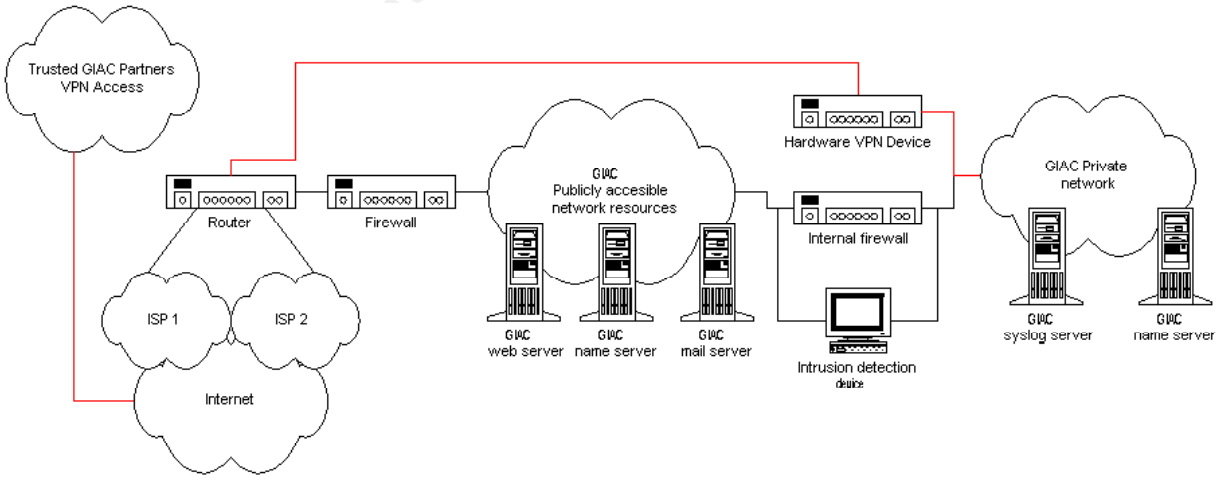
The attack should be realistic. The purpose of this exercise is for the student to clearly demonstrate that they understand that firewall and perimeter systems are not magic "silver bullets" immune to all attacks.

The attack should be reasonable. The firewall does not necessarily have to be impenetrable (perfectly configured with all of the up-to-the-minute patches installed). However, you should not assume that it is an unpatched, out-of-the-box firewall installed on an unpatched out-of-the-box OS. (Remember, you designed GIAC Enterprises' firewall; would you install a system like that?)

You must supply documentation (e.g., a URL to the security bulletin, bugtraq archive, or exploit code used) for any vulnerability you use in your attack.

The attack does not necessarily have to succeed (though a successful attack is often the more interesting approach). If, given the perimeter and network configuration you have described above, the attack would fail, you can describe this result as well.

The network design selected is that of Ken Colson, http://www.sans.org/y2k/practical/ken_colson_gcfw.doc. Here's the Proposed GIAC Network Design from Ken's document:



Before we launch any attacks, we would want to research all available public information about this site. We would start by using the "whois" (http://www.internic.net/cgi-bin/whois) service and the ARIN registry (http://www.arin.net/whois) to check for ownership of the IP addresses. The site's own web site may be a source of useful

information, both names and telephone numbers. A few telephone calls may turn up names that may be useful later on.

4.1 Attack against the Firewall itself.

We can check for useful attack information in a variety of places:

http://www.securityfocus.com for the bugtraq and vunl-dev lists,
http://cve.mitre.org/ for the CVE (Common Vulnerabilities and Exposures) list
http://www.cert.org/ for advisories about the main firewall.

Normally we would approach the firewall not knowing anything about other than what we might have learned in our initial "social engineering." The "nmap" program may be useful in determining what type of system is providing the firewall. First a ping scan could be done on the entire subnet, to determine possible IP addresses. If done over a few days, this wouldn't attract much attention; if we're concerned about that we might use "nmap" to scan commonly accessed ports like 80 and 25 first to map out the subnet. We may get some help here, if the firewall itself is set to REJECT instead of DROP. Of course we have to be careful in case there is an internal IDS system watching not to set off alarms!

Here's a possible list of attacks that we might choose from:

A search of the Security Focus site turns up discussion regarding the "Linux Kernel IP Masquerading Vulnerability" (http://www.securityfocus.com/cgi-bin/archive.pl?id=1&mid=200361), and more details can be found at the Razor team's site: http://razor.bindview.com/publish/advisories/adv_LkIPmasq.html. Unfortunately, since the site under attack was smart enough to turn off FTP access entirely, this probably won't do us much good. However, it is worth exploring since it might be possible that the firewall patch has not been applied just for that reason! We'll save that idea for later.

Another possibility is a rather old problem with IP fragmentation, see (http://www.securityfocus.com/cgi-bin/archive.pl?id=1&mid=19810, and http://www.securityfocus.com/bid/543). By manipulating the IP fragmentation, we can potentially pass traffic through the firewall that is not allowed by the rules. The "fragrouter" tool can be useful in determining whether or not the site is vulnerable to this exploit (http://www.w00w00.org/files/sectools/fragrouter/). Unfortunately for our attack, the site is running a much newer version of the Linux kernel and is not vulnerable.

Finally, there is another IP masquerading attack, from BugTraq: http://www.securityfocus.com/bid/1078, that may bear some fruit. However, closer inspection of this indicates that it is a fairly specialized hole through the firewall, probably not of use here.

4.2 A Denial of Service Attack

One of the classic attacks that we might choose to use is the "smurf" attack [HackingExposed01]. Assuming that we also have a large collection of previously compromised "zombie" sites, we could enlist some of these in randomly starting the attack, in order to make it harder to trace. A list of sites that will act as "smurf" amplifiers can be obtained from http://netscan.org/lamers-r-us.html. This attack is described in detail in http://www.cert.org/advisories/CA-1998-01.html. One source of even more detailed information including how to mitigate such an attack is available at http://www.pentics.net/denial-of-service/white-papers/smurf.cgi.

Another other possibility for attack would be the TCP SYN Flood attack, described in detail by CERT at http://www.cert.org/advisories/CA-1996-21.html. Since many systems now have ways to defend against this type of attack, we move on to the UDP flooding attack.

The UDP flooding attack seems that it might be interesting, especially since we know that the UDP DNS port is open. There is detailed information about this in the CERT advisory, http://www.cert.org/incident_notes/IN-2000-04.html. Even more useful in carrying this out, since we have a collection of compromised hosts, would be the trinoo (or trin00) (http://staff.washington.edu/dittrich/misc/trinoo.analysis) or Tribal Flood Network (http://staff.washington.edu/dittrich/misc/tfn.analysis).

4.3 A plan to compromise an internal system.

The approach to compromising an internal system is to use the same techniques that the "ramen" (http://xforce.iss.net/alerts/advise71.php) or "lion" ("1i0n", http://www.whitehats.com/library/worms/lion/) worms used, to enter through vulnerabilities in the SMTP mail server or DNS (named/bind) server (http://www.cert.org/advisories/CA-2001-02.html). Our approach certainly doesn't need to be as noisy as the worm, we directly attack the DNS server without any kind of preliminary scan that might tip off an IDS. The exploit is available from the http://www.tlsecurity.net/archive/code/linux/ web sight as linx86_bind.c. Once inside the DNS server, we can cover our tracks and setup a "sniffing" operation hoping to get more usernames, passwords, etc. Who knows, we may get lucky and catch an unsuspecting sysadmin logging in via telnet or ftp!

If all else fails, we might decide to use other approaches. In this case we make use of the information obtained previously to target an internal system. We might consider a variety of approaches, depending upon the level of risk we are willing to assume. In addition to the strictly electronic means, we might begin by doing some outside "dumpster diving" to obtain discarded documents. You never know what might turn up there, perhaps even employee information! We might even go so far as to observe the practices of the local

janitorial staff and pose as a janitor during there hours of operation. It's quite common for them to open locked doors of an entire floor; it may be possible to observe user IDs and passwords taped to screens. Another approach might be to pose as a telephone worker and find wiring closets that have network equipment; placing our own wireless node as a sniffer would yield a huge amount of information regarding user IDs and passwords. Another approach might be to craft an email that appears to come from one "The System Administrator" telling everyone about an incredibly dangerous new virus and asking them to immediately apply the attached fix, counting on the fact that out of a fairly large company at least one person might do it! [HackProofing01, p. 44-49] The Trojan would hide itself and "sniff" for internal passwords, IP addresses and other useful information and send it off via an encrypted back channel at random times.

Another approach might be to see how well protected the ISP is, and if possible use one of the two ISPs as an attack platform. Once compromised, the ISP might offer a place where we could capture emails and additional IP addresses, or even launch an attack. The partner sites might be another great place to attack as well, since they may be less well defended.

5 References

Security information:
http://www.sans.org/infosecFAQ/firewall/blocking_cisco.htm
http://www.securitfocus.com
http://www.cert.org
http://www.sans.org
http://razor.bindview.com
http://cve.mitre.org

IPTables/NetFilter information:
http://www.linuxguruz.org/iptables/howto/
http://netfilter.samba.org/
http://www.fwbuilder.org

Other tools and web sites:
Nmap – http://www.insecure.org
Fragrouter – http://www.w00w00.org/files/sectools/fragrouter/

[Brenton01] Brenton, Chris with Hunt, Cameron; Active Defense, A Comprehensive Gude to Network Security, Sybex, 2001.

[Brenton02] Brenton, Chris, Mastering Cisco Routers, Sybex, 2000.

[HackingExposed01] McClure, Stuart, Scambray, Joel, Kurtz, George; Hacking Exposed, Osborne McGraw Hill, 2001.

[HackProof01] Russel, Ryan and Cunningham, Stace, Hack Proofing your Network, Syngress, 2000.

6 Appendix

6.1 NetFilter/IPTables script for main firewall produced by fwbuilder.

```
#!/bin/sh

PATH="/usr/local/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr/bin:/bin"

echo "1" > /proc/sys/net/ipv4/ip_forward

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




if [ -f /proc/net/ip_tables_names ]; then
  set `cat /proc/net/ip_tables_names`
else
  set nat filter
fi

while [ -n "$1" ]; do
    table=$1
    iptables -t $table -L -n  |
    awk '/Chain/ {print $2;}' |
    while read chain; do
      iptables -t $table -F $chain
    done
    iptables -t $table -X
    shift
done



/bin/netstat -rn | /usr/bin/awk '{
    if ($3=="255.255.255.255") { printf "route delete %s\n",$1;}
}' | /bin/sh

arp -an |   /usr/bin/awk '/PERM/ {
    iface=$NF; addr=$2;
    gsub("[()]","",addr);
    printf "arp -i %s -d %s pub\n",iface,addr;
}' | /bin/sh




iptables -A INPUT     -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT    -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
#
#   NAT Rule #0
#
iptables -t nat -A POSTROUTING  -o eth1 -s 172.19.0.0/255.255.0.0 -d 0/0 -j SNAT --to
198.11.12.2
#
#   Interface Rule #0
#
#    Anti-spoofing rule
#
```

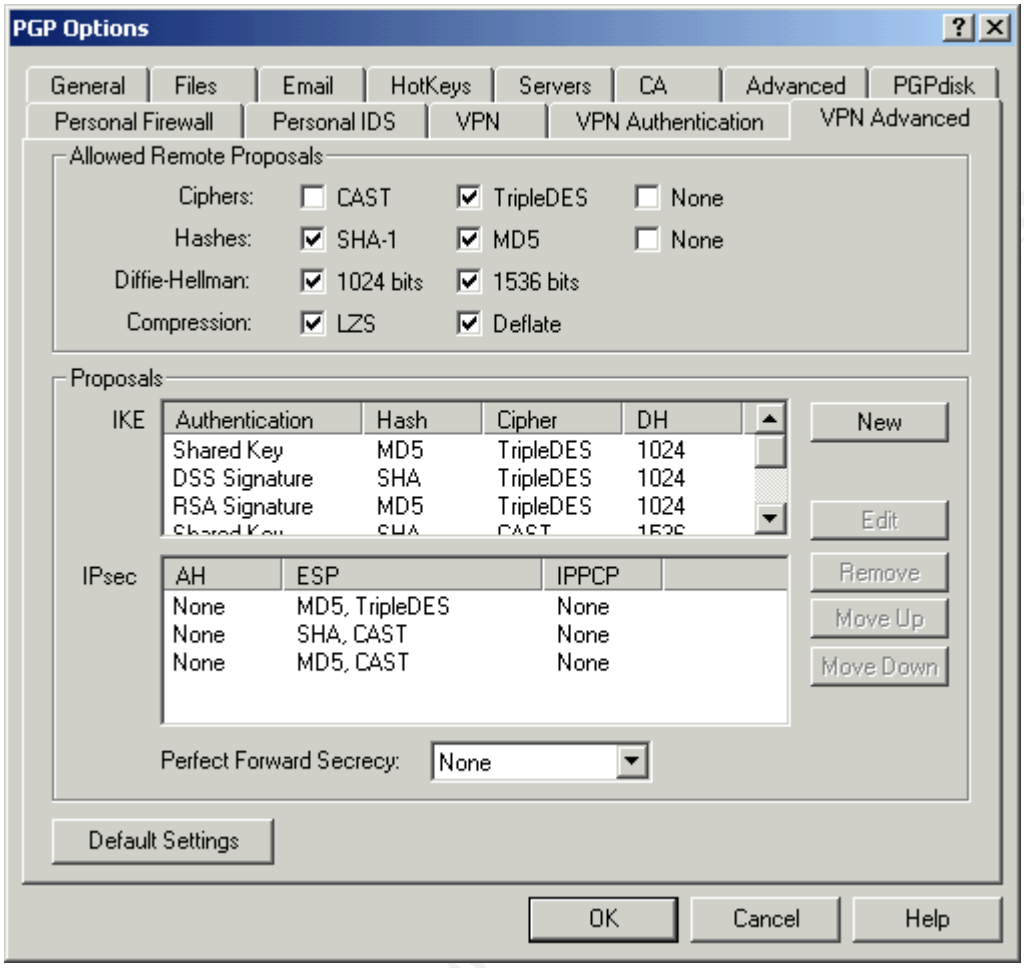
```
iptables -N IRULE_0_eth1
iptables -A INPUT -i eth1  -s 172.19.0.0/255.255.0.0 -j IRULE_0_eth1
iptables -A FORWARD -i eth1  -s 172.19.0.0/255.255.0.0 -j IRULE_0_eth1
iptables -A INPUT -i eth1  -s 192.11.12.16/255.255.255.240 -j IRULE_0_eth1
iptables -A FORWARD -i eth1  -s 192.11.12.16/255.255.255.240 -j IRULE_0_eth1
iptables -A INPUT -i eth1  -s 192.168.3.0/255.255.255.0 -j IRULE_0_eth1
iptables -A FORWARD -i eth1  -s 192.168.3.0/255.255.255.0 -j IRULE_0_eth1
iptables -A INPUT -i eth1  -s 198.11.12.2 -j IRULE_0_eth1
iptables -A INPUT -i eth1  -s 172.0.0.1 -j IRULE_0_eth1
iptables -A INPUT -i eth1  -s 192.168.3.1 -j IRULE_0_eth1
iptables -A INPUT -i eth1  -s 192.11.12.33 -j IRULE_0_eth1
iptables -A INPUT -i eth1  -s 127.0.0.0 -j IRULE_0_eth1
iptables -A FORWARD -i eth1  -s 198.11.12.2 -j IRULE_0_eth1
iptables -A FORWARD -i eth1  -s 172.0.0.1 -j IRULE_0_eth1
iptables -A FORWARD -i eth1  -s 192.168.3.1 -j IRULE_0_eth1
iptables -A FORWARD -i eth1  -s 192.11.12.33 -j IRULE_0_eth1
iptables -A FORWARD -i eth1  -s 127.0.0.0 -j IRULE_0_eth1
iptables -A IRULE_0_eth1  -j LOG   --log-level debug --log-prefix "RULE 0  -- Deny  "
iptables -A IRULE_0_eth1  -j DROP
#
#   Interface Rule #1
#
#    Anti-spoofing rule
#
iptables -N IRULE_1_eth1
iptables -N TI_IRULE_1_eth1
iptables -N TO_IRULE_1_eth1
iptables -N TF_IRULE_1_eth1

iptables -A FORWARD -o eth1  -j TF_IRULE_1_eth1
iptables -A TF_IRULE_1_eth1 -o eth1  -s 172.19.0.0/255.255.0.0 -j RETURN
iptables -A TF_IRULE_1_eth1 -o eth1  -s 192.11.12.16/255.255.255.240 -j RETURN
iptables -A TF_IRULE_1_eth1 -o eth1  -s 192.11.12.4 -j RETURN
iptables -A OUTPUT -o eth1  -j TO_IRULE_1_eth1
iptables -A TO_IRULE_1_eth1 -o eth1  -j RETURN
iptables -A TF_IRULE_1_eth1 -o eth1  -j IRULE_1_eth1
iptables -A TO_IRULE_1_eth1 -o eth1  -j IRULE_1_eth1
iptables -A IRULE_1_eth1  -j LOG   --log-level debug --log-prefix "RULE 1  -- Deny  "
iptables -A IRULE_1_eth1  -j DROP
#
#   Interface Rule #0
#
iptables -N IRULE_0_lo
iptables -A INPUT -i lo  -m state --state NEW  -j IRULE_0_lo
iptables -A IRULE_0_lo  -m state --state NEW  -j ACCEPT
#
#   Interface Rule #1
#
iptables -N IRULE_1_lo
iptables -A OUTPUT -o lo  -m state --state NEW  -j IRULE_1_lo
iptables -A IRULE_1_lo  -m state --state NEW  -j ACCEPT
#
#    Rule #0
#
#    block fragments
#
iptables -N RULE_0
iptables -A OUTPUT  -j RULE_0 -f
iptables -A INPUT  -j RULE_0 -f
iptables -A FORWARD  -j RULE_0 -f
iptables -A RULE_0  -j LOG   --log-level debug --log-prefix "RULE 0  -- Deny  "
iptables -A RULE_0  -j DROP
#
#    Rule #1
#
#    Reduce visibility of FW
#
```

```
iptables -N RULE_1
iptables -A OUTPUT  -p icmp -m state --state NEW  -d 198.11.12.2 --icmp-type 8 -j RULE_1
iptables -A OUTPUT  -p icmp -m state --state NEW  -d 172.0.0.1 --icmp-type 8 -j RULE_1
iptables -A OUTPUT  -p icmp -m state --state NEW  -d 192.168.3.1 --icmp-type 8 -j RULE_1
iptables -A OUTPUT  -p icmp -m state --state NEW  -d 192.11.12.33 --icmp-type 8 -j RULE_1
iptables -A OUTPUT  -p icmp -m state --state NEW  -d 127.0.0.0 --icmp-type 8 -j RULE_1
iptables -A INPUT  -p icmp -m state --state NEW  --icmp-type 8 -j RULE_1
iptables -A OUTPUT  -p udp -m state --state NEW  -d 198.11.12.2 --destination-port
33434:33464 -j RULE_1
iptables -A OUTPUT  -p udp -m state --state NEW  -d 172.0.0.1 --destination-port
33434:33464 -j RULE_1
iptables -A OUTPUT  -p udp -m state --state NEW  -d 192.168.3.1 --destination-port
33434:33464 -j RULE_1
iptables -A OUTPUT  -p udp -m state --state NEW  -d 192.11.12.33 --destination-port
33434:33464 -j RULE_1
iptables -A OUTPUT  -p udp -m state --state NEW  -d 127.0.0.0 --destination-port
33434:33464 -j RULE_1
iptables -A INPUT  -p udp -m state --state NEW  --destination-port 33434:33464 -j RULE_1
iptables -A RULE_1  -m state --state NEW  -j LOG   --log-level debug --log-prefix "RULE 1  --
Deny  "
iptables -A RULE_1  -m state --state NEW  -j DROP
#
#    Rule #2
#
#    Reduce visibility of FW
#
iptables -N RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 198.11.12.2 --icmp-type 11 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 172.0.0.1 --icmp-type 11 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 192.168.3.1 --icmp-type 11 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 192.11.12.33 --icmp-type 11 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 127.0.0.0 --icmp-type 11 -j RULE_2
iptables -A OUTPUT  -p icmp -m state --state NEW  --icmp-type 11 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 198.11.12.2 --icmp-type 0 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 172.0.0.1 --icmp-type 0 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 192.168.3.1 --icmp-type 0 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 192.11.12.33 --icmp-type 0 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 127.0.0.0 --icmp-type 0 -j RULE_2
iptables -A OUTPUT  -p icmp -m state --state NEW  --icmp-type 0 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 198.11.12.2 --icmp-type 3 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 172.0.0.1 --icmp-type 3 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 192.168.3.1 --icmp-type 3 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 192.11.12.33 --icmp-type 3 -j RULE_2
iptables -A INPUT  -p icmp -m state --state NEW  -s 127.0.0.0 --icmp-type 3 -j RULE_2
iptables -A OUTPUT  -p icmp -m state --state NEW  --icmp-type 3 -j RULE_2
iptables -A RULE_2  -m state --state NEW  -j LOG   --log-level debug --log-prefix "RULE 2  --
Deny  "
iptables -A RULE_2  -m state --state NEW  -j DROP
#
#    Rule #3
#
#    External DNS
#
iptables -N RULE_3
iptables -A OUTPUT  -p udp -m state --state NEW  --destination-port 53 -j RULE_3
iptables -A INPUT  -p udp -m state --state NEW  --destination-port 53 -j RULE_3
iptables -A FORWARD  -p udp -m state --state NEW  --destination-port 53 -j RULE_3
iptables -A RULE_3  -m state --state NEW  -j ACCEPT
#
#    Rule #4
#
#    'masquerading' rule
#
iptables -N RULE_4
iptables -A INPUT  -m state --state NEW  -s 172.19.0.0/255.255.0.0 -j RULE_4
iptables -A FORWARD  -m state --state NEW  -s 172.19.0.0/255.255.0.0 -j RULE_4
iptables -A RULE_4  -m state --state NEW  -j ACCEPT
```

```
#
#    Rule #5
#
#    Syslog server access
#
iptables -N RULE_5
iptables -A FORWARD  -p udp -m state --state NEW  -s 192.168.3.0/255.255.255.0 -d
172.19.100.22 --destination-port 514 -j RULE_5
iptables -A FORWARD  -p udp -m state --state NEW  -s 192.11.12.16/255.255.255.240 -d
172.19.100.22 --destination-port 514 -j RULE_5
iptables -A OUTPUT  -p udp -m state --state NEW  -d 172.19.100.22 --destination-port 514 -
j RULE_5
iptables -A RULE_5  -m state --state NEW  -j ACCEPT
#
#    Rule #6
#
#    MySQL server access for WWW server
#
iptables -N RULE_6
iptables -A FORWARD  -p tcp -m state --state NEW  -s 192.11.12.18 -d 172.19.10.24 --
destination-port 3306 -j RULE_6
iptables -A RULE_6  -m state --state NEW  -j ACCEPT
#
#    Rule #7
#
#    VPN Partner access to Development
#
iptables -N RULE_7
iptables -A FORWARD  -m state --state NEW  -s 192.11.12.4 -d 172.19.0.0/255.255.0.0 -j
RULE_7
iptables -A RULE_7  -m state --state NEW  -j ACCEPT
#
#    Rule #8
#
#    'catch all' rule
#
iptables -N RULE_8
iptables -A OUTPUT  -j RULE_8
iptables -A INPUT  -j RULE_8
iptables -A FORWARD  -j RULE_8
iptables -A RULE_8  -j LOG   --log-level debug --log-prefix "RULE 8  -- Deny  "
iptables -A RULE_8  -j DROP
#
#  Final rules
#
iptables -A INPUT      -j DROP
iptables -A OUTPUT     -j DROP
iptables -A FORWARD    -j DROP
```

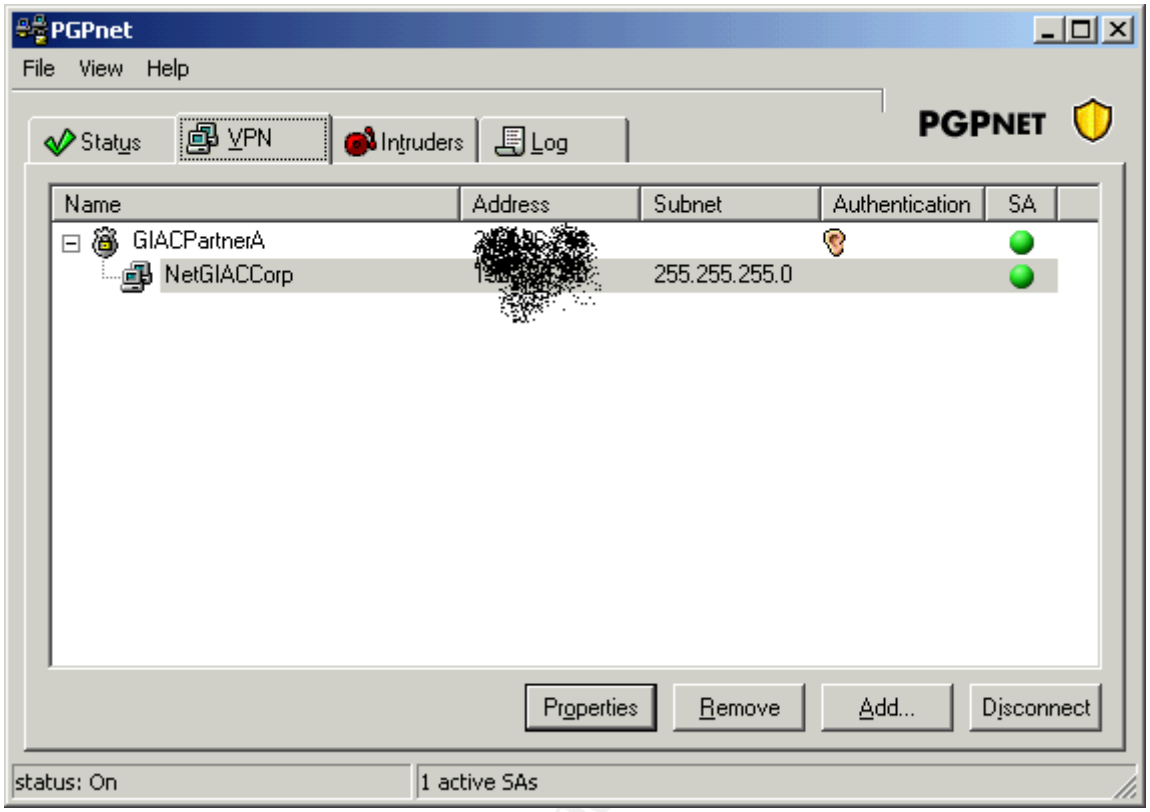
6.2 VPN definitions for PGP security.

First theVPN options are setup for our purposes:



We will not be using the CAST encryption algorithm, so it is unchecked and proposals that contain CAST are moved down. This leaves us with the IKE Shared Key using MD5 hash, 3DES encryption and 1024 bit Diffie-Hellman for our key exchange in this instance. The IPsec tunnel will be encrypted with MD5 hash and 3DES encryption.

When the tunnel is brought up, it looks something like this:



6.3 OpenBSD configuration for GIAC Corporate VPN.

```
KeyNote-Version: 2
Comment: This policy is for our PGP connections
Authorizer: "POLICY"
Licensees: "passphrase:WhyNotACookieToday"
Conditions: app_domain == "IPsec policy" &&
            esp_enc_alg == "3des" &&
            esp_present == "yes" -> "true" ;

KeyNote-Version: 2
Comment: From our partner site, GIACPartnerA
Authorizer: "POLICY"
Licensees: "passphrase:TheOtherCookieCrumbler"
Conditions: app_domain == "IPsec policy" &&
            esp_enc_alg == "aes" &&
            esp_present == "yes" -> "true" ;
```

Figure 6-1, VPN Policy definition for PGP desktop

The policy file for GIAC Corporate site defines two policies, one for the PGP connections and the other for the GIACPartnerA site. The details of the connections are shown in the isakmp.conf file.

```
#
# Adapted from www.allard.nu/openbsd and the www.openbsd.org
# documentation and sample files to allow a connection from PGP clients
# and also between GIACCorp and GIACPartnerA
#

[Phase 1]
Default=              ISAKMP-clients-PGP
192.168.10.13= ISAKMP-peer-GIACPartnerA

[Phase 2]
Passive-Connections=  IPsec-clients-PGP
Connections=          IPsec-GIACCorp-GIACPartnerA


# Phase 1 peer sections
########################

[ISAKMP-clients-PGP]
Phase=                1
Transport=            udp
Configuration=        PGP-main-mode
Authentication=       WhyNotACookieToday

# Phase 2 sections
##################

[IPsec-clients-PGP]
Phase=                2
Configuration=        PGP-quick-mode
Local-ID=             PGP-default-route
Remote-ID=            PGP-dummy-remote

# Client ID sections
####################

[PGP-default-route]
ID-type=       IPV4_ADDR_SUBNET
Network=       0.0.0.0
Netmask=       0.0.0.0

[PGP-dummy-remote]
ID-type=       IPV4_ADDR
Address=       0.0.0.0

[ISAKMP-peer-GIACPartnerA]
Phase=                1
Transport=            udp
Local-address=        192.168.3.3
Address=              192.168.10.13
Configuration=        Default-main-mode
Authentication=       TheOtherCookieCrumbler

[IPsec-GIACCorp-GIACPartnerA]
Phase=                2
ISAKMP-peer=          ISAKMP-peer-GIACPartnerA
Configuration=        Default-quick-mode
Local-ID=             Net-GIACCorp
Remote-ID=            Net-GIACPartnerA

[Host-GIACCorp]
ID-type=              IPV4_ADDR
Address=              192.168.11.1
```

```
[Host-GIACPartnerA]
ID-type=                IPV4_ADDR
Address=                192.168.12.1

[Net-GIACCorp]
ID-type=                IPV4_ADDR_SUBNET
Network=                192.168.11.0
Netmask=                255.255.255.0

[Net-GIACPartnerA]
ID-type=                IPV4_ADDR_SUBNET
Network=                192.168.12.0
Netmask=                255.255.255.0


[PGP-main-mode]
DOI=                    IPSEC
EXCHANGE_TYPE=          ID_PROT
Transforms=             3DES-MD5

[PGP-quick-mode]
DOI=                    IPSEC
EXCHANGE_TYPE=          QUICK_MODE
Suites=                 QM-ESP-3DES-MD5-SUITE

# PGP main mode

[3DES-MD5]
ENCRYPTION_ALGORITHM=   3DES_CBC
HASH_ALGORITHM=         MD5
AUTHENTICATION_METHOD=  PRE_SHARED
GROUP_DESCRIPTION=      MODP_1024
Life=                   LIFE_1_DAY

# From OpenBSD examples...
############
[Default-main-mode]
DOI=                    IPSEC
EXCHANGE_TYPE=          ID_PROT
Transforms=             3DES-SHA

[Default-quick-mode]
DOI=                    IPSEC
EXCHANGE_TYPE=          QUICK_MODE
Suites=                 QM-ESP-AES-SHA-PFS-SUITE

[LIFE_1_DAY]
LIFE_TYPE=              SECONDS
LIFE_DURATION=          86400,79200:93600
```


6.4 GIACPartnerA configuration files

First the policy file:

```
KeyNote-Version: 2
Comment: This policy accepts ESP SAs from a remote that uses the
right password
      $OpenBSD: policy,v 1.5 2000/10/09 23:27:29 niklas Exp $
      $EOM: policy,v 1.6 2000/10/09 22:08:30 angelos Exp $
Authorizer: "POLICY"
Licensees: "passphrase:Benzmekmitasdigoatxyzzy"
Conditions: app_domain == "IPsec policy" &&
          esp_enc_alg == "aes" &&
          esp_present == "yes" -> "true" ;
```

And finally the isakmpd.conf file for GIACPartnerA:

```
[General]
Retransmits=            5
Exchange-max-time=      120
Listen-on=              192.168.10.13

[Phase 1]
192.168.2.11=           ISAKMP-peer-GIACCorp

[Phase 2]
Connections=            IPsec-GIACPartnerA-GIACCorp

[ISAKMP-peer-GIACCorp]
Phase=                  1
Transport=        udp
Local-address=          192.168.3.3
Address=          192.168.10.13
Configuration=          Default-main-mode
Authentication=         Benzothermekmitasdigoatxyzzy

[IPsec-GIACPartnerA-GIACCorp]
Phase=                  2
ISAKMP-peer=            ISAKMP-peer-GIACCorp
Configuration=          Default-quick-mode
Local-ID=         Net-GIACPartnerA
Remote-ID=        Net-GIACCorp

[Host-GIACCorp]
ID-type=          IPV4_ADDR
Address=          192.168.11.1

[Host-GIACPartnerA]
ID-type=          IPV4_ADDR
Address=          192.168.12.1

[Net-GIACCorp]
ID-type=          IPV4_ADDR_SUBNET
Network=          192.168.11.0
Netmask=          255.255.255.0

[Net-GIACPartnerA]
ID-type=          IPV4_ADDR_SUBNET
```

```
Network=          192.168.12.0
Netmask=          255.255.255.0

[Default-main-mode]
DOI=              IPSEC
EXCHANGE_TYPE=    ID_PROT
Transforms=       3DES-SHA

[Default-quick-mode]
DOI=              IPSEC
EXCHANGE_TYPE=    QUICK_MODE
Suites=           QM-ESP-AES-SHA-PFS-SUITE
```