



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

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

GIAC GCFW Practical
Assignment Version 1.9 Rewrite 1

Open Source Firewalls for Security at GIAC Enterprises

Thomas A. Kyle

© SANS Institute 2003, Author retains full rights

Table of Contents

Abstract.....	4
Network Security Architecture.....	5
Goals.....	5
Security Principles.....	5
Access Requirements	5
Customers	5
Suppliers.....	6
Partners	6
On-site employees.....	6
Telecommuters and mobile sales force	7
Ecommerce system	8
Mail relay	8
Design Architecture.....	9
IP addressing scheme	9
Split DNS	10
Network Segmentation	11
Network Security Components	12
Filtering router.....	12
Firewall	12
VPN Concentrator.....	13
Snort IDS	14
Central Logging Server	16
Physical Network Connections	17
On-Site Employee Dataflow.....	18
Customer Order Dataflow	19
Security Policy	21
Border Router.....	21
Ingress/Egress Filtering.....	21
SSH Management	24
Controlling access to the router with access lists.....	25
Firewall.....	27
Options	28
Traffic normalization	29
NAT and redirection.....	29
Filter.....	30
VPN.....	35
Client Addressing.....	35
IPSec Configuration.....	37
Configuration of OpenBSD firewall tutorial.....	42
Configuration file	42
Options	42

Packet Normalization.....	42
NAT and Redirection	43
Rules	44
Firewall Verification.....	50
Plan.....	50
Information Gathering.....	50
Architecture Review.....	51
Network Scanning.....	51
DNS resolution.....	52
Results	53
Scans of Firewall	53
Screened Subnet scans.....	54
Email Connectivity	56
HTTP and HTTPS to Corporate Web Services.....	57
SSH from Intranet.....	57
Evaluation of audit.....	58
Design Under Fire.....	61
Firewall Attack.....	62
Attack against firewall.....	62
Denial-of-Service	64
Internal Compromise Through Perimeter.....	66
Appendix A: OpenBSD installation and hardening.....	70
Hardening.....	76
Disabling unnecessary daemons.....	77
Securing necessary services	77
Configuring logging	78
Keeping correct time.....	78
Logging packets.....	78
Networking	79
Configuring interfaces.....	79
Default Route.....	79
Enabling Firewalling.....	79
Administrivia.....	80
Adding users.....	80
Applying Patches.....	80
Appendix B: Network and System Information.....	82
References	83

Abstract

GIAC Enterprises is a leading vendor of fortune cookie sayings. Today, through its international partnerships, GIAC Enterprises sells fortune sayings in 20 countries and 16 languages.

Recently, a major competitor of GIAC Enterprises was forced into bankruptcy after inexplicably shipping thousands of cookies with offensive fortunes to elementary schools. It was soon discovered that an unpatched public webserver had been compromised, by which the fortunes were modified before distribution. Wishing to avoid such a predicament, GIAC Enterprises management ordered a redesign of the company's perimeter network security.

Currently, IT security at GIAC Enterprises is limited to router access lists and up-to-date patching of its core servers. The network infrastructure of GIAC Enterprises consists of Cisco routers and switches. GIAC Enterprises is connected to the Internet through a DS3 to a major backbone vendor, who terminates the connection in 100Mbps Ethernet. Most services, including web and database, are located on open source Unix hosts, while mail services are provided through a Microsoft Exchange 2000 server.

GIAC Enterprises IT staff have indicated a strong desire to leverage their existing Cisco and Unix hardware and knowledge, and also are extremely comfortable working with open source software. Since the fortune cookie market has plummeted since the dot-com bubble burst (it is a little-known fact that Silicon Valley startups often provided employees with large quantities of free fortune cookies), GIAC Enterprises management has emphasized the need for a cost-effective solution. As such, this design will seek to emphasize these strengths to increase security while minimizing costs.

This paper will describe the network security architecture, the components used, and the actual security policy for the GIAC Enterprises network. Additionally, a verification of this design, as well as an attack on another design, are included as well.

Network Security Architecture

Goals

Security is a process that works towards three goals: privacy, risk mitigation, and logging. The first goal, privacy, involves hiding information about an organization and its resources, including information, systems, and services. Second, risk mitigation involves shielding and protecting the information, systems and services. Any of these that are accessible or public are vulnerable to attack from the outside. Finally, logging involves an awareness of what is happening on the network, including exceptional events, attempted intrusions from the outside, and unexpected behavior of internal systems.

Security Principles

With these goals in mind, the principles behind the redesign of the GIAC Enterprises network are defense-in-depth, segregated services, and least privilege. As its name implies, defense-in-depth is a multilayered approach, which affords a greater level of security through its layers. Each security component acts as a layer, with overlapping responsibilities. Should one component be misconfigured or otherwise fail, another layer can prevent an intrusion from occurring. By segregating services, or having single-purpose hosts, a single incident or intrusion is less likely to affect multiple services. Finally, the principle of least privilege is restricting access to resources on an as-required basis. Where there is no business need, no access is permitted.

Access Requirements

Customers

Customers of GIAC Enterprises interact with the company online, through its website. From the website, fortunes may be purchased through a secure server, and customers may also interact with the company, primarily through email forms available on the website. Purchased fortunes are “delivered” to customers through a download section on the secure webserver.

Customers will therefore need HTTP and HTTPS access to the public webserver, be able to query the public external, and be able to send and receive email messages.

protocol	port	destination
HTTP	80/tcp	public webserver
HTTPS	443/tcp	public webserver
DNS	53/udp	external DNS
SMTP	25/smtp	mail relay

Suppliers

GIAC Enterprises has contracted development of some fortunes to other businesses. These companies submit their fortunes through the secure portion of the corporate website. For various business operations and technical support, suppliers may contact GIAC Enterprises units via email.

Suppliers will therefore HTTP and HTTPS access to the public web servers, be able to query the external DNS, and be able to send and receive email messages.

protocol	port	destination
HTTP	80/tcp	public web servers
HTTPS	443/tcp	public web servers
DNS	53/udp	external DNS
SMTP	25/smtp	mail relay

Partners

In order to support its global operations, GIAC Enterprises partners with several companies that translate its fortunes into other languages. Due to the widely varying levels of IT sophistication at each of these companies, GIAC Enterprises has chosen to provide partners with a secure web application, which has a modern web browser as its only technical requirement. This web application has read and write access to various portions of the fortunes database.

protocol	port	destination
HTTP	80/tcp	public web servers
HTTPS	443/tcp	public web servers
DNS	53/udp	external DNS
SMTP	25/smtp	mail relay

On-site employees

On-site employees, who are located on the internal, corporate network, will require various levels of access, dependent upon function.

All employees will need access to the World Wide Web, which is provided through a Squid web proxy on the internal network. The Squid proxy also provides FTP functionality. Email is provided through the corporate Exchange server. All DNS queries from the internal network will be routed through the internal DNS.

Since GIAC Enterprises is in the business of creating fortunes, all on-site employees will need to be able to submit new fortunes and review previously submitted fortunes. An application server that works with the database directly will handle interaction with the fortunes database.

IT workers, particularly network and systems managers, will require administrative access to hosts in screened subnets. All administrative access will be via encrypted protocols, such as SSH. For protocols that do not natively support encryption, an encrypted tunnel, such as stunnel or SSH, will be used.

Other than VPN access to the Exchange server, no access to the internal network from outside the corporate LAN is needed. Therefore, no traffic initiated from the Internet or any screened subnet is allowed through the firewall.

protocol	port	destination
HTTP	80/tcp	via squid proxy
HTTPS	443/tcp	via squid proxy
HTTPS	443/tcp	application server
SSH	22/tcp	various screened subnets

Telecommuters and mobile sales force

GIAC Enterprises operates a large-scale mobile sales force, as well as having a small number of telecommuters. These mobile users communicate with the main corporate office through the corporate Exchange server. Since it is imperative to protect these internal communications, off-site workers will connect to the corporate intranet through VPN in order to synchronize their Outlook mail and calendar. No other access is required for these users.

protocol	port	destination
IPSec	500/isakmp	VPN Concentrator
IPSec/UDP	10000/udp	VPN Concentrator

Since Microsoft Exchange 2000 typically communicates on many, random ports above 1024, it must be configured to use statically-defined ports. Microsoft details how to modify the registry in the support pages of their website¹ in order to accomplish this. Additionally, the VPN Concentrator will need to authenticate users via the RADIUS protocol to a domain controller.

¹ <http://support.microsoft.com/?kbid=280132>

protocol	port	destination
DNS	53/tcp	Exchange
DNS	53/udp	Exchange
Kerberos	88/tcp	Exchange
NTP	123/tcp	Exchange
EndPointMapper	135/tcp	Exchange
LDAP	389/tcp	Exchange
LDAP	389/udp	Exchange
SMB	445/tcp	Exchange
Active Directory	1025/tcp	Exchange
LDAP	3268/tcp	Exchange

Ecommerce system

The ecommerce application needs to access an outside system for payment authorization. GIAC Enterprises uses a service named MonkeyPay. MonkeyPay authorizes credit card purchases over an HTTPS session to verify.monkeypay.com.

protocol	port	destination
HTTPS	443/tcp	payment authorization service

Mail relay

To protect the corporate Exchange server, all inbound and outbound SMTP traffic will be sent through a relay located on a screened subnet. This will prevent any malicious outsiders from communicating directly with the Exchange server. Additionally, the mail relay may be used as a choke point. The Unix staff at GIAC Enterprises has expressed interest in the possibility of filtering unwanted, spam email with pf and blacklists, a so-called “tar pit” currently being developed by the OpenBSD founder, Theo DeRaadt, and developer Daniel Hartmeier².

The system itself is an OpenBSD 3.2 system running Sendmail 8.12.6, which is included in the base install. It is configured as a normal mail relay host. As such, it will need to communicate with the Exchange server using normal SMTP.

² “Annoying spammers with pf and spamd.” <http://www.benzedrine.cx/relaydb.html>

protocol	port	destination
SMTP	25/tcp	Outside mail servers
SMTP	25/tcp	Internal Exchange 2000 Server

Design Architecture

Careful consideration was taken to bring inherent security into the architectural design of the GIAC Enterprises network. Design elements such as the use of internal addressing, split DNS, and segmentation add additional protection to the overall network.

IP addressing scheme

A private addressing scheme, using addresses outlined in RFC 1918³, will be used for all hosts at GIAC Enterprises. The firewall will provide the necessary address translation dynamically.

Using such a schema will increase security of internal GIAC Enterprises hosts, as there will not be a direct mapping between an external, globally-known address and an internal one. Therefore, a malicious outsider will not be able to target a specific host on the internal network from the outside, as there will be no direct path to the inside. In order to attack an internal host, an intruder would have to compromise a hardened system on the DMZ, or the firewall itself.

segment	internal address	global address	description
Internet	-	192.0.2.0/28	Outside of firewall
Backend	10.30.20.0/24	-	Backend servers, such as servlet engine, application server
Webservers	10.30.10.0/24	192.0.2.1, 192.0.2.2	Public webservers
Mail Relay	10.20.10.0/24	192.0.2.3	SMTP Relay
External DNS	10.20.0.0/24	192.0.2.4	Contains external DNS servers
VPN	10.40.0.0	192.0.2.5	"Private" network on VPN Concentrator
Syslog	10.10.9.1	192.0.2.11	Centralized syslog server
NTP	10.10.9.7	192.0.2.12	NTP time server for clock synch

³ "Address Allocation for Private Internets." <http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1918.html>

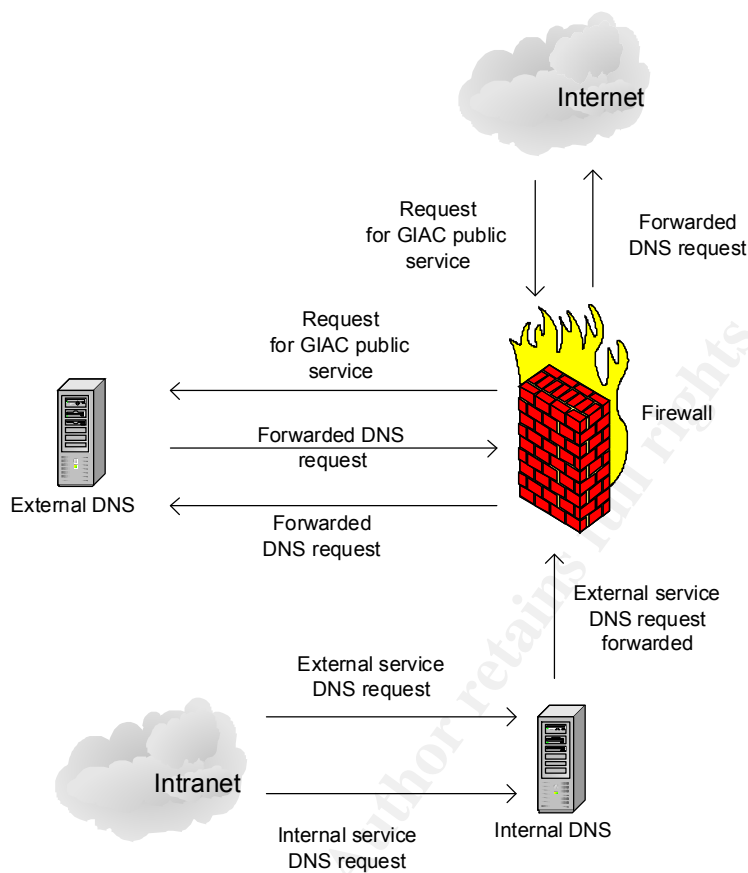
Border router	-	192.0.2.13	Outside of firewall
ISP router	-	192.0.2.14	ISP router
Database	10.30.0.0/24	-	Fortunes database servers
Internal	10.10.0.0/16	-	Corporate LAN

Split DNS

In order to prevent information about the internal layout of the GIAC Enterprises network from being accessible from the Internet, we have chosen to implement split DNS. In this layout, two sets of DNS servers, one external and the other internal, will be used.

The external DNS will exist on a screened subnet, and will be able to answer queries from the outside world. However, the external DNS will only have knowledge about GIAC Enterprises' publicly accessible services, such as the public webserver. The system will also not be allowed to initiate any connections (other than syslog). Should this system be compromised, whether by misconfiguration or even by complete system subversion ('rooting'), the attacker will not be able to gain access into or knowledge about the GIAC Enterprises internal network.

The internal DNS will service the internal GIAC Enterprises network. DNS records about the inside network will be kept here. The internal DNS will also forward requests to the external DNS for information about remote domains.



DNS duties at GIAC Enterprises will be handled by OpenBSD 3.2 servers running the Internet Software Consortium's BIND. Although an audited version of the BIND 4.9 is included with the base install of OpenBSD 3.2, we have elected to utilize the newer ISC BIND product, 9.2.1, as the DNS server. GIAC Enterprises currently uses and is comfortable with BIND 9; we feel that there is no significant reason to migrate the organization to another system, such as djbdns. Additionally, it has been announced that in OpenBSD 3.3, a version of BIND 9 will replace BIND 4.9 in the base system⁴.

For the sake of reliability, we will have four total DNS servers: two external and two internal. Should any one server fail, GIAC Enterprises customers, partners, and employees will still have functionality.

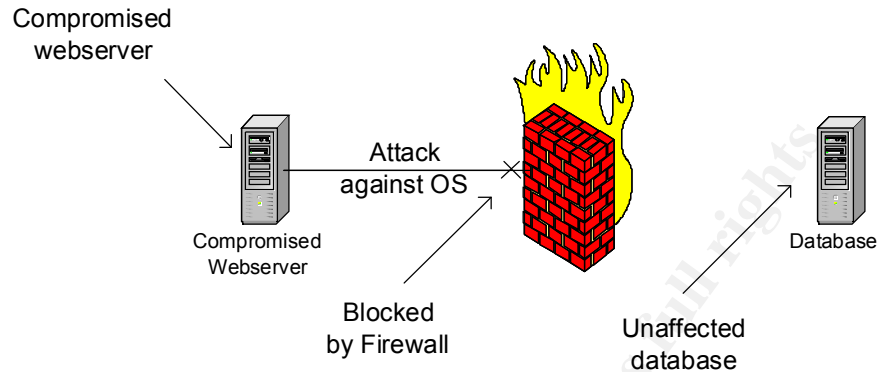
Network Segmentation

Through the use of segmentation, the network is modularized. Hosts and services on each segment are insulated from hosts and services on other segments. Access is more easily controlled, as all traffic between segments must pass through the firewall, which acts as a choke point.

One major benefit of a segmented network is that should a system on one segment be compromised, it is much more difficult for an attacker to use that

⁴ "Bind 9 in -current" <http://www.deadly.org/article.php3?sid=20030121022208&mode=flat>

system to compromise hosts on other segments. Hosts that provide similar services, such as web servers, are separated from hosts that provide other services, such as databases. In this example, this limits the ability of an attacker to use a compromised webserver to attack a database on another segment:



As helpful as this design is, it does have its drawbacks. One potential drawback to a segmented network is that it often requires a large number of firewall interfaces. For proprietary firewalls, there may be a substantial cost involved in adding additional interfaces. Also, since more traffic must traverse the firewall, care must be taken in ensuring the firewall is capable of handling the traffic load, including packets and state table entries. Finally, and most seriously, since all traffic between segments traverses the firewall, which in this case is a general-purpose BSD system, should the firewall be compromised, an attacker would potentially be able to view all traffic between segments.

Network Security Components

Filtering router

A Cisco 3550-24-EMI multilayer switch, running IOS 12.1.12c.EA1, have been chosen as GIAC Enterprises' filtering router. Cisco has proven products and an excellent post-sales support structure, and coupled with GIAC Enterprises' familiarity with Cisco products and the availability of knowledgeable administrators, was an easy choice.

The 3550-series switch was chosen for its impressive performance and low cost. Cisco claims 6.6Mpps, which should be easily sufficient for handling the DS3 connection. Additionally, the 3550-24-EMI is a fairly inexpensive switch.

Since the 3550 is a 24-port multilayer switch, SPAN (switch port analyzer) can be used to "sniff" traffic bound to other ports. This allows plugging an IDS directly into the border router.

Firewall

Since the management of GIAC Enterprises has low cost as a requirement, and the IT staff prefers open source solutions, we have found it an

easy decision to recommend the use of an OpenBSD 3.2 system as the primary GIAC Enterprises.

OpenBSD is a freely-available, BSD-licensed operating system that specializes in security. It has a proven security track record, and the development team proudly states on its homepage that there has been “only one remote hole in the default install, in more than 7 years”⁵. Stateful firewalling is built into the kernel using the “pf” module, and OpenBSD can be remotely administered securely using OpenSSH.

Additionally, since the kernel and userland source are co-developed by the same development team (unlike GNU/Linux), we find it relatively less complicated to keep OpenBSD systems updated.

OpenBSD 3.2 runs on many different hardware platforms, from i386 to PowerMac to UltraSparc, however we recommend using off-the-shelf, commodity Intel hardware for greatest value.

Our basic requirements (beyond the bare necessities of running a computer) are:

- 1GHz AMD or Intel processor
- 256MB RAM
- 4GB or larger IDE hard drive
- CDROM (IDE)
- Two quad-port Intel EtherExpress PRO/100 NICs
- Quality, brand-name motherboard
- Dual “approved” power-supply

Supported hardware list at <http://www.openbsd.org/i386.html>

Most of these components are off-the-shelf and fairly inexpensive commodities. GIAC Enterprises typically uses a local company that will build PCs to their specifications, and we see no problem with this.

Daniel Hartmeier, an OpenBSD developer, in a presentation to Usenix 2002, has shown that an OpenBSD firewall with used 64MB RAM for 64,000 state entries, and that the cost increased linearly⁶. From this information, we may conservatively estimate our firewall of handling 192,000 state entries. This compares similarly to a PIX 515E, which, according to Cisco, can handle 125,000 connections⁷.

VPN Concentrator

A Cisco 3015 VPN Concentrator has been chosen to handle VPN duties for GIAC Enterprises. The Concentrator supports up to 100 concurrent IPSec sessions, with 4Mbps throughput, and is upgradeable for additional

⁵ <http://www.openbsd.org>

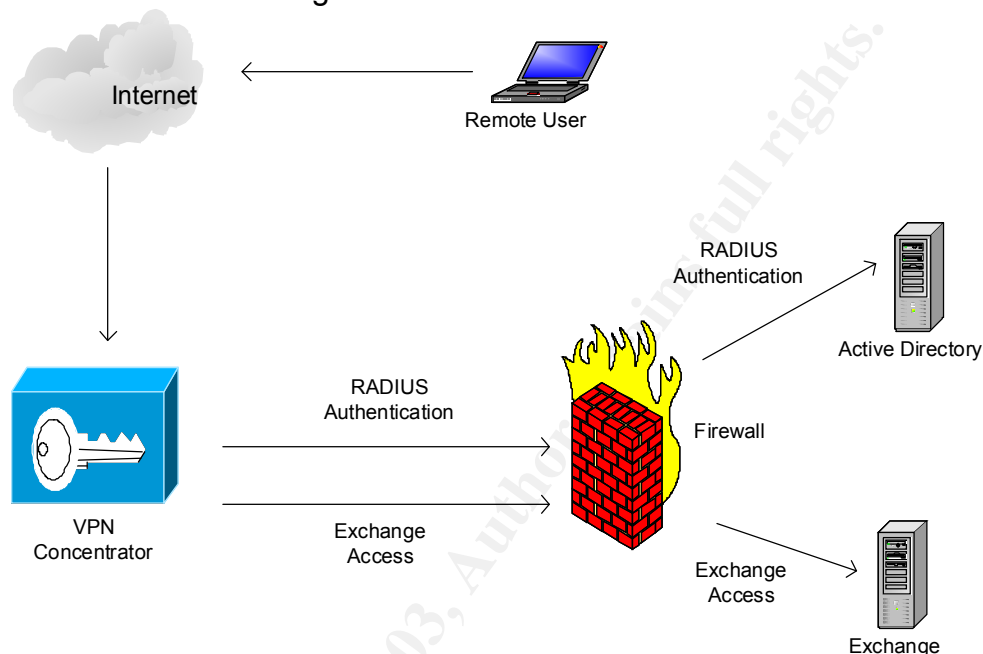
⁶ “Design and Performance of the OpenBSD Stateful Packet Filter” <http://www.benzedrine.cx/pf-slides.pdf>

⁷ <http://www.cisco.com/warp/public/cc/pd/fw/sqfw500/index.shtml>

performance⁸. The VPN Concentrator also supports L2TP and PPTP VPNs. Additionally, Cisco includes their Cisco VPN Client.

We will be using the 3015 to provide IPSec connectivity for GIAC Enterprise's mobile salesforce, as well as its growing numbers of telecommuters. Authentication will be handled via RADIUS to GIAC Enterprises' Active Directory.

Currently, these workers only need access to the corporate Exchange server for email and calendaring, however should that need grow in the future, the 3015 should be able to grow with it.



Snort IDS

Although its configuration and deployment are beyond the scope of this document, perimeter security is not complete without the deployment of an Intrusion Detection Sensor. Several IDS sensors will be placed on the GIAC Enterprises network, one for each firewall segment. The sensors will not actually reside on the segments; they will passively “sniff” the traffic on each network. The sensors will then log any alerts to a central MySQL database, where the results will be analyzed by Roman Danyliw's Analysis Console for Intrusion Databases (ACID)⁹.

We have chosen to use Snort, the open source IDS, version 1.9.2. Snort is extremely flexible and well-supported, its ruleset is easily extensible, and its price is hard to beat.

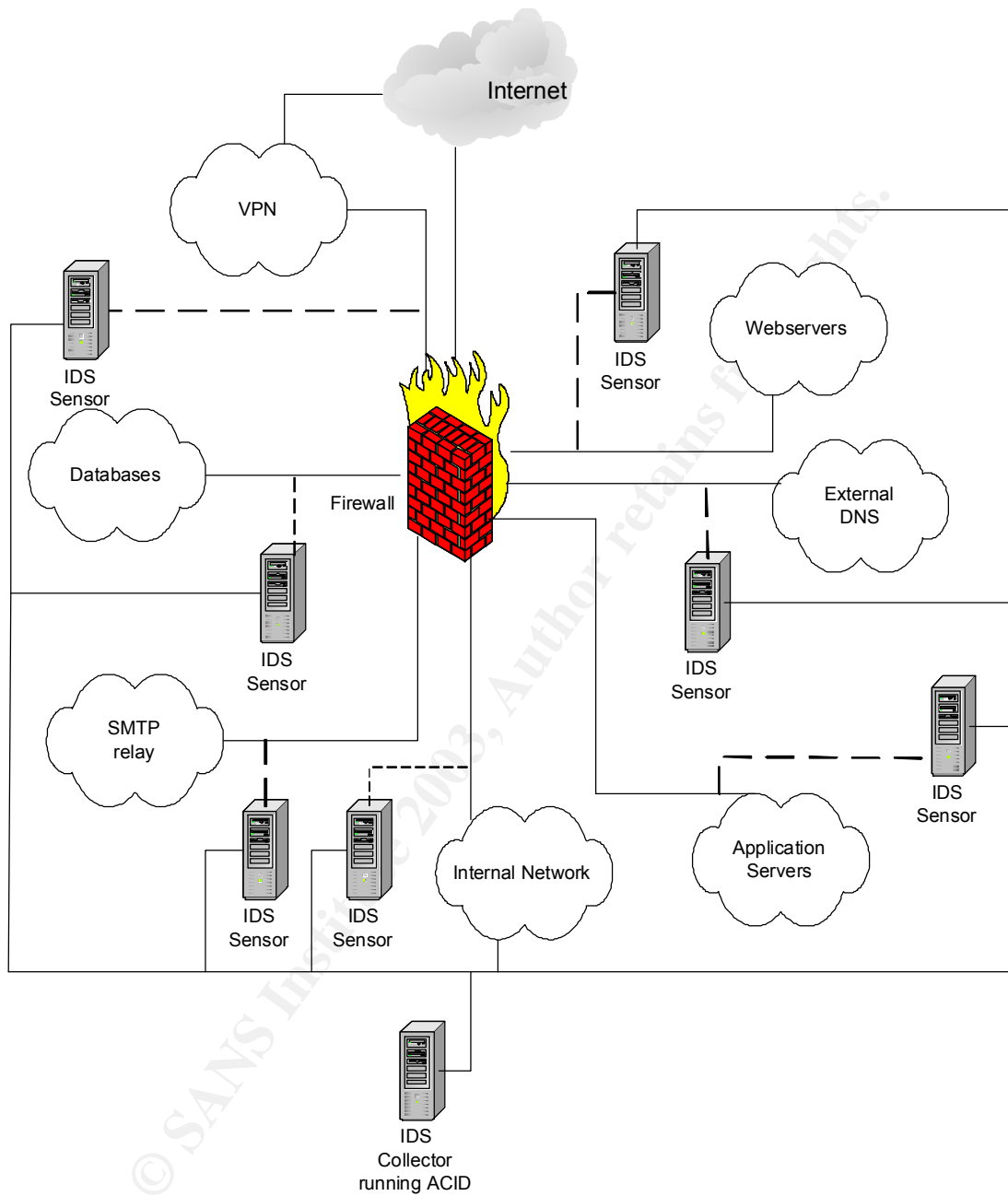
In the following diagram, the solid lines indicate network connections, while the dashed lines indicate “silent” connections the IDS sensors use for

⁸

http://www.cisco.com/en/US/products/hw/vpndevc/ps2284/products_data_sheet09186a0080091e4f.html

⁹ <http://www.andrew.cmu.edu/~rdanyliw/snort/snortacid.html>

packet sniffing. Note that the IDS sensors are all connected to the internal network, and that they log back to a central IDS collector:



Central Logging Server

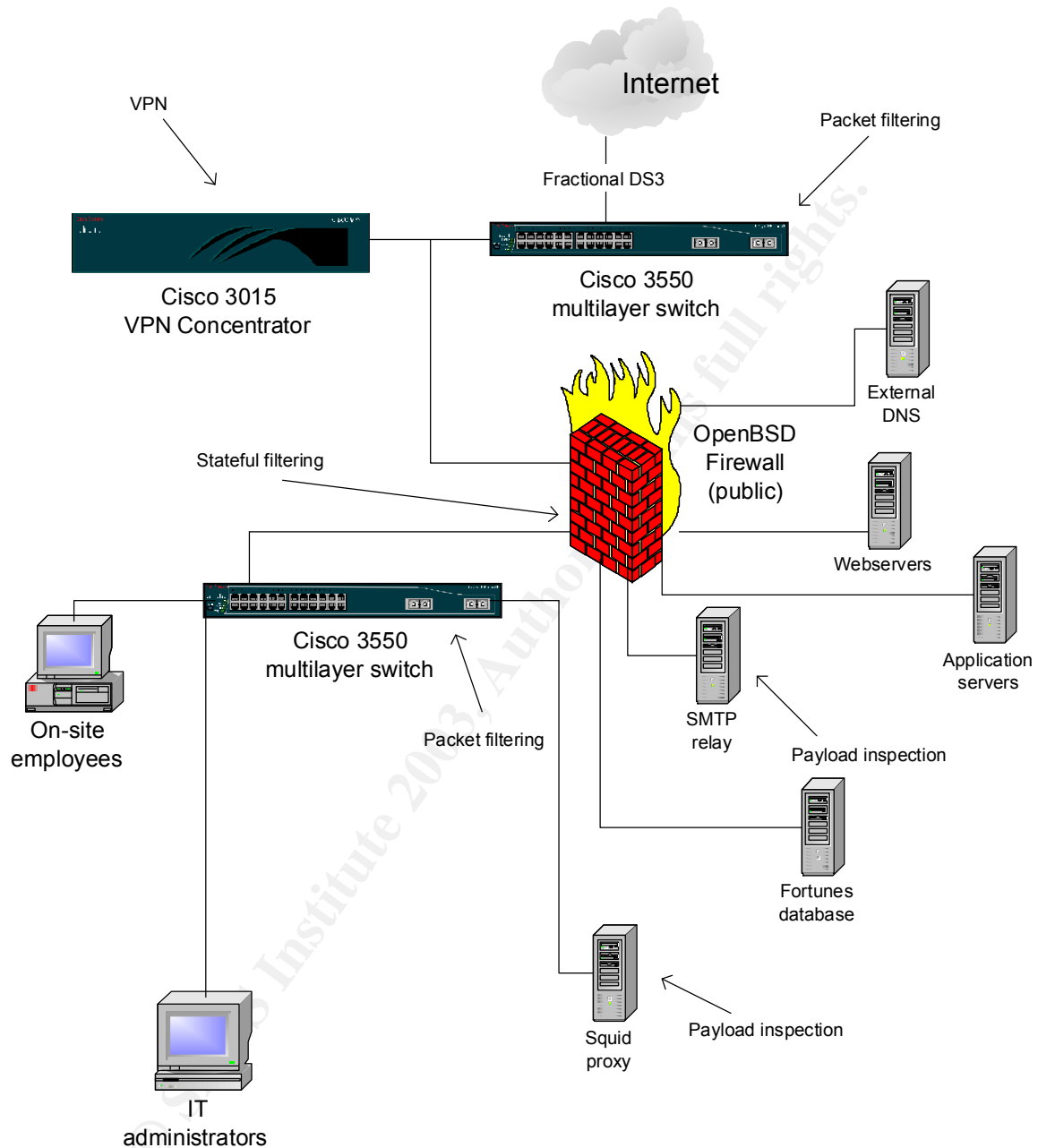
All network security devices will log back to a central, logging server. Syslog-ng¹⁰ is used, so that we may identify devices by IP, rather than by traditional syslog levels and services. The logging server is located on the internal network.

Databases

GIAC Enterprises operates its databases with MySQL version 3.23.55 running on RedHat 8.0. As per our suggestion, they have begun using stunnel to create encrypted tunnels between the application server and the MySQL server, as described at <http://www.stunnel.org/examples/mysql.html>.

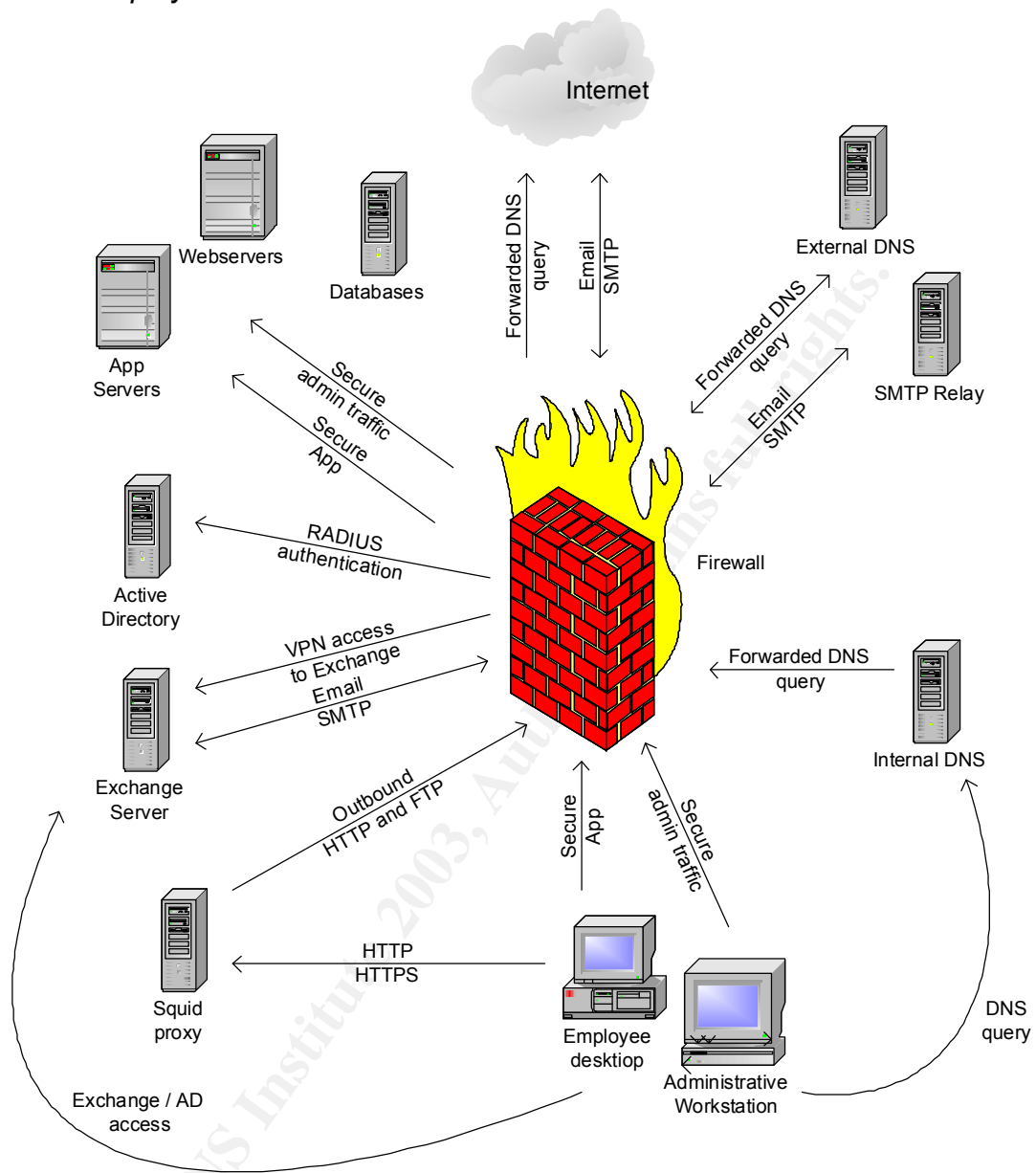
¹⁰ See <http://www.balabit.hu/en/downloads/syslog-ng/> for further information.

Physical Network Connections

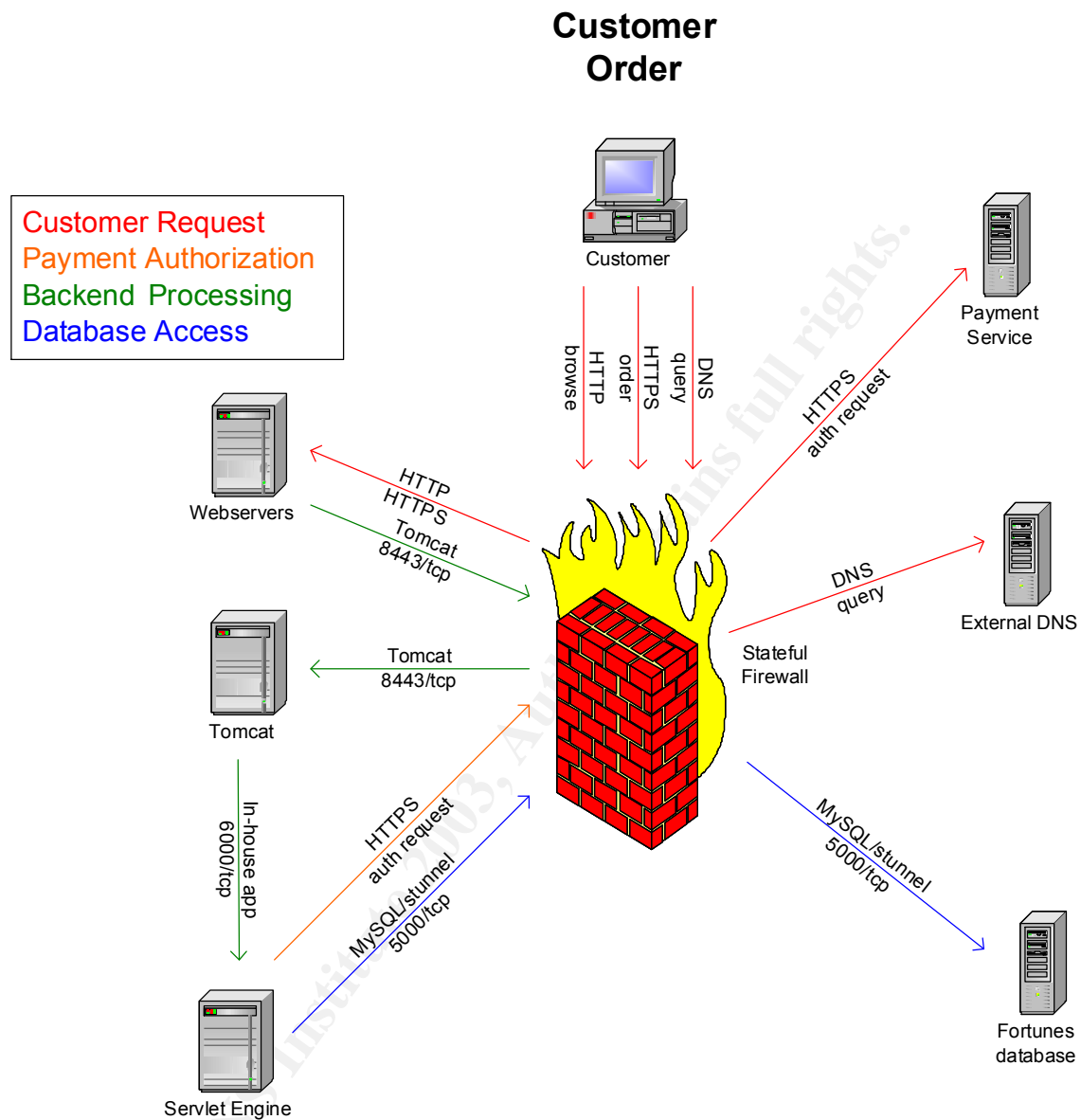


(IDS Sensors not shown)

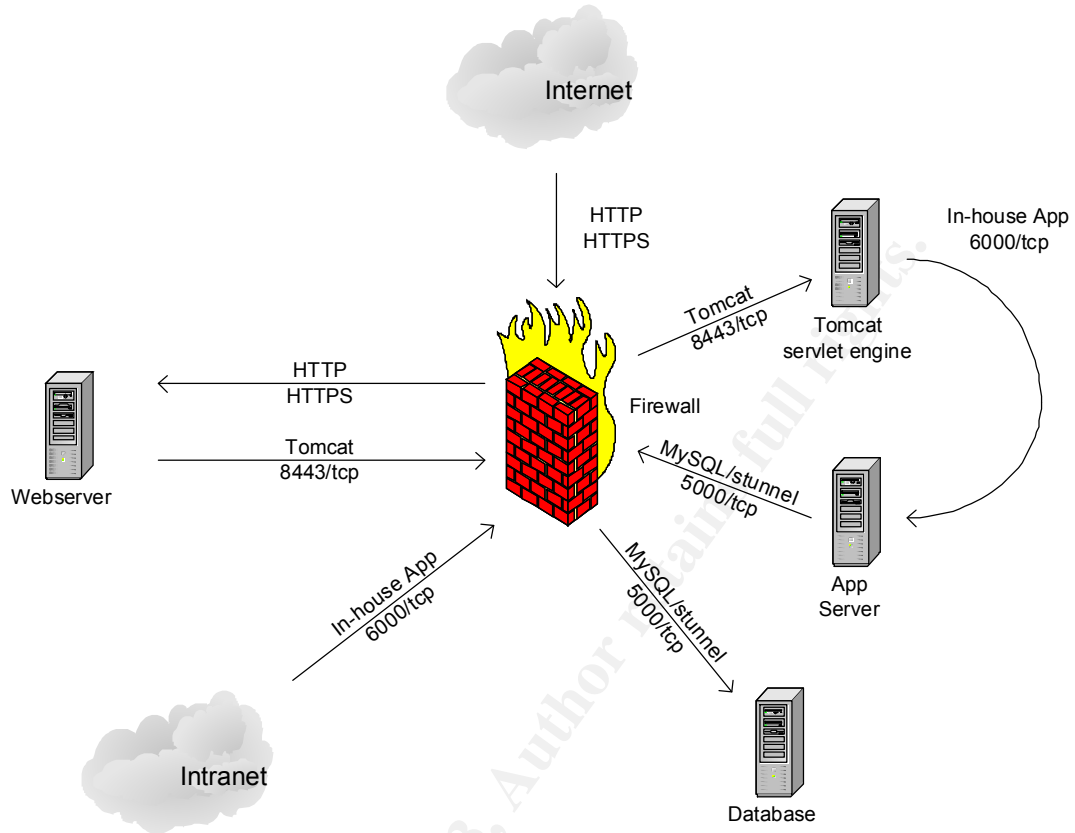
On-Site Employee Dataflow



Customer Order Dataflow



Application Server Dataflow



Security Policy

Border Router

The primary security duty of the border router is to filter inbound and outbound traffic in a very general sense. Since complex ACL rules tend to cause performance degradation, we have kept the ruleset fairly short and simple.

Ingress/Egress Filtering

Ingress/egress filtering of invalid network addresses using extended access lists in IOS. We are not allowing packets from networks reserved by the IANA¹¹, and those set aside by RFC 1918 as internal networks.

```
! IANA-reserved
! http://www.iana.org/assignments/ipv4-address-space
access-list 103 deny ip 1.0.0.0 0.255.255.255 any
access-list 103 deny ip 2.0.0.0 0.255.255.255 any
access-list 103 deny ip 5.0.0.0 0.255.255.255 any
access-list 103 deny ip 23.0.0.0 0.255.255.255 any
access-list 103 deny ip 27.0.0.0 0.255.255.255 any
access-list 103 deny ip 31.0.0.0 0.255.255.255 any
access-list 103 deny ip 36.0.0.0 1.255.255.255 any
access-list 103 deny ip 39.0.0.0 0.255.255.255 any
access-list 103 deny ip 41.0.0.0 0.255.255.255 any
access-list 103 deny ip 42.0.0.0 0.255.255.255 any
access-list 103 deny ip 58.0.0.0 1.255.255.255 any
access-list 103 deny ip 60.0.0.0 0.255.255.255 any
access-list 103 deny ip 70.0.0.0 1.255.255.255 any
access-list 103 deny ip 72.0.0.0 7.255.255.255 any
access-list 103 deny ip 83.0.0.0 0.255.255.255 any
access-list 103 deny ip 84.0.0.0 3.255.255.255 any
access-list 103 deny ip 88.0.0.0 7.255.255.255 any
access-list 103 deny ip 96.0.0.0 31.255.255.255 any
access-list 103 deny ip 198.18.0.0 0.0.1.255 any
access-list 103 deny ip 222.0.0.0 1.255.255.255 any
access-list 103 deny ip 224.0.0.0 15.255.255.255 any
access-list 103 deny ip 255.255.255.255 0.0.0.0.0 any
! RFC 1918 private nets
access-list 103 deny ip 0.0.0.0 0.255.255.255 any
access-list 103 deny ip 10.0.0.0 0.255.255.255 any
access-list 103 deny ip 127.0.0.0 0.255.255.255 any
access-list 103 deny ip 169.254.0.0 0.0.255.255 any
access-list 103 deny ip 172.16.0.0 0.7.255.255 any
access-list 103 deny ip 192.168.0.0 0.0.255.255 any
```

¹¹ <http://www.iana.org/assignments/ipv4-address-space>

```
access-list 103 deny ip 240.0.0.0 7.255.255.255 any
access-list 103 deny ip 248.0.0.0 7.255.255.255 any
```

Since there is no legitimate reason for us to be receiving packets within our address range from the outside world, we will filter these as well.

```
access-list 103 deny ip 192.0.2.0 0.0.0.15 any
```

Note that Cisco IOS uses *wildcard masks* for access lists, not the usual *netmasks*. A wildcard mask is similar to a netmask, except that the bits are inverted. This has the potential cause confusion when copying configurations from other security devices, and is not always obvious when examining a router's configuration. However, with familiarity and caution, the problem is easily avoided.

Since we are placing our VPN Concentrator in parallel with our firewall, we should allow allow ISAKMP, ESP, and AH traffic to the Concentrator. All other traffic to the Concentrator will be filtered:

```
! to VPN
access-list 103 permit udp any eq 500 \
    host 192.0.2.5 eq 500
access-list 103 permit 50 any host 192.0.2.5
access-list 103 permit 51 any host 192.0.2.5
access-list 103 deny ip any host 192.0.2.5
```

Many client VPN connections will be established from behind a firewall providing NAT services. Since VPN traffic carries the source address of the client computer in the payload, this would normally prevent such connections from succeeding. However, many vendors, including Cisco, support IPsec-over-UDP in order to allow these users to connect through a VPN. This type of connections communicates with the VPN Concentrator on UDP port 10000. The rule to allow this traffic is:

```
! IPsec-over-UDP
access-list 103 permit udp any any eq 10000
```

It is important to remember that IOS adds an implicit "deny any" statement to the end of any access list. Therefore, we must add a "permit" statement to allow traffic into our network:

```
access-list 103 permit ip any 192.0.2.0 0.0.0.15
```

To apply this ACL to all traffic incoming to our network, the following commands are issued:

```
border(config)# interface FastEthernet 0/0
border(config-if)# ip access-group 103 in
```

While it is also possible to filter this traffic when it is outbound from an inside interface, CPU cycles will be wasted processing packets that will be dropped. Therefore, we will filter traffic as it enters the router.

For sanity, we will also filter traffic from GIAC Enterprises to the Internet. Not only does egress filtering make the Internet a better place, it can prevent a misconfigured network device from forwarding (potentially sensitive) internal traffic across the DS3 link and on to the public Internet. We will also filter traffic destined for the nonroutable networks. Since we should be aware of this happening, we will log any such denies. Finally, even though a “deny any any” is the implicit last rule in IOS ACLs, to get messages logged about filtered packets, we must force the issue with a “deny any any log” rule.

```
! IANA-reserved and RFC 1918 private
! http://www.iana.org/assignments/ipv4-address-space
access-list 105 deny ip any 0.0.0.0 0.255.255.255 log
access-list 105 deny ip any 1.0.0.0 0.255.255.255 log
access-list 105 deny ip any 2.0.0.0 0.255.255.255 log
access-list 105 deny ip any 5.0.0.0 0.255.255.255 log
access-list 105 deny ip any 10.0.0.0 0.255.255.255 log
access-list 105 deny ip any 23.0.0.0 0.255.255.255 log
access-list 105 deny ip any 27.0.0.0 0.255.255.255 log
access-list 105 deny ip any 31.0.0.0 0.255.255.255 log
access-list 105 deny ip any 36.0.0.0 1.255.255.255 log
access-list 105 deny ip any 39.0.0.0 0.255.255.255 log
access-list 105 deny ip any 41.0.0.0 0.255.255.255 log
access-list 105 deny ip any 42.0.0.0 0.255.255.255 log
access-list 105 deny ip any 58.0.0.0 1.255.255.255 log
access-list 105 deny ip any 60.0.0.0 0.255.255.255 log
access-list 105 deny ip any 70.0.0.0 1.255.255.255 log
access-list 105 deny ip any 72.0.0.0 7.255.255.255 log
access-list 105 deny ip any 83.0.0.0 0.255.255.255 log
access-list 105 deny ip any 84.0.0.0 3.255.255.255 log
access-list 105 deny ip any 88.0.0.0 7.255.255.255 log
access-list 105 deny ip any 96.0.0.0 31.255.255.255 \
log
access-list 105 deny ip any 127.0.0.0 0.255.255.255 \
log
access-list 105 deny ip any 169.254.0.0 0.0.255.255 \
log
access-list 105 deny ip any 172.16.0.0 0.15.255.255 \
log
access-list 105 deny ip any 192.168.0.0 0.0.255.255 \
log
access-list 105 deny ip any 198.18.0.0 0.0.1.255 log
access-list 105 deny ip any 222.0.0.0 1.255.255.255 \
log
```



```

access-list 105 deny ip any 224.0.0.0 15.255.255.255 \
    log
access-list 105 deny ip any 240.0.0.0 7.255.255.255 \
    log
access-list 105 deny ip any 248.0.0.0 7.255.255.255 \
    log
access-list 105 deny ip any 255.255.255.255 \
    0.0.0.0 log
! Ourselves
access-list 105 permit ip 192.0.2.0 0.0.0.15 any
! default deny won't log so we do it here ourselves
access-list 105 deny ip any any log

```

Again, we apply the ACL to inbound traffic on our inside interface:

```

border(config)# interface FastEthernet0/1
border(config-if)# ip access-group 105 in

```

Since we are placing our VPN Concentrator in parallel with our firewall, we should allow only ISAKMP, ESP, and AH traffic from the Concentrator:

```

! from VPN
access-list 112 permit udp host 192.0.2.5 eq 500 \
    any eq 500
access-list 112 permit 50 host 192.0.2.5 any
access-list 112 permit 51 host 192.0.2.5 any

```

Additionally, IPSec-over-UDP clients will communicate with the Concentrator via 10000/udp:

```

access-list 112 permit udp host 192.0.2.5 \
    eq 10000 any

```

This access-list will also be applied to inbound traffic:

```

border(config)# interface FastEthernet0/3
border(config-if)# ip access-group 112 in

```

SSH Management

Configuring the router for ssh and only ssh logins, using 2048-bit RSA keys. 2048-bit keys are chosen simply because they are the strongest possible:

```

border(config)# crypto key generate rsa

```

The name for the keys will be: border.giac-fortunes.com
Choose the size of the key modulus in the range of 360 to 2048 for your General Purpose Keys. Choosing a key modulus greater than 512 may take a few minutes.

How many bits in the modulus [512]: 2048
Generating RSA keys ...

To prevent non-ssh login attempts:

```
border(config)# line vty 0 4
border(config-line)# transport input ssh
```

Require login with username:

```
border(config)# aaa new-model
border(config)# username <user> password 0 <password>
```

The router may now be logged into only by:

```
unix% ssh user@router
```

For further information, see “Configuring Secure Shell on Cisco IOS Routers”¹².

Controlling access to the router with access lists

IOS ACLs can be used to control access to router resources, including login access

```
! permit management workstations betta and molly
access-list 9 permit 192.0.2.50
access-list 9 permit 192.0.2.51
```

ACLs are applied to vty's in much the same way as to interfaces:

```
border(config)# line vty 0 4
border(config-line)# access-class 9 in
```

The Center for Internet Security provides an IOS auditing tool that includes the NSA's Router Security Configuration Guide¹³. This guide is a useful tool in securing Cisco IOS routers. The networking department at GIAC Enterprises has found it extremely useful and uses it as a reference in configuring the security of their routers.

The following global options are set:

¹² <http://www.cisco.com/warp/public/707/ssh.shtml>

¹³ Available for download with the Router Assessment Tool at <http://www.cisecurity.org>. CIS also provides excellent baseline scanners and configuration guidelines for Solaris, Linux, Windows 2000, and other server operating systems.

Cisco devices typically provide information to one another using the Cisco Discovery Protocol. Since this information can be useful to an attacker, this service will be disabled:

```
no cdp run
```

For some unknown reason, Cisco provides echo, chargen, daytime, and discard servers, which are enabled by default in IOS. These should be disabled:

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

The finger service allows users to see who is logged into the router. It goes without saying that this should be disabled:

```
no service finger
```

Cisco routers may be administered through a web console. This has historically had severe security problems, and can consume a fair amount of precious router resources. It will not be missed:

```
no ip http server
```

Again, another service that should be disabled:

```
no ip bootp server
```

Source-routed packets should not be allowed through:

```
no ip source route
```

The following options will be specified in the interface configurations.

Proxy-arping allows hosts without a default gateway communicate with hosts on other subnets. As such, we will not need it:

```
no ip proxy-arp
```

This is a protection against Smurf DoS attacks. It's turned off by default in IOS, but it won't hurt to double-check:

```
no ip-directed-broadcasts
```

Our router can be configured not to send ICMP-Unreachables to requests for nonexistent hosts. This and the following two can provide malicious users with information about the internal makeup of our network, and will be disabled:

```
no ip unreachable  
no ip redirect  
no ip mask-reply
```

Additionally, for correct typing-challenged administrators, the “no ip-domain-lookup” option has been specified. This prevents the router from attempting to `nslookup` and then `telnet` to a host by the name of any mistyped command.

Syslog will be configured, so that we can receive information on access-list denies, as well as hardware problems and other, exceptional events:

```
logging trap debugging
logging facility local0
logging source-interface FastEthernet0/1
logging 192.0.2.11
```

SNMP is used for statistical purposes. So rather than disable it, we have changed the community names from the mind-boggling defaults of “public” and “private.”

```
snmp-server community lilsecret RO
snmp-server community bigsecret RW
```

A password has been set on the router’s console, in order to minimize the risk of the router being accessed by someone with physical access.

```
line con 0
  exec-timeout 5 0
  password not4u
  login
```

Remote access should be limited to only ssh connections.

```
line vty 0 4
  access-class 99 in
  exec-timeout 5 0
  password not4u
  login
  transport input ssh
```

Firewall

The following configuration is the firewall ruleset.

The following variables, defined at the beginning of the `/etc/pf.conf` file, will be used in the rules¹⁴:

¹⁴ This document uses 192.0.2.0/24 as GIAC Enterprises’ global address range, rather than the actual address used. This firewall ruleset reflects this usage and therefore does not include this network in the “private_net” variable. For production usage, the actual GIAC Enterprises network range will be used, and 192.0.2.0/24 filtered. Additionally, the address used here for MonkeyPay is within an IANA-reserved network range.

Options

```
## global config variables
# interfaces
internet_if="fxp0"
backend_if="fxp1"
web_if="fxp2"
smtp_if="fxp3"
vpn_if="fxp4"
db_if="fxp5"
dns_if="fxp6"
internal_if="fxp7"

## networks
# See RFCs 1918 and 3330
# and http://www.iana.org/assignments/ipv4-address-space
private_net="{ 1.0.0.0/8, 2.0.0.0/8, 5.0.0.0/8, \
    7.0.0.0/8, 10.0.0.0/8, 23.0.0.0/8, \
    27.0.0.0/8, 31.0.0.0/8, 36.0.0.0/7, \
    39.0.0.0/8, 41.0.0.0/8, 42.0.0.0/8, \
    58.0.0.0/7, 60.0.0.0/8, 70.0.0.0/7, \
    72.0.0.0/5, 83.0.0.0/8, 84.0.0.0/6, \
    88.0.0.0/5, 96.0.0.0/3, 169.254.0.0/16, \
    172.16.0.0/12, 192.168.0.0/16, \
    198.18.0.0/15, 222.0.0.0/7, 224.0.0.0/4, \
    240.0.0.0/5, 248.0.0.0/5, 255.255.255.255/32 }"
global_net="192.0.2.0/28"
backend_net="10.30.20.0/24"
web_net="10.30.10.0/24"
smtp_net="10.20.10.0/24"
vpn_net="10.40.0.0/24"
db_net="10.30.0.0/24"
dns_net="10.20.0.0/24"
internal_net="10.10.0.0/16"
management_net="10.10.254.0/24"
# hosts on screened subnets
tomcat_srv="10.30.20.1"
app_srv="10.30.20.2"
http_srv="10.30.10.1"
https_srv="10.30.10.11"
smtp_rly="10.20.10.1"
vpn_cnc="10.40.0.253"
db_srv="10.30.0.1"
dns_ext="10.20.0.1"

# internal hosts
```

```
proxy_srv="10.10.8.1"
exchange_srv="10.10.10.1"
radius_srv="10.10.10.2"
dns_int="10.10.7.1"
syslog_srv="10.10.9.1"
ntp_srv="10.10.9.7"
fw_int="10.10.255.254"

# global addresses
http_srv_global="192.0.2.1"
https_srv_global="192.0.2.2"
smtp_rly_global="192.0.2.3"
dns_ext_global="192.0.2.4"
syslog_srv_global="192.0.2.11"
ntp_srv_global="192.0.2.12"
proxy_srv_global="192.0.2.13"

# outside hosts
monkeypay="169.254.23.217"
router="192.0.2.254"

# management stations
betta="10.10.254.50"
betta_global="192.0.2.50"
molly="10.10.254.51"
molly_global="192.0.2.51"
```

Traffic normalization

By enabling traffic normalization, pf reassembles IP fragments, eliminating any overlapping data. This protects hosts behind the firewall with vulnerable IP stacks. The directive to enable traffic normalization is:

```
scrub in all
```

NAT and redirection

NAT and redirection rules are evaluated from top-to-bottom, and the first matching rule determines the action taken. Since we are implementing private addressing at GIAC Enterprises, we will need to configure NAT to enable our networks to communicate.

Also, although we not be using it, it is possible to configure redirection to forward certain requests to proxy servers.

Notice that the web, mail, external DNS, syslog, and NTP hosts have bidirectional NAT rules, while the proxy server and management have stations

only unidirectional NAT. This is because external hosts will need to initiate communications with the other services, but not to the proxy server or management stations. Therefore, their private addresses will not be translated for inbound traffic.

```
binat on $internet_if from $http_srv to \
    any -> $http_srv_global
binat on $internet_if from $https_srv to \
    any -> $https_srv_global
binat on $internet_if from $smtp_rly to \
    any -> $smtp_rly_global
binat on $internet_if from $dns_ext to \
    any -> $dns_ext_global
binat on $internet_if from $syslog_srv to \
    any -> $syslog_srv_global
binat on $internet_if from $ntp_srv to \
    any -> $ntp_srv_global
nat on $internet_if from $proxy_srv to \
    any -> $proxy_srv_global
nat on $internet_if from $betta to \
    $router -> $betta_global
nat on $internet_if from $molly to \
    $router -> $molly_global
```

Filter

Filter rules are evaluated from top-to-bottom, and the last matching rule determines the action taken (can be short-circuited with “quick” statement). By default, pf’s behavior is that of implicit allow. To change this to the more secure default deny, the first two filter rules must be:

```
block in log all
block out log all
```

This will prevent any packet from traversing the firewall, unless it is explicitly allowed later in the ruleset.

First, we will allow traffic in on the loopback interface. We need to specify this, since the above default deny applies to all interfaces.

```
pass in quick on lo0 from any to any
pass out quick on lo0 from any to any
```

Next, we will prevent any private addresses from entering our firewall from the outside, where they should not be coming from. Although our border router is configured to perform this filtering, we are checking for it here for defense-in-depth. We will also filter our own network addresses from the outside. In both cases, we will log the packets. Since these packets will require no more

processing, and to do so would merely waste CPU time, the “quick” directive breaks the evaluation. Before we block traffic from ourselves from the outside, we should allow syslog and back from the router.

```
block return-rst in log on { $internet_if, \
    $backend_if, $web_if, $smtp_if, $vpn_if, $db_if, \
    $dns_if, $internal_if } proto tcp all label \
    "default tcp deny in"
block return-icmp in log on { $internet_if, \
    $backend_if, $web_if, $smtp_if, $vpn_if, $db_if, \
    $dns_if, $internal_if } proto udp all label \
    "default udp deny in"
block return-icmp in log on { $internet_if, \
    $backend_if, $web_if, $smtp_if, $vpn_if, $db_if, \
    $dns_if, $internal_if } proto icmp all label \
    "default icmp deny in"
block out log all label "default deny out"

pass in on $internet_if proto udp from $router to \
    $syslog_srv port 514
block in log quick on $internet_if from \
    $private_net to any label "private net deny in"
block in quick on $internet_if from $global_net to \
    any label "our address deny"
```

The SMTP relay needs to be able to accept and send mail from the outside as well as the corporate Exchange server. The “modulate state” option instructs the firewall to randomize TCP sequences for greater protection.

```
pass in on $internet_if proto tcp from any to \
    $smtp_rly port 25 modulate state
pass in on $internal_if proto tcp from $exchange_srv \
    to $smtp_rly port 25 modulate state
pass in on $smtp_if proto tcp from $smtp_rly to any \
    port 25 modulate state
block in log on $smtp_if proto tcp from $smtp_rly to \
    $internal_net port 25
pass in on $smtp_if proto tcp from $smtp_rly to \
    $exchange_srv port 25 modulate state
```

Additionally, the mail server needs to be able to query the DNS to be able to deliver messages.

```
pass in on $smtp_if proto udp from $smtp_rly \
    to $dns_ext eq 53 keep state
```


The external DNS server will need to receive DNS queries from outside hosts, as well as acting as a forwarder for the internal DNS server. Since the internal DNS is restricted to initiating queries on port 53, we may be further restrictive in its rule.

```
# External DNS
pass in on $internet_if proto udp from any to \
    $dns_ext port 53 keep state
pass in on $dns_if proto udp from $dns_ext to any \
    port 53 keep state
pass in on $internal_if proto udp from $dns_int port \
    53 to $dns_ext port 53 keep state
```

Customers and partners will need to access the webserver and secure webserver, as will GIAC Enterprises employees:

```
# Public webserver
pass in on { $internet_if, $internal_if } proto tcp \
    from any to $http_srv port 80 modulate state
pass in on { $internet_if, $internal_if } proto tcp \
    from any to $https_srv port 443 modulate state
```

The webserver will communicate with the servlet engine for dynamic content. Note that this is a secure connection, using Tomcat's default 8443 for SSL.

```
# Webserver -> Servlet engine
pass in on $web_if proto tcp from $web_net \
    to $tomcat_srv eq 8443 modulate state
```

The application server, which processes online orders, will need to contact MonkeyPay, a company used by GIAC Enterprises to authorize credit card transactions.

```
# App server -> authorization
pass in on $backend_if proto tcp from $backend_net \
    to $monkeypay eq 443 modulate state
```

The application server will need to initiate communications to the database. Stunnel is used to create a secure connection for MySQL on an arbitrary port, in this case, 5000/tcp:

```
# App Server -> database
pass in on $backend_if proto tcp from $app_srv \
    to $db_net eq 5000 modulate state
```

All of our devices send syslogs back to a centralized logging server. Since syslog uses UDP and does not use any form of acknowledgements, state does not need to be kept.

```
pass in on $backend_if proto udp from $backend_net \
    to $syslog_srv port 514
pass in on $web_if proto udp from $web_net to \
    $syslog_srv port 514
pass in on $smtp_if proto udp from $smtp_net to \
    $syslog_srv port 514
pass in on $vpn_if proto udp from $vpn_cnc to \
    $syslog_srv port 514
pass in on $db_if proto udp from $db_net to \
    $syslog_srv port 514
pass in on $dns_if proto udp from $dns_ext to \
    $syslog_srv port 514
```

VPN users will authenticate via RADIUS, and will only be able to access the Exchange 2000 server. Since Exchange 2000 communicates on random ports by default, and GIAC Enterprises' administrators have deemed the necessary registry modifications dangerous, all communications between the VPN pool and the Exchange server have been left open. Management has decided to revisit this issue at a later date.

```
# VPN to RADIUS and VPN users to MS Exchange
pass in on $vpn_if proto udp from $vpn_cnc \
    to $exchange_srv eq 53 keep state
pass in on $vpn_if proto tcp from $vpn_cnc \
    to $exchange_srv eq 53 modulate state
pass in on $vpn_if proto tcp from $vpn_cnc \
    to $exchange_srv eq 88 modulate state
pass in on $vpn_if proto tcp from $vpn_cnc \
    to $exchange_srv eq 123 modulate state
pass in on $vpn_if proto tcp from $vpn_cnc \
    to $exchange_srv eq 135 modulate state
pass in on $vpn_if proto tcp from $vpn_cnc \
    to $exchange_srv eq 389 modulate state
pass in on $vpn_if proto udp from $vpn_cnc \
    to $exchange_srv eq 389 keep state
pass in on $vpn_if proto tcp from $vpn_cnc \
    to $exchange_srv eq 445 modulate state
pass in on $vpn_if proto tcp from $vpn_cnc \
    to $exchange_srv eq 1025 modulate state
pass in on $vpn_if proto tcp from $vpn_cnc \
    to $exchange_srv eq 3268 modulate state
```

On-site GIAC Employees will work with fortune sayings through an in-house application that communicates with an application server, located on a screened subnet. As such, employees' systems will need to communicate with the application server, on port 6000/tcp.

```
# Internal access
pass in on $internal_if proto tcp from \
    $internal_net to $app_srv eq 6000 modulate state
```

The proxy server needs to be able to communicate with external systems, however it should not be accessing systems on the screened subnets.

```
# Web access
pass in on $internal_if proto tcp from $proxy_srv \
    to any modulate state
block in on $internal_if proto tcp from $proxy_srv \
    to $internal_net
```

IT administrators access systems using encrypted management protocols. They also need to be able to check on public services.

```
# IT access
pass in on $internal_if proto tcp from \
    $management_net to $tomcat_srv port 22 modulate \
    state
pass in on $internal_if proto tcp from \
    $management_net to $app_srv port 22 modulate \
    state
pass in on $internal_if proto tcp from \
    $management_net to $http_srv port 22 modulate \
    state
pass in on $internal_if proto tcp from \
    $management_net to $https_srv port 22 modulate \
    state
pass in on $internal_if proto tcp from \
    $management_net to $smtp_rly port 22 modulate \
    state
pass in on $internal_if proto tcp from \
    $management_net to $db_srv port 22 modulate state
pass in on $internal_if proto tcp from \
    $management_net to $dns_ext port 22 modulate \
    state
pass in on $internal_if proto udp from \
    $management_net to $dns_ext port 53 keepstate
pass in on $internal_if proto tcp from \
    $management_net to $router port 22 modulate \
    state
```

```
# management access to firewall
pass in on $internal_if proto tcp from \
    $management_net to $fw_int eq 22 modulate state
```

The firewall will also need to be able to log to syslog, as well as synchronize its clock with an NTP server.

```
# syslog from firewall
pass out on $internal_if proto udp from \
    $fw_int to $syslog_srv eq 514
pass out on $internal_if proto udp from \
    $fw_int to $ntp_srv eq 123 keep state
```

The router will also need access to the NTP server.

```
pass in on $internet_if proto udp from \
    $router to $ntp_srv eq 123 keep state
```

Finally, we need to allow our approved traffic out the other side of the firewall for each interface.

```
# outbound traffic
pass out all keep state
block out log on $internet_if from \
    $private_net to any label "private net deny out"
```

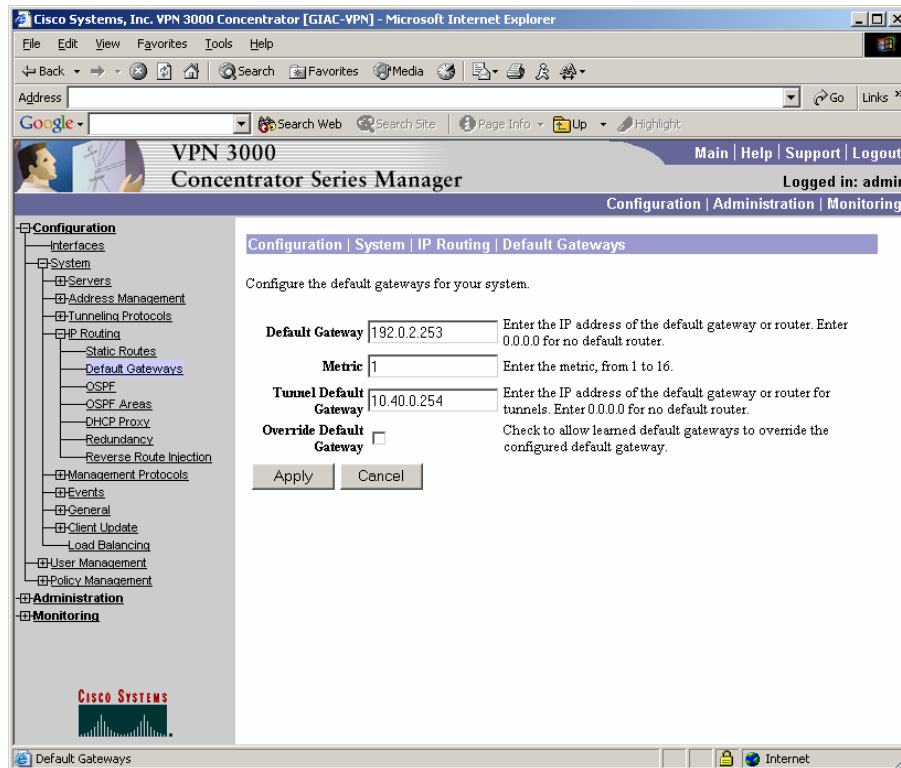
VPN

GIAC Enterprises VPN clients will connect to the corporate Intranet by using the Cisco VPN Client. This client supports strong encryption and is designed to work with Cisco VPN Concentrators, PIX firewalls, and IOS routers.

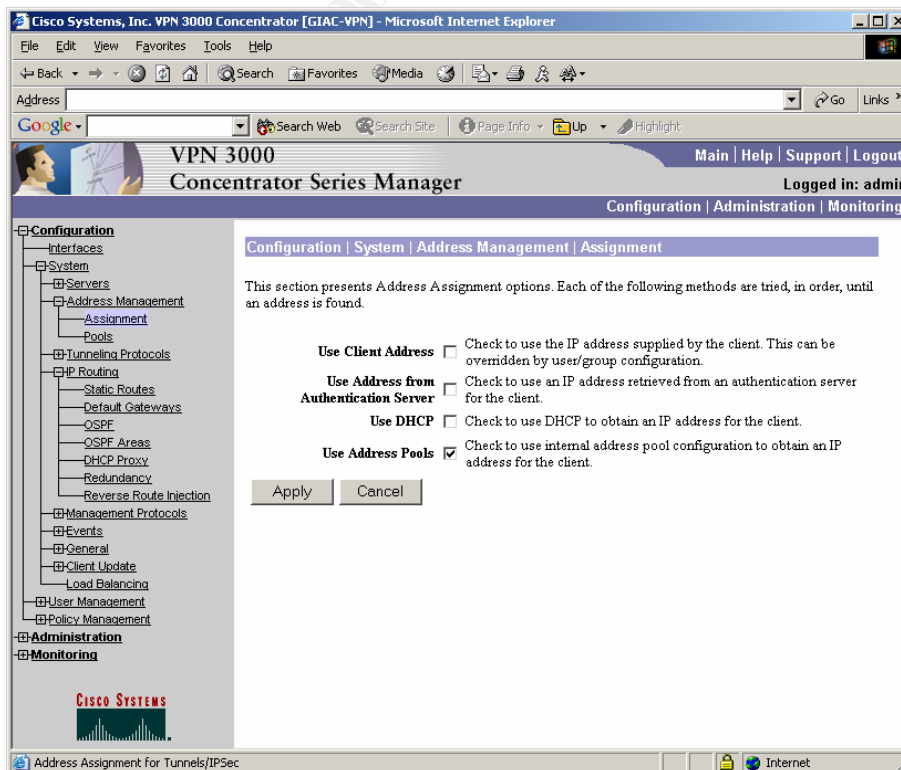
GIAC Enterprises mobile users and telecommuters will authenticate against the Active Directory using RADIUS, and will have access to (and only to) the corporate Microsoft Exchange server. Here we will detail the settings that deal with IPsec.

Client Addressing

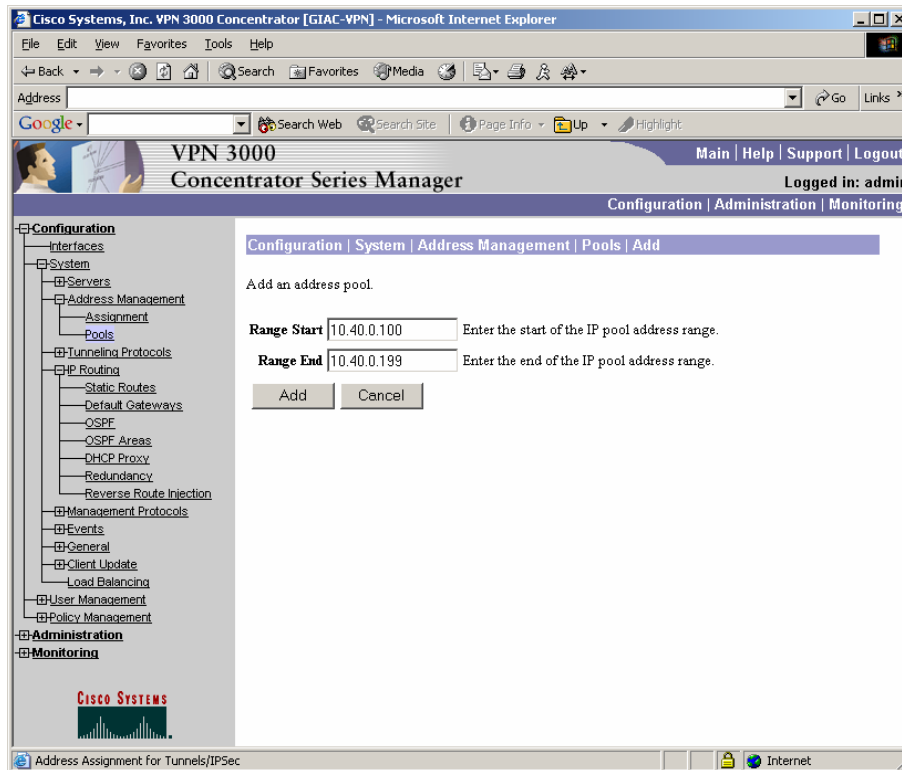
The default gateways for the VPN must be set for the public and private interfaces:



The Concentrator will assign addresses from a pool of addresses on its private interface. It is also possible to use a DHCP server on the private network to do this.



The 3015 supports up to 100 simultaneous sessions, but may be upgraded with encryption hardware to support over 1,000. Since we have a standard 3015, there is no need for more than 100 addresses in the pool.



IPSec Configuration

We are using a VPN group named 'giac.' The group name and its password are used by the Concentrator to create a hash when connecting. Notice that this group is going to use an external authentication method.

Cisco Systems, Inc. VPN 3000 Concentrator [GIAC-VPN] - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Search Favorites Media Up Highlight

Address

Google Search Web Search Site Page Info Up Highlight

VPN 3000
Concentrator Series Manager

Main | Help | Support | Logout

Logged in: admin

Configuration | Administration | Monitoring

Configuration | User Management | Groups | Add

This section lets you add a group. Check the **Inherit?** box to set a field that you want to default to the base group value. Uncheck the **Inherit?** box and enter a new value to override base group values.

Identity General IPsec Mode Config Client FW HW Client PPTP/L2TP

Attribute	Value	Description
Group Name	giac	Enter a unique name for the group.
Password	*****	Enter the password for the group.
Verify	*****	Verify the group's password.
Type	External	External groups are configured on an external authentication server (e.g. RADIUS). Internal groups are configured on the VPN 3000 Concentrator's Internal Database.

Add Cancel

CISCO SYSTEMS

Group Parameters

Internet

© SANS Institute 2003, Author

This is the “General” tab under the group configuration. Here we have limited the number of simultaneous logins to one, as opposed to the three allowed by default. Additionally, we will only be allowing members of this group to connect with an IPSec client.

The screenshot shows the Cisco VPN 3000 Concentrator web interface in Microsoft Internet Explorer. The page title is "VPN 3000 Concentrator Series Manager". The user is logged in as "admin". The navigation menu on the left includes Configuration, System, User Management, Policy Management, Administration, and Monitoring. The main content area shows the "General Parameters" tab for a group configuration. The table below represents the data visible in the interface.

Attribute	Value	Inherit?	Description
Access Hours	No Restrictions	<input checked="" type="checkbox"/>	Select the access hours assigned to this group.
Simultaneous Logins	1	<input type="checkbox"/>	Enter the number of simultaneous logins for this group.
Minimum Password Length	6	<input checked="" type="checkbox"/>	Enter the minimum password length for users in this group.
Allow Alphabetic-Only Passwords	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Enter whether to allow users with alphabetic-only passwords to be added to this group.
Idle Timeout	30	<input checked="" type="checkbox"/>	(minutes) Enter the idle timeout for this group.
Maximum Connect Time	0	<input checked="" type="checkbox"/>	(minutes) Enter the maximum connect time for this group.
Filter	None	<input checked="" type="checkbox"/>	Enter the filter assigned to this group.
Primary DNS	10.20.0.1	<input type="checkbox"/>	Enter the IP address of the primary DNS server.
Secondary DNS	10.20.0.2	<input type="checkbox"/>	Enter the IP address of the secondary DNS server.
Primary WINS	10.10.10.2	<input type="checkbox"/>	Enter the IP address of the primary WINS server.
Secondary WINS	10.10.10.3	<input type="checkbox"/>	Enter the IP address of the secondary WINS server.
SEP Card Assignment	<input checked="" type="checkbox"/> SEP 1 <input checked="" type="checkbox"/> SEP 2 <input checked="" type="checkbox"/> SEP 3 <input checked="" type="checkbox"/> SEP 4	<input checked="" type="checkbox"/>	Select the SEP cards this group can be assigned to.
Tunneling Protocols	<input type="checkbox"/> PPTP <input type="checkbox"/> L2TP <input checked="" type="checkbox"/> IPSec <input type="checkbox"/> L2TP over IPSec	<input type="checkbox"/>	Select the tunneling protocols this group can connect with.
Strip Realm	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Check to remove the realm qualifier of the user name during authentication.

Here are the IPSec options for this group. Note that the Security Association is ESP-3DES-MD5, and that the tunnel type is “Remote Access.”

Cisco Systems, Inc. VPN 3000 Concentrator [GIAC-VPN] - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address: https://134.124.252.1/access.html

VPN 3000 Concentrator Series Manager

Main | Help | Support | Logout

Configuration | Administration | Monitoring

Configuration | User Management | Groups | Add

This section lets you add a group. Check the **Inherit?** box to set a field that you want to default to the base group value. Uncheck the **Inherit?** box and enter a new value to override base group values.

Identity | General | IPsec | Mode Config | Client FW | HW Client | PPTP/L2TP

IPsec Parameters			
Attribute	Value	Inherit?	Description
IPsec SA	ESP-3DES-MD5	<input checked="" type="checkbox"/>	Select the group's IPsec Security Association.
IKE Peer Identity Validation	If supported by certificate	<input checked="" type="checkbox"/>	Select whether or not to validate the identity of the peer using the peer's certificate.
IKE Keepalives	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Check to enable the use of IKE keepalives for members of this group.
Tunnel Type	Remote Access	<input checked="" type="checkbox"/>	Select the type of tunnel for this group. Update the Remote Access parameters below as needed.
Remote Access Parameters			
Group Lock	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Lock users into this group.
Authentication	RADIUS	<input checked="" type="checkbox"/>	Select the authentication method for members of this group. This parameter does not apply to Individual User Authentication .
IPComp	None	<input checked="" type="checkbox"/>	Select the method of IP Compression for members of this group.
Reauthentication on Rekey	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Check to reauthenticate the user on an IKE (Phase-1) rekey.
Mode Configuration	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Check to initiate the exchange of Mode Configuration parameters with the client. This must be checked if version 2.5 (or earlier) of the Altiq/Cisco client are being used by members of this group.

Add Cancel

Filters and Access Policies

Internet

At our recommendation, split tunneling, which allows clients to only use the VPN connection when talking to the GIAC Enterprises network, is not allowed. Although the option improves performance, since less traffic is encrypted, it opens the possibility of a client to be subverted while connected to the VPN, which would give the attacker a protected pathway into the corporate network. Also note the use of a banner prohibiting “unauthorized use.” IPSec-over-UDP tunneling is enabled for users behind firewalls at home, which is how telecommuters are configured.

Attribute	Value	Inherit?	Description
Banner	Unauthorized use prohibited.	<input type="checkbox"/>	Enter the banner for this group. Only software clients see the banner.
Allow Password Storage on Client	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Check to allow the IPSec client to store the password locally.
Split Tunneling Policy	<input checked="" type="radio"/> Tunnel everything <input type="checkbox"/> Allow the networks in list to bypass the tunnel <input type="radio"/> Only tunnel networks in the list	<input checked="" type="checkbox"/>	Select the method and network list to be used for Split Tunneling. Tunnel Everything: Send all traffic through the tunnel. Allow the networks in the list to bypass the tunnel: The VPN Client may choose to send traffic to addresses in this list to the client's LAN. Send all other traffic through the tunnel. NOTE: This setting does not apply to the VPN 3002 Hardware Client. Tunnel networks in the list: Send traffic to addresses in this list through the tunnel. Send all other traffic to the client's LAN.
Split Tunneling Network List	--None--	<input checked="" type="checkbox"/>	
Default Domain Name		<input checked="" type="checkbox"/>	Enter the default domain name given to users of this group.
IPSec over UDP	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Check to allow the IPSec client to operate through a firewall using NAT via UDP.
IPSec over UDP Port	10000	<input checked="" type="checkbox"/>	Enter the UDP port to be used for IPSec through NAT (4001 - 49151).
IPSec Backup Servers	Use Client Configured List	<input checked="" type="checkbox"/>	<ul style="list-style-type: none"> Select a method for VPN 3002 to use or disable backup servers. Enter up to 10 IPSec backup server addresses/names starting from high priority to low. Enter each IPSec backup server address/name on a single line.

Configuration of OpenBSD firewall tutorial

This tutorial details installation and configuration of OpenBSD as a firewall. Since the use of OpenBSD as a production firewall is not as prevalent as Solaris or Linux, it was decided to include a basic installation and hardening guide in Appendix A.

The pf project was initially started as a result of a dispute between the OpenBSD core and Darren Reed, creator and maintainer of the IPFilter, in May 2001. Through the 2.9 release, IPFilter was included as part of the OpenBSD base system. For 3.0, however, the project removed IPFilter in favor of a homegrown project that was tightly integrated with the system. Development took place rapidly, and pf sported many of the same features of IPFilter within several months. With the 3.2 release, we feel that the project has matured sufficiently for use in a production environment.

Configuration file

Firewall rules are typically placed in the default pf configuration file, `/etc/pf.conf`. The following tutorial follows the order and format of rules listed in the `pf.conf(5)`¹⁵ manpage, for easier reference.

There are four sections in the configuration file. The first section defines options, the second deals with packet normalization, the third handles NAT and redirection, and the final section defines the packet-filtering and stateful firewall rules. The sections must be in this order.

Options

Pf supports many options, including timeouts, limits, and optimization rules. However, they are typically left alone and not changed from the defaults. Therefore, they will not be covered in this tutorial.

Packet Normalization

```
scrub

no-df

min-ttl <number>

max-mss <number>
```

¹⁵ <http://www.openbsd.org/cgi-bin/man.cgi?query=pf.conf&apropos=0&sektion=0&manpath=OpenBSD+3.2&arch=i386&format=html>

The 'scrub' option turns on packet normalization and defragmentation. It is primarily used to protect networks against attacks involving carefully crafted packet fragments that, when reassembled, cause problems for certain TCP/IP stacks.

The 'no-df' option tells pf to remove DF (don't fragment) bits. 'Min-ttl' sets a minimum TTL (time-to-live) value that will be allowed through the firewall, while 'max-mss' sets a maximum segment size.

NAT and Redirection

```
binat on <interface> from <address> to <address> ->  
<address>
```

Performs bidirectional mapping between an external and internal IP address. This allows a host's address to be translated not only for outbound connections, but also allows inbound traffic to be translated.

For example, to allow a webserver on a private network to be seen on a global address:

```
binat on $wan_link from $web_int to any -> $web_global  
nat on <interface> from <address> to <address>  
    <address>  
no nat on <interface> from <address> to <address>
```

Performs NAT (or not) on the packet matching the rule. Unlike filter rules, NAT rules are first match. So if there is an address that should *not* be NAT'd, a "no nat" rule should be placed before it:

```
# give everyone but bill access  
no nat on $internet from $bill to any -> $pat_ip  
nat on $internet from $intranet to any -> $pat_ip
```

A NAT rule commonly seen on home DSL or cable connections is:

```
nat on tun0 from $homenet to any -> tun0
```

Any packets from the "homenet" entering the firewall are translated to the outside interface. Note that, in this example, the outside address is not specified, only the interface. This is because short DHCP leases on broadband connections typically force new addresses frequently.

rdr

Redirects the packet matching the rule to another designation address and/or port. Packets sent in reply will appear to come from the external address.

Additionally, an entry is made into the state table for such connections.

The difference between a redirect and binat rule is that a

The following example shows inbound HTTP traffic on a DSL link being forwarded to two web servers on a NAT'd internal network, depending on the external port specified. Inbound SMTP is sent to a separate host, all using the same external address:

```
rdr on tun0 proto tcp from any to any port 80 \  
-> 192.168.0.11 port 80  
rdr on tun0 proto tcp from any to any port 8080 \  
-> 192.168.0.12 port 80  
rdr on tun0 proto tcp from any to any port 25 \  
-> 192.168.0.14 port 25
```

It is important to note that NAT and redirection rules are processed *before* any rules. Therefore, when using NAT, be sure to use internal IP addresses in your rules, because packets inbound from the outside will have already been translated.

Since the nat, binat, and rdr rules perform similar functions, it is important to understand their differences. The nat rule provides address translation to a packet traversing the firewall, in one direction. The binat provides address translation to packets traversing the firewall, in two directions. The redirection rule, rdr, does not translate addresses, but does forward a particular packet to a particular host and/or port.

Rules

Pf rules take the following general form:

<action> <parameter>

Actions

block

Blocks the packet matching the rule. It may also be specified that the firewall will return a TCP RESET or ICMP UNREACHABLE packet to the source.

```
block return-rst in log on $internet_if proto tcp all  
block return-icmp in log on $internet_if proto udp all
```

pass

Allows the packet matching the rule. Passed TCP and UDP packets may also be entered into the state table, by using the "keep state" directive. TCP

connections may also have their states “modulated” with randomized sequence numbers.

```
pass out on le0 from 192.168.0.2 to 172.16.0.76
```

antispoof for <interface>

The antispoof directive is essentially a shorthand for defining rules to prevent spoofed packets from traversing the firewall.

For example, assume 192.168.0.5/24 is assigned to dc0:

```
antispoof for dc0
```

becomes

```
block in on ! dc0 inet from 192.168.0.5/24 to any
```

Additionally, non-loopback (lo) interfaces have another option, ‘inet.’ Once processed, the rule expands, blocking traffic with the specified interface’s address from entering the system.

```
antispoof for dc0 inet
```

becomes

```
block in on ! dc0 inet from 192.168.0.0/24 to any  
block in inet from 192.168.0.5 to any
```

Parameters

in | out

As the name implies, describes a packet either coming in or out of the firewall. An “in” or “out” is a required parameter. You must use separate statements to deal with both inbound and outbound packets.

```
pass in all  
block out all
```

log

Copy the matching packet to the pflog0 interface, in pcap format. Packet-sniffing tools, such as tcpdump, may then be used to analyze the packet.

```
block in log inet tcp from any to $web
```

log-all

When tracking state, copy the matching packet and all other packets in the established connection to pflog0.

```
pass in log-all inet tcp from any to $web keep state
```

quick

If a packet matches this rule, short-circuit the ruleset and take this action. No further rules are evaluated. Due to the fall-through (last matching rule wins) nature of pf, there are times when no further evaluation is required or desired.

```
block in quick on $dmz from any to any
```

on <interface>

Applies the rule to an interface. Only packets on this interface will be evaluated by this rule.

inet | inet6

Specifies whether rule deals with IPv4 or IPv6 packets. It is preferred to always specify either “inet” or “inet6,” though it is not necessary. OpenBSD and pf fully support the IPv6 protocol.

proto <protocol>

Specifies to which protocol type the rule applies. Values for protocol include tcp, udp, icmp, and ipv6-icmp. When “tcp” or “udp” are specified as values, ports may then be specified in the rule.

For example,

```
from <source address> [port <source port>] to \  
    <dest address> [port <dest port>]
```

Source and destination addresses can be host or network (using CIDR notation). Additionally, the keywords “any” and “no-route” may be used to signify any address and any nonroutable address, respectively.

Source and destination ports may be listed singly, or as ranges. When specifying ranges, the following operators may be used:

= (equal), != (unequal), < (lesser), <= (lesser or equal), > (greater),
>= (greater or equal), >< (range) and <> (except range)

For example, to specify X Windows ports 6000 to 6004, the parameter

```
port 5999 >< 6005
```

Expands to “ports above 5999 and below 6005”

user <user> group <group>

Pf can also evaluate packets based upon sockets owned by specific users or groups. The arguments may be either numeric or names. Effective, rather than real, user and group ID's are evaluated. For inbound connections, it is the owner of the port or socket listening on the destination port. For outbound connections, it is the user or group that initiated the communication.

For example, to only allow traffic to port 53/udp when `named` owns the process listening on port 53:

```
pass in on fxp0 proto udp from any to 192.168.0.1 \
port 53 user named group named
```

flags <a> | <a>/ | /

This rule is used in the evaluation of TCP packets, enabling the filtering of packets according to which flags are set. For obvious reasons, the “flags” parameter cannot be used with UDP or ICMP packets.

Valid flags are:

(A)CK	C(W)R
(E)CE	(F)IN
(P)USH	(R)ST
(S)YN	(U)RG

Packets may also be filtered according to the TCP flags set. This is very useful in filtering illegal packets, such as those used by port scanners like `nmap` (Xmas, FIN, etc).

For flags `<a>`, `<a>` must be the only flag set.

For flags `<a>/`, of the flags listed in ``, `<a>` is the only one allowed to be set.

For flags `/`, `` may not be set.

For example, `S/SA`, means that only SYN of SYN and ACK may be set. Any other flags specified are ignored. This is useful in tracking state, as you may specify to allow the first packet of a session to only have the SYN flag set:

```
pass in on le0 from any to 10.232.42.13 flags S \
```


`modulate state`

`icmp-type <type> code <code> and ipv6-icmp-type <type>
code <code>`

Specifies what types of ICMP or ICMPv6 packets to apply the rule to.

`allow-opts`

By default, any packets with IP options set are denied by the firewall due to the `scrub` directive, which performs packet normalization. Since the firewall cannot determine why a particular IP option is set, the packet is denied. This parameter is how to allow them through.

`label <string>`

The “label” parameter is used for keeping statistics in pf.

`block in on $internet all label “Default deny”`

The statistics are shown by running `pfctl -s labels`

`fastroute, route-to, dup-to`

Routing may be configured for packets matching rules with one of these options set. “Fastroute” acts normally, “route-to” forwards the packet to a specified interface, and “dup-to” sends a copy of the packet to the specified interface.

`keep state, modulate state`

These rules instruct the firewall to “keep state” of the allowed packet. Therefore, when the next packet in the established session enters the firewall, it will not be processed by the ruleset, but will be forwarded onto its destination. This greatly simplifies the ruleset, and requires the session to behave normally. The “modulate state” parameter instructs the firewall to randomize TCP sequences for greater protection, as well as keep state.

Example configuration

The following is simple examples of how pf may be configured in a common situation.

For a home DSL connection, a short, simple configuration is desirable, and the users are normally considered to be “trusted.” In this example, our user will be an IT professional who occasionally logs into his home network for use as an outside shell account.

```
# interfaces
$tunnel="tun0"
$outside_if="dc0"
$inside_if="dc1"

# nets
$home_net="192.168.0.0/24"
$work_net="192.0.2.0/27"

scrub in all

# translate my internal network to my external IP
nat on $tunnel from $home_net to any -> $tunnel

block in all
block out all
pass in quick on lo0 any
pass out quick on lo0 any

# how about ssh from work?
pass in on $tunnel from $work_net to $tunnel

# these have already been allowed...
pass out all proto tcp modulate state
pass out all proto udp keep state

# i trust myself ;^)
pass in on $inside_if all
```

Firewall Verification

Plan

GIAC Enterprises has requested that we verify that their firewall supports security policy and that it does not interfere with their business operations. During the planning phase of verification, we held several meetings with the network staff at GIAC Enterprises. As such, we have agreed to perform a preliminary, limited audit of the GIAC Enterprises firewall as a test. Should both parties be satisfied with the procedure, we will then perform a complete audit of the entire network.

Cost

The following table details the time and associated costs involved in the firewall audit.

Team Member	Hours	Rate	Amount
Lead Technical Auditor	60	\$175	\$10,500
Technical Auditor	40	\$125	\$5,000
Technical Writer	20	\$80	\$1,600
Total	120	-	\$17,100

Our validation will include the following steps:

- Information gathering
- Architecture review
- Network scanning

Information Gathering

As the first phase of our verification, we will hold several meetings with the networking staff at GIAC Enterprises. During these meetings, we will determine what differences exist in the documentation supplied to us and the actual implementation as it currently stands. With the help of the networking staff, we will piece together any gaps in our documentation. Also, we will get a small bit of face time to ease the anxiety of having outsiders present and given access to sensitive networks.

We will hold these meetings during normal business hours, as that is when the majority and most senior network administrators will be present.

We do not foresee any risk, with the possible exception that we may not receive accurate information from nervous employees who feel that anything we find would have negative results for them. However, GIAC Enterprises management has informed its employees that there will be no reprimands of any kind as a result of our work. Our largest risk, actually, is that the printed

documentation is so divergent from reality that our documentation process takes much longer than anticipated.

Additionally, we see little cost involved. Although there will be some lost productivity due to meetings, it is not out of the ordinary for much of the networking team to meet with vendors several times throughout the week. Therefore, we do not feel that our meetings will result in cost for GIAC Enterprises.

Architecture Review

Once we have a clear view of the configuration at GIAC Enterprises, we will examine the design and look for any potential problem areas. During this time, it will be helpful to be able to contact a knowledgeable staff member at GIAC Enterprises.

The review will be held during normal business hours, since there is no compelling reason to do it at another time. And, except for a possible papercut or two, we see absolutely no risk during the review. Finally, there is no risk involved. Everything during this time is performed by auditors.

Network Scanning

During this phase of our validation, we will be actively probing the GIAC Enterprises network. In doing so, we seek to verify that it is implementing the GIAC Enterprises security policy while allowing business operations to take place normally.

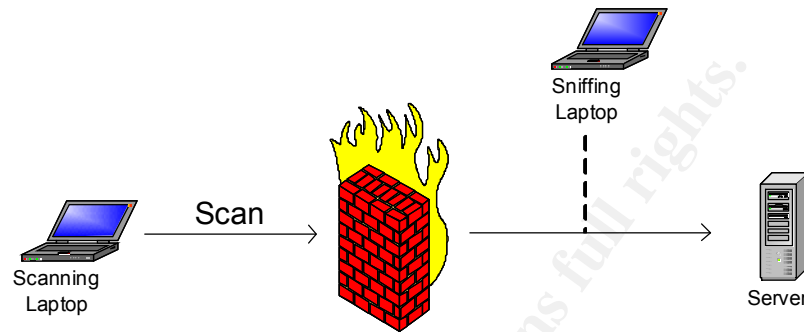
Since we will be actively probing the network and doing a few things that are out-of-spec during this validation, there is a very small, yet real, risk. However unlikely, there is a possibility that our scanning could cause a network device or server to crash or otherwise stop functioning normally.

Therefore, we will be perform our testing on Tuesday morning at 4:00am, the time GIAC Enterprises management has determined as the time when business traffic is at its lowest. Key IT staff will be present, in order to deal with any problems, should they arise. We will also utilize the “polite” setting in Nmap, which scans at a slower rate, so as to reduce the possibility of causing problems.

As we are testing the firewall itself, we are not testing other components at this time, including the border router, which also performs packet-filtering. To perform the audit, we will be using several tools to check whether the firewall is permitting the traffic it should, while blocking any other traffic. These tools are:

- nmap (<http://www.insecure.org>)
- tcpdump (<http://www.tcpdump.org>)
- PuTTY (<http://www.chiark.greenend.org.uk/~sgtatham/putty/>)
- Internet Explorer
(<http://www.microsoft.com/windows/ie/default.asp>)
- nslookup

As detailed in the following diagram, we will use two laptops. One laptop will generate scans directed at a host behind the firewall. The second laptop will sniff any traffic that comes through the firewall, verifying that the firewall is not allowing any extraneous traffic through. This is necessary, because portscan results do not necessarily imply that a firewall is or is not allowing traffic.



Our procedure is:

Nmap scans of firewall

- outside of firewall (between filtering router and firewall)
- internal segment
- screened segment

Nmap scans of screened subnet host

- from outside of firewall
- internal segment
- different screened subnet

Nmap scans of internal host

- from outside of firewall
- screened segment

DNS resolution

- from Internet
- from Intranet
- screened subnet

Email connectivity

- from Internet to DMZ mail relay
- from DMZ mail relay to internal Exchange server
- from internal Exchange server to DMZ mail relay

HTTP and HTTPS to corporate web servers

- from Internet
- from Intranet

SSH

- from Intranet

Database

- from web servers
- from Intranet

Results

Validation that the primary firewall is implementing security policy is determined by several factors. These factors include that each set of users is able to access resources as required by business needs, and that everything else is filtered by the firewall.

This has been demonstrated by our scans and tests, which follow.

Scans of Firewall

When scanning firewall itself, attacker should not be able to gain any information about the firewall through scanning.

TCP connect():

```
nmap -O -P0 -T 3 -v 192.0.2.254
```

All ports reported as closed. Attacker will not see filtered ports, which is indicative of some sort of packet filter or firewall. Appears to be an extremely boring host that isn't running anything at all.

UDP scan:

```
nmap -sU -O -P0 -T 3 -v 192.0.2.254
```

all ports reported as closed

Xmas scan:

```
nmap -sX -O -P0 -T 3 -v 192.0.2.254
```

The Xmas scan sends TCP packets with FIN, PUSH, and URG flags set. There is no legitimate reason for these flags to be set in the same packet. So-called because the packet is "lit up like a Christmas tree."

This scan did not go quite as we had expected. The duration of the scan was quite long, and the firewall did not log these packets as denied. Although the kernel simply ignored the packets as it should, it appears that they did pass

through the pf ruleset. We can add additional rules to filter invalid TCP flags, which is easily done.

ACK scan:

```
nmap -sA -O -P0 -T 3 -v 192.0.2.254
```

“Hi, I’m on the list, see?” This scan is an attempt to bypass packet filter-type firewalls, which are unable to keep state. Since the last sequence in a three-way TCP handshake is the “ACK,” a packet filter is typically configured to pass any TCP packet with this flag set.

Again, Nmap found no open ports, although it did believe that it had found all ports to be unfiltered, since the firewall is configured to respond with TCP RSTs.

A word on the Nmap fingerprint: although the fingerprint includes the phrase “i386-unknown-openbsd3.2” it does *not* mean that Nmap believes that this the machine it is scanning. It simply states that that was the type of system this particular build of Nmap was *compiled* on, so that when someone sends a fingerprint of a new device, it aids the author (Fyodor) in understanding it.

Fragmented SYN scan:

```
nmap -sS -O -v -T 3 -f 192.0.2.254
```

Nmap has the ability to send fragmented packets, attempting to bypass firewalls that do not have the capability, or are not configured to reassemble packets.

Screened Subnet scans

After verifying that the firewall is protecting itself, we have moved onto verifying that the firewall is protecting hosts within screened subnets.

To Webserver from Internet

TCP connect() scan:

```
nmap -sT -O -P0 -T 3 -v 192.0.2.1
```

TCP SYN scan:

```
nmap -sS -O -P0 -T 3 -v 192.0.2.1
```

UDP scan:

```
nmap -sU -O -P0 -T 3 -v 192.0.2.1
```

TCP XMAS scan:

```
nmap -sX -O -P0 -T 3 -v 192.0.2.1
```

Nmap’s TCP connect() scan found only port 80/tcp open on the webserver, as expected. However, this port was enough for Nmap to make a very good guess that the firewall was OpenBSD 3.0 on SPARC with “scrub in all” enabled. The SYN scan showed similar results.

The UDP and XMAS scans failed to discover any open ports or gather any OS information.

To Webserver from Intranet

TCP connect () scan:

```
nmap -sT -O -P0 -T 3 -v 10.30.10.1
```

TCP SYN scan:

```
nmap -sS -O -P0 -T 3 -v 10.30.10.1
```

UDP scan:

```
nmap -sU -O -P0 -T 3 -v 10.30.10.1
```

TCP XMAS scan:

```
nmap -sX -O -P0 -T 3 -v 10.30.10.1
```

As above, Nmap was able to find only port 80/tcp open using TCP connect() and SYN scans, and it correctly identified it as being protected by OpenBSD.

The UDP scan failed to discover any information. Unfortunately, the XMAS scan would not run, as this scan was being performed on a laptop running Windows 2000. After an initial surprise, it made sense, since XMAS scans are never successful against Windows platforms, because Windows does not conform to specifications.

To Webserver from another Segment

TCP connect () scan:

```
nmap -sT -O -P0 -T 3 -v 10.30.10.1
```

TCP SYN scan:

```
nmap -sS -O -P0 -T 3 -v 10.30.10.1
```

UDP scan:

```
nmap -sU -O -P0 -T 3 -v 10.30.10.1
```

TCP XMAS scan:

```
nmap -sX -O -P0 -T 3 -v 10.30.10.1
```

As expected, no scans were able to find anything.

User Access to Services

Most importantly, the firewall must allow GIAC Enterprises' employees and customers to access needed services and resources. Should the firewall fail this role, it has failed completely.

External DNS from Internet

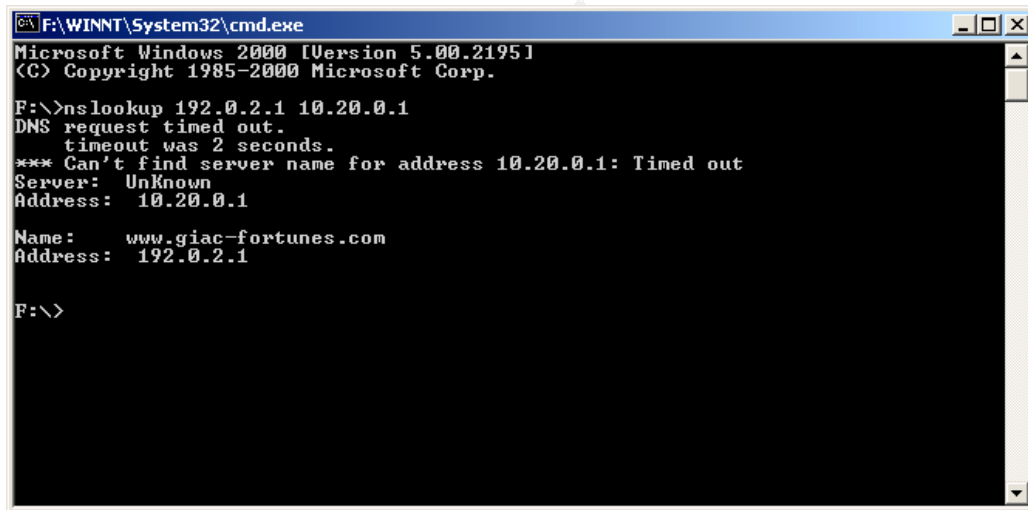
Customers and business partners must be able to query the external DNS so that they may then access the GIAC Enterprises websites, send email, and so on.

We have verified that Internet users are able to query the external DNS. Additionally, the external DNS can only show information about external services. Any DNS requests for internal services, even when the addresses are known, are sent to root nameservers, since they are truly not known by the server.

```
16:21:06.856461 10.20.0.1.53 > 198.32.64.12.53: 6111 PTR?  
50.254.10.10.in-addr.arpa. (43)
```

External DNS from Management stations

For testing purposes and troubleshooting, network and systems administrators periodically need access to the external DNS server. We have verified that this management station was able to query the external DNS successfully. Notice that the external DNS has no knowledge of the internal addressing of the GIAC Enterprises network, as it is unable to find its own, internal address:



```
F:\WINNT\System32\cmd.exe  
Microsoft Windows 2000 [Version 5.00.2195]  
(C) Copyright 1985-2000 Microsoft Corp.  
  
F:\>nslookup 192.0.2.1 10.20.0.1  
DNS request timed out.  
timeout was 2 seconds.  
*** Can't find server name for address 10.20.0.1: Timed out  
Server: Unknown  
Address: 10.20.0.1  
  
Name: www.giac-fortunes.com  
Address: 192.0.2.1  
  
F:\>
```

External DNS from screened subnets

The SMTP relay at GIAC Enterprises will need to query the external DNS to verify the validity of mail it receives, and it has been found to work as expected.

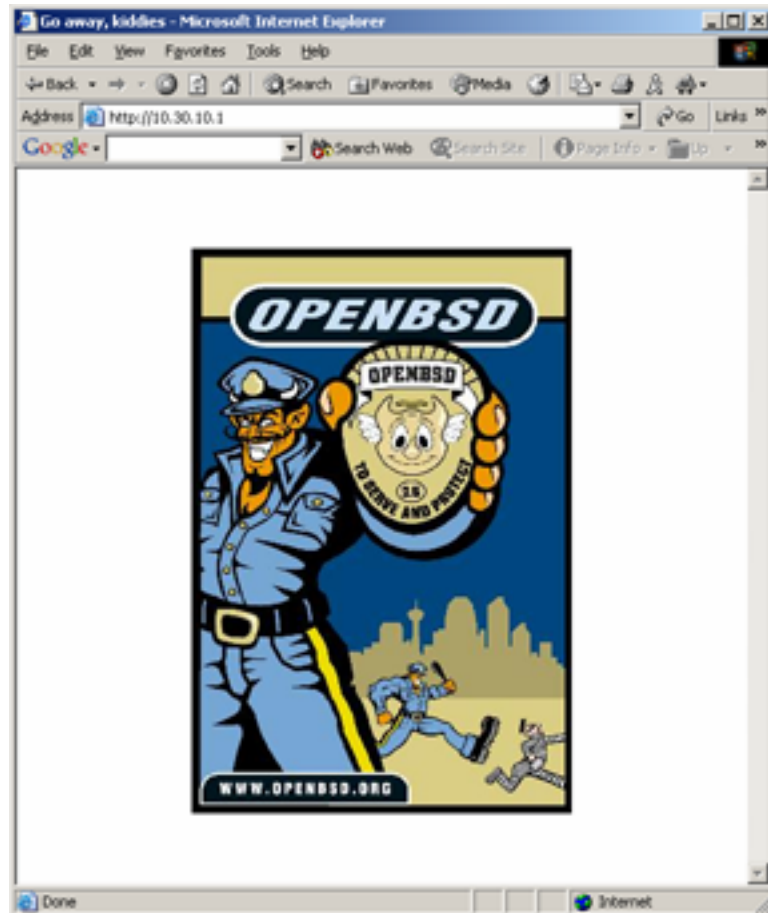
Email Connectivity

Since a mail relay has been implemented in a screened subnet, we will check to see that mail is able to be sent and received between the Internet and the mail relay, and between the relay and the corporate Exchange server.

Email connectivity is working.

HTTP and HTTPS to Corporate Web Services

Access to the corporate websites from both the Internet and Intranet work as expected. With the daemon cop on patrol, the GIAC Enterprises fortunes are well-guarded:

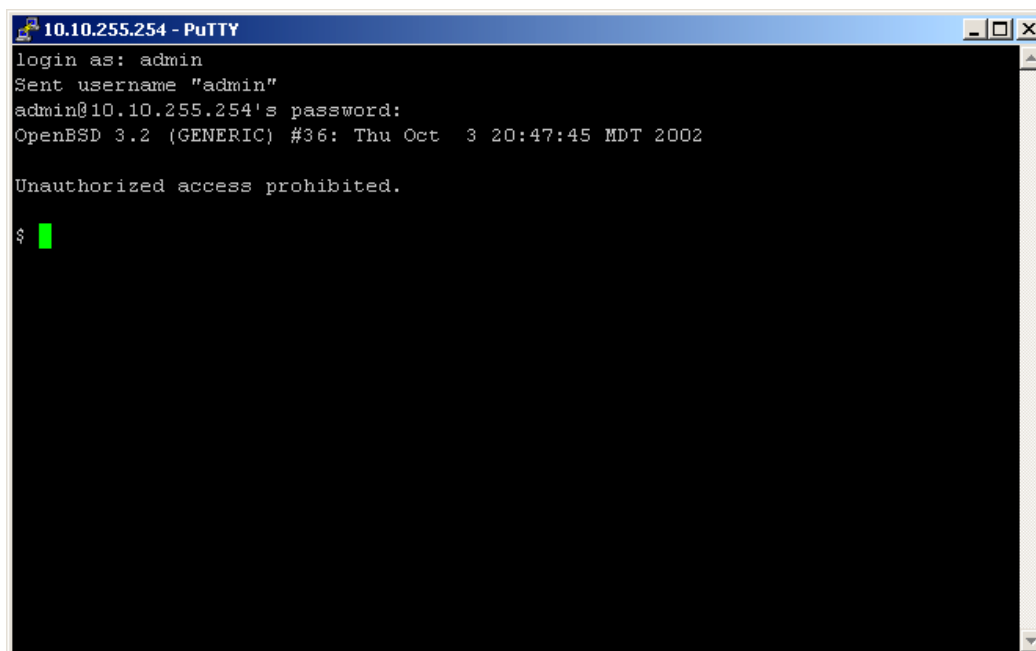


SSH from Intranet

Network and systems administrators at GIAC Enterprises use ssh to connect remotely to systems in the screened subnets. Therefore, we must verify that they are able to do so.

Connections from the management network within the intranet, 10.10.254.0/24, are allowed secure shell access, which is otherwise denied.

From the management network we see that users are able to use ssh:



```
10.10.255.254 - PuTTY
login as: admin
Sent username "admin"
admin@10.10.255.254's password:
OpenBSD 3.2 (GENERIC) #36: Thu Oct  3 20:47:45 MDT 2002

Unauthorized access prohibited.
$
```

However, from a different, internal network, access is prohibited:



Evaluation of audit

From this preliminary audit, it appears that the GIAC Enterprises firewall is functioning properly. Business access is not restricted, while unnecessary and unwanted access is denied. Since this preliminary audit was a success, we hope to continue working with GIAC Enterprises to provide a full-scale network audit.

Aside from the XMAS scans taking quite a bit longer than we expected, and our problems with Windows 2000, we feel that the audit went smoothly.

We do have some recommendations. One potential improvement that we do see would be the addition of a second firewall, that would handle all traffic between the internal network and the screened subnets. In the current design, if the firewall is compromised, the attacker potentially has access to all communications within the GIAC Enterprises network. This risk is somewhat mitigated by the fact that all sensitive information is encrypted before it goes through the firewall.

We also recommend the use of strong authentication through the corporate VPN. Password authentication is inherently weak, and there are better, stronger methods. We recommend the use of token devices, which may

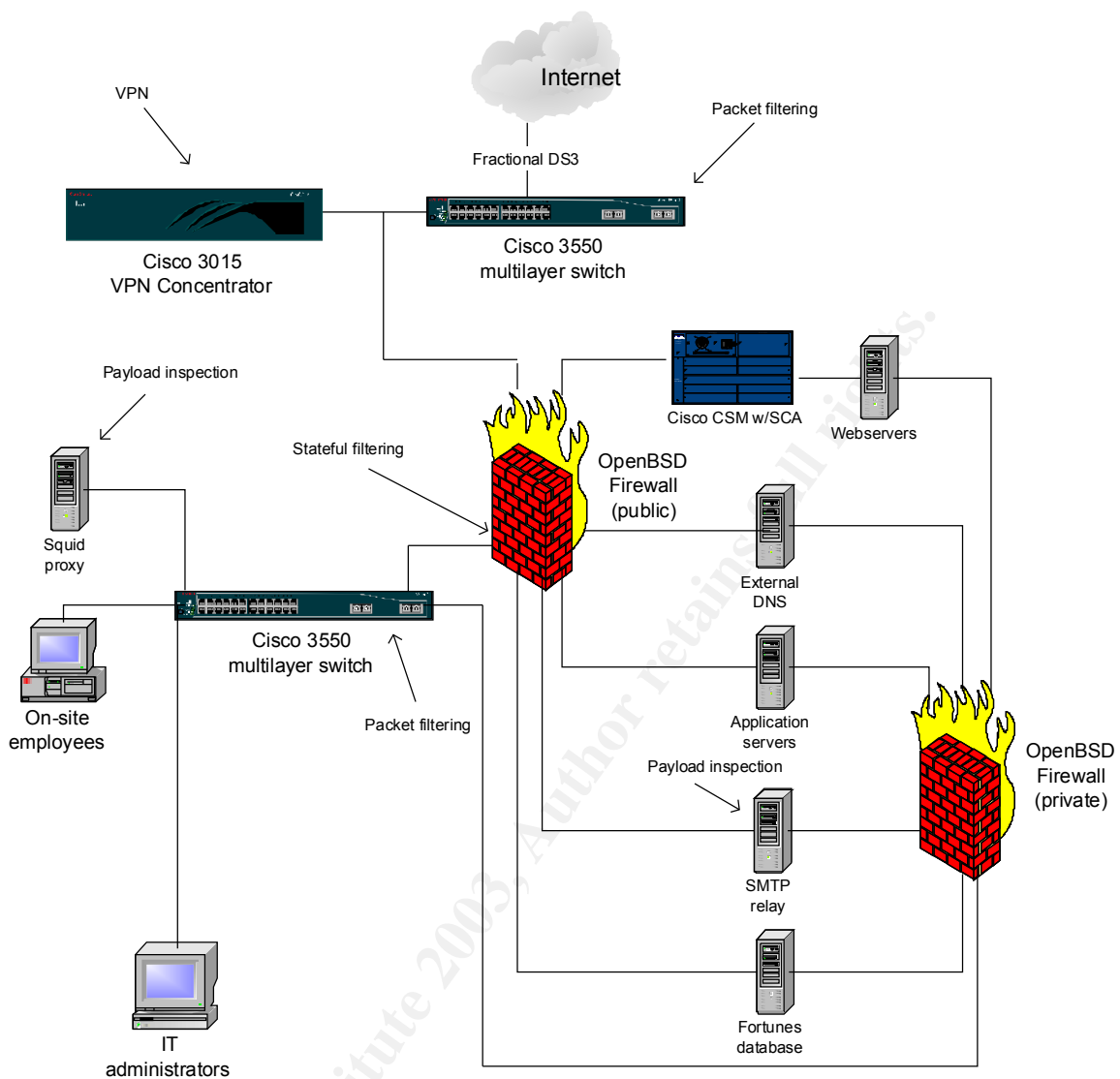
be carried on keychains by remote GIAC Enterprises employees. A token device holds authentication keys and digital certificates, which is much more secure than a username/password combination.

Although the GIAC Enterprises network has IDS sensors deployed on every network, these are not able to detect problems with encrypted traffic, for obvious reasons. An attack on a secure web server through HTTPS would not be noticed by the IDS system. To allow for this, a network device, such as the Cisco Secure Content Accelerator, could be deployed to offload the SSL/TLS encryption from the webserver to the network. As a result, the web traffic to the secure webserver would be cleartext, providing a performance increase for the server and allowing the IDS to examine the secure traffic.

Our final recommendation is the addition of a secondary link to the Internet. Since GIAC Enterprises depends heavily on its Internet services for revenue and marketing, we believe that a short outage could be damaging, and an extended outage doubly so.

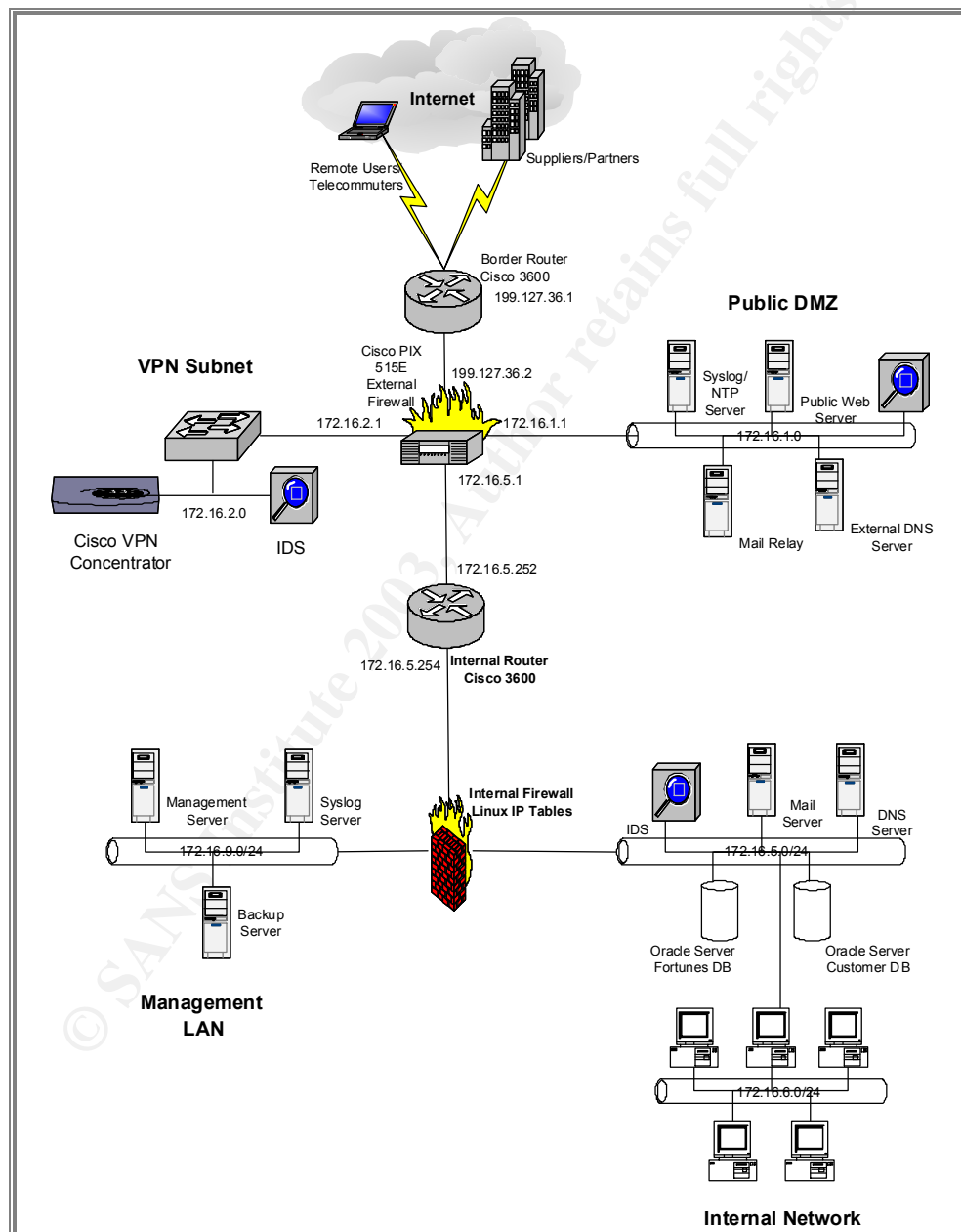
Nonetheless, based on our findings, we feel that the overall design and implementation of the perimeter security is sound.

© SANS Institute 2003, Author retains full rights.



Design Under Fire

We have chosen the network design described in the posted practical by Janice Robinson-Wells, available at http://www.giac.org/practical/Janice_Robinson-Wells_GCFW.doc. In this section, we will perform three attacks against this network: one against the firewall, one Denial-of-Service, and one attack through the perimeter.



Firewall Attack

Attack against firewall

Our first attack will be against the external firewall in this design. The external firewall is a Cisco PIX 515E running firmware version 6.2. The PIX is a robust, single-purpose appliance, and has a good security track record.

Our first step is to gather information on the PIX and any vulnerabilities or weaknesses that are known to be present in the 6.2 code. Unfortunately for us, there have not been many vulnerabilities disclosed about this firewall recently, and we do not have any contacts in the underground, so it does not appear that there are any canned “kiddie scripts” available for our use.

One promising lead is a vulnerability discussed on Cisco’s website, from June 27, 2002, entitled “Scanning for SSH Can Cause a Crash”¹⁶. It confirms that the 6.2 code is vulnerable to specially-made SSH packet. CERT has issued a Vulnerability Note (VU#945216¹⁷) regarding this, as it affects several SSH1 implementations, not just the Cisco PIX. Based on the information in CERT Incident Note IN-2001-12¹⁸, there appear to be exploit tools available in the underground, perhaps we can find one.

While looking for exploit code, we came across a good analysis¹⁹ of these tools by David Dittrich of the University of Washington. While the information is mostly irrelevant to our attack on the firewall, it does confirm that these tools do exist.

In the Neohapsis archives of the vuln-dev mailing list, we came across a posting where an H. D. Moore gave a URL²⁰ where he or she had a tarball of the exploit available for download. Moore stated that the file would only be available for 24 hours, and the post was made in March of 2002, however there was still a tarball available. We downloaded and unpacked the tarball, only to find that the actual exploit had been removed from the tarball. However, we did learn that the exploit was also known as ‘x2.’

After more searching on Google, we went back to one of our more useful sites that archives such things, Packet Storm²¹. Here, we played with search terms, and after a few tries, found our exploit with “ssh x2.” After downloading and unpacking the file, it appears that we have the real thing, although it is only a Linux binary with no source code. We also downloaded a targets file for the exploit.

After testing the exploit, it seems that the exploit may be for a different SSH exploit than we are looking for. Going back to the Packet Storm site, we ran

¹⁶ <http://www.cisco.com/warp/public/707/SSH-scanning.shtml#details>

¹⁷ <http://www.kb.cert.org/vuls/id/945216>

¹⁸ http://www.cert.org/incident_notes/IN-2001-12.html

¹⁹ <http://staff.washington.edu/dittrich/misc/ssh-analysis.txt>

²⁰ <http://www.digitaloffense.net/autossh.tgz>

²¹ <http://packetstorm.decepticons.org>

across a patch²² for OpenSSH 2.1.1 that enables it to exploit the crc32 attack detector vulnerability.

Therefore, we grabbed the “portable” version of OpenSSH 2.1.1²³, applied the patch, and compiled the binaries. We’re not programmers, and the patch is merely for demonstration of what one can do with the exploit, but it does connect to a remote host and attempt an overflow. For most hosts, we don’t think this would work, but since the PIX is vulnerable to the packet and not the payload, we think this tool has a good chance of crashing a vulnerable PIX.

Unfortunately, we do not have any spare PIX firewalls to test the exploit against. Therefore, our first test will be against the GIAC Enterprises 515E.

According to the patch instructions, the command should look something like:

```
./ssh -v -p 7777 localhost 2>&1 -c blowfish -T -l root
```

We will replace “7777” with the default “22” and “localhost” with the address of the GIAC Enterprises PIX. Also, since the PIX does not support the Blowfish algorithm, we will force use of 3DES.

```
./ssh -v -p 22 192.0.2.254 2>&1 -c 3des -T -l root
```

Unfortunately for us, the exploit was unsuccessful. We were unable to attack the GIAC Enterprises firewall. This is probably due to at least one of three factors, including a patched firewall, a firewall not listening on the SSH port, and a nonworking exploit.

Had our exploit worked as we had hoped, the PIX firewall would have consumed all of its CPU, trying to process the invalid data of the exploit. This would have prevented any traffic from traversing the firewall, keeping customers and partners from accessing the GIAC Enterprises site, as well as keeping GIAC Enterprises employees from accessing outside resources. This condition would have lasted until the PIX firewall was power-cycled.

This attack could have been prevented in several ways. The first and most basic, is a firewall running a current, nonvulnerable codebase. Although many network administrators (and management) are loath to patch firewalls, due to the downtime necessary, firewalls need to be patched and kept up-to-date just like any other host, perhaps even more so. The second way this attack could have been prevented was by the PIX not listening for SSH connections on its outside, public interface. Since this is not something that the PIX does by default, this is typically enabled by an administrator who is not normally on-site. Finally, applying the concept of defense-in-depth would have protected even a vulnerable PIX firewall. Since defense-in-depth entails multiple layers of overlapping security, a border router should have been configured not to allow outside connections to the firewall’s outside interface (or at least limit access to a

²² <http://packetstormsecurity.org/0102-exploits/ssh1.crc32.txt>

²³ <ftp://ftp.openbsd.org/pub/OpenSSH/portable/openssh-2.1.1p4.tgz>

limited set of IP addresses in the case of the previously-mentioned remote administrator).

Denial-of-Service

Denial-of-Service attacks differ from cracking (popularly known as “hacking”) in that they do not attack a host or a service in order to gain control. Rather, they typically flood a host or network with requests and/or traffic, wasting resources and thereby either causing the victim to crash or to run so slowly as to be useless.

Early Denial-of-Service (DoS) attacks simply pitted a more powerful system against another. If the system were powerful enough, or the service on the victim host was buggy, the victim would be knocked offline. These attacks typically consisted of floods of ICMP, UDP, or TCP SYN packets.

As the Internet evolved, so did DoS attacks. One clever technique was known as the “smurf attack,”²⁴ as it involved fooling numerous hosts into sending ICMP echo-replies to an unsuspecting victim. The attacker accomplished this by spoofing an ICMP echo-request to the broadcast address of a large network, using the victim’s IP address as the source. Following definition, the ICMP echo-request would be broadcast to every host within the network, and these hosts would then respond individually to what seemed a legitimate request, drowning the victim in a flood of ICMP. The attack was highly efficient: for each packet sent, it could potentially cause thousands of packets to descend upon its victim. Additionally, it was hard to pinpoint the culprit, as ICMP packets are easily forged. To deal with the problem, sites blocked broadcast ICMP, and router vendors changed the default behavior of traffic sent to the broadcast address of a network: the router would simply reply once and not forward the request.

Distributed Denial of Service (DDoS) takes this one step forward. By using a large number of compromised hosts, an attacker has much more bandwidth, is harder to block, and is harder to trace back to the attacker. With the advent of broadband at home, large numbers of unpatched and unadministered hosts, typically running Windows or Linux and with respectable bandwidth, are available for nefarious uses. Once compromised, these hosts, known as “zombies,” await instructions from a “master,” by listening on a port or set of ports or even by connecting to an IRC channel. Once they receive these instructions, they flood the victim with packets from dozens or even hundreds of directions at once.

We have fifty such systems at our disposal, and we have decided to use them to commit a DDoS on the GIAC Enterprises network. The tool we will use for our nefarious purposes is the infamous TFN2K, running on compromised Linux hosts, although we also considered its close cousins Trinoo and Stacheldraht.

Getting a hold of a copy of TN2K proved to be a surprisingly difficult task. Many security sites mention it, but apparently its reputation is a little too dark for

²⁴ Described in CERT Advisory CA-1998-01, “Smurf IP Denial-of-Service Attacks,” <http://www.cert.org/advisories/CA-1998-01.html>

most of the white hat sites. However, we were able to find a copy at the Packetstorm Security²⁵ site.

TFN2K is possibly the most sophisticated of all DDoS tools available. Not only does it have several options for attack, it also encrypts its traffic, hides its communications in ICMP Echo Replies, and lends itself to an extremely complex, multilayered command system when used with netcat. All of these features make it difficult to detect and difficult to follow the trail back to the attacked. As the author, Mixer, claims, "It is time for everyone to wake up, and realize the worst scenario that could happen to him if he does not care enough about security issues²⁶."

Once TF2K is compiled and our compromised systems are now running the server, we can now command them to do our bidding. With our TFN2K client, we have many options of what we can tell our zombies to do:

```
$ ./tfn
usage: ./tfn <options>
[-P protocol]    Protocol for server communication. Can be ICMP, UDP or TCP.
                  Uses a random protocol as default
[-D n]           Send out n bogus requests for each real one to decoy targets
[-S host/ip]     Specify your source IP. Randomly spoofed by default, you need
                  to use your real IP if you are behind spoof-filtering routers
[-f hostlist]    Filename containing a list of hosts with TFN servers to contact
[-h hostname]   To contact only a single host running a TFN server
[-i target string] Contains options/targets separated by '@', see below
[-p port]       A TCP destination port can be specified for SYN floods
<-c command ID> 0 - Halt all current floods on server(s) immediately
                  1 - Change IP antispoof-level (evade rfc2267 filtering)
                     usage: -i 0 (fully spoofed) to -i 3 (/24 host bytes spoofed)
                  2 - Change Packet size, usage: -i <packet size in bytes>
                  3 - Bind root shell to a port, usage: -i <remote port>
                  4 - UDP flood, usage: -i victim@victim2@victim3@...
                  5 - TCP/SYN flood, usage: -i victim@... [-p destination port]
                  6 - ICMP/PING flood, usage: -i victim@...
                  7 - ICMP/SMURF flood, usage: -i victim@broadcast@broadcast2@...
                  8 - MIX flood (UDP/TCP/ICMP interchanged), usage: -i victim@...
                  9 - TARGA3 flood (IP stack penetration), usage: -i victim@...
                 10 - Blindly execute remote shell command, usage -i command
```

To reduce the chance of our attack getting filtered, we will use the TCP/SYN attack on port 80. With a list of our zombies in the file 'zombie-roo,' we launch our attack:

```
bash$ ./tfn -f zombie-roo -P icmp -c 5 -p 80 -i
www.giac-fortunes.com
```

The attack is now under way. Fifty broadband systems are now sending bandwidth to fill up the GIAC Enterprises T-1 line. Is it enough to fill up the T-1? Assuming an extremely low average of 128kbps per host, we have 6.4Mbps of traffic attacking the victim, which is several multiples of the 1.54Mbps available bandwidth. Victory!

Most countermeasures against DDoS attacks are actually aimed at reducing the number base of hosts and networks that can be compromised and

²⁵ <http://packetstorm.decepticons.org/distributed/>

²⁶ README file, TFN2K source code.

used in these attacks. The SANS Institute provides a guideline²⁷ for preventing a network from being used as a source or intermediary in such an attack. Unfortunately, as with so many other things in network security, it only takes a few unwatched networks to cause a lot of headaches. In this example, these fifty hosts could have been located anywhere in the world.

Recommendations for GIAC Enterprises specifically? Get more bandwidth and multiple pipes, if that is an economical option. Since our attack was aimed at port 80 on its main webserver, it would not be possible for GIAC Enterprises or its ISP to simply filter this traffic. To do so would be a complete denial-of-service. Ed Skoudis, of Security Strategy for Predictive Systems, argues that “adequate bandwidth, redundant paths through multiple ISPs, and traffic shaping tools are a must,”²⁸ when attempting to defend a network from DDoS attack. Skoudis also recommends having a very responsive ISP that can perform filtering. Also, it may be possible to have redundant services provided in separate geographic areas, although this is normally only an option for larger operations.

Additionally, the use of monitoring systems may aid in speeding the response to such an attack. BB4 Technologies’ Big Brother²⁹ system can be deployed easily to monitor vital systems and configured to alert operations staff to any potential problems. The software is easily configured and, as an added bonus, is available for free. A bandwidth monitor, such as the Network Performance Monitor³⁰ toolset of SolarWinds, can monitor an Internet pipe (or other network link) and be configured to sound an alarm, such as emailing an alert to a pager, when network resources surpass a threshold. Although these tools cannot prevent such an attack, they can provide an early warning system, enabling an organization to respond quickly and to contact network providers upstream for support. As always, a formalized contingency plan can expedite this process.

Internal Compromise Through Perimeter

Our third and final attack will be to compromise an internal system through the perimeter. Since attacking an internal system, that is, a system on the internal segment of the network, involves attacking a system on the perimeter, it tends to be somewhat more involved than a DoS attack. Typically, hosts on the perimeter are firewalled, with only the absolute necessary ports accessible by the outside world. Therefore, compromise of these systems is typically accomplished through exploiting a vulnerability in a service. Many of these services run with elevated privileges, such as “root” in Unix or “Administrator” in Windows. If compromised, significant or complete control of the system may be obtained. The compromised system can possibly then be leveraged to attack one or more systems located on the internal network.

²⁷ http://www.sans.org/rr/threats/dos_attacks.php

²⁸ *Counter Hack*, p396

²⁹ <http://bb4.com/>

³⁰ http://www.solarwinds.net/Tools/Network_Monitoring/Network_Perf_Monitor/index.htm

After a session of brainstorming, we have come up with a general plan for such an attack.

The first step of our plan is to compromise the SMTP mail relay. It is an ideal candidate for an attack, since it is accessible from anywhere on the Internet, and it also is allowed by the GIAC Enterprises firewall to initiate communications with an internal host. This is because this host, due to its function, must communicate with any external mail server as well as GIAC Enterprises' internal mail server.

Due to the staggering size of its installed base - whether its total percentage of mail servers the 75% of claimed by Sendmail, Inc. or the 42% claimed by gmail creator D. J. Bernstein³¹ - Sendmail has long been a favorite target of hackers. The fact that many of these installations are attractive Unix or Linux hosts that can provide an attacker with many resources only adds to the attractiveness of such systems.

Should our attack on the SMTP mail relay be successful, we would then use it to mount a similar attack on the internal mail server. Although Robinson-Wells does not specify the MTA on the internal mail server, there is a strong possibility that it is also running Sendmail. We know it is not running Exchange 2000, since the "fixup smtp" feature which is enabled on the PIX firewall "may cause...Exchange servers to function unpredictably"³², according to Cisco. If the internal mail server is in fact running Sendmail, chances are quite good that these systems are running the same patchlevel, since it follows that if an admin is conscientious enough to patch one system, he or she will most likely patch all vulnerable systems. The opposite is also a distinct possibility.

According to Robinson-Wells, GIAC Enterprises runs Sendmail 8.12.6 on RedHat Linux 7.3 servers. Recently, a vulnerability in all versions of Sendmail prior to 8.12.8 was made public³³. Additionally, proof-of-concept exploit code³⁴ was posted by a Polish security team, Last Stage of Delirium Research Group. The group believed that the vulnerability to be potentially exploitable on Linux. In particular, they were able to get Sendmail 8.11.6 on Slackware 8 to execute their own code.

The exploit works by sending an email, which contains the exploit, to a vulnerable Sendmail server. The server becomes compromised as it processes the contents of the message, leading to a buffer overflow. Protecting against an intruder leveraging this problem is extremely difficult, as it cannot be easily blocked by a network device, such as a packet-filtering router or stateful firewall. Additionally, a successful exploit generates no log messages, making it difficult to detect an intrusion.

³¹ <http://cr.yip.to/surveys/smtpsoftware6.txt>

³² http://www.cisco.com/en/US/products/sw/secursw/ps2120/products_configuration_guide_chapter_09186a00800eb727.html#1063757

³³ Described in CERT Advisory CA-2003-07, "Remote Buffer Overflow in Sendmail".

<http://www.cert.org/advisories/CA-2003-07.html>

³⁴ http://www.security.nnov.ru/files/linux86_sendmail.c

Although we have almost no programming skill, it may be possible (however unlikely) for us to replace the sample exploit code with a Linux program that would allow us to gain a root login. A skilled coder may be able to wreak substantial havoc, given such a vulnerability.

After perusing a guide³⁵ on creating shellcode by Murat Balaban, we have decided to ungraciously steal his simple shellcode, named “sc.c” and use it to replace the code supplied in the Sendmail sample exploit.

Once compiled, the “improved” exploit gives us several options:

```
# ./linx86_sendmail
copyright LAST STAGE OF DELIRIUM mar 2003 poland //lsd-pl.net/
sendmail 8.11.6 for Slackware 8.0 x86

usage: ./linx86_sendmail target [-l localaddr] [-b localport] [-p ptr]
[-c count] [-t timeout] [-v 80]
```

We will try it against the GIAC Enterprises external SMTP relay, and see what happens.

```
# ./linx86_sendmail eos.ums1.edu -c 10 -p 0xbfff9f1c -v 80
copyright LAST STAGE OF DELIRIUM mar 2003 poland //lsd-pl.net/
sendmail 8.11.6 for Slackware 8.0 x86
```

```
.....
.....
.....
```

The attack appears to continue on indefinitely; we don’t believe that it was successful. A quick telnet to port 25 on the SMTP relay indicates that GIAC Enterprises has patched up to the latest version of Sendmail:

```
# telnet mail.giac-fortunes.com 25
Trying 1.2.3.4...
Connected to mail.giac-fortunes.com.
Escape character is '^]'.
220 mail.giac-fortunes.com ESMTP Sendmail 8.12.8/8.12.8; Tue, 25 Mar
2003 23:18:08 -0600 (CST)
```

Although this attack was unsuccessful, the concept demonstrates the fact that, even with sound network design and firewall implementation, an internal compromise is still possible. This is due to the fact that some DMZ systems must be allowed to initiate communication with certain, internal hosts, and therefore can be used as a vector into the internal network. This becomes especially apparent when the same services are provided in the DMZ as in the internal network, as they may both be vulnerable to the same attack.

There are several possibilities to explore to combat this risk. The first would be to use different software and hardware platforms in the internal network

³⁵ “Designing Shellcode Demystified”. <http://www.enderunix.org/docs/en/sc-en.txt>

and DMZ. A good example of this would be to use, say, Postfix on Linux as the external MTA, and Sendmail on Solaris as the internal MTA. Unfortunately, this can lead to additional risks, as the load of configuring and supporting additional services and platforms may lead to misconfigurations and vulnerabilities. Another possibility would be to use a specially-hardened system in the DMZ. Cisco

```
/// propolice http://www.trl.ibm.com/projects/security/ssp/  
/// okena stormwatch  
http://www.okena.com/areas/products/products\_stormwatch.html
```

In conclusion, we have found that a good, segmented firewall design and patched systems makes compromising systems and data extremely difficult.

```
// detail detail detail - they killed me on this one  
// tools and methods to compromise Sendmail
```

Recommended protections against an attack such as this include meticulously applying security patches, employing an IDS that can automatically respond to threats as they develop, and additional layers of security.

© SANS Institute 2003, Author retains full rights.

Appendix A: OpenBSD installation and hardening

OpenBSD can be installed several ways, the most popular being from CDROM or by anonymous FTP. Since it will be desirable to have the media on hand when performing installs, GIAC Enterprises has opted to purchase several copies of the official OpenBSD distribution³⁶.

Installation of the OS is straightforward and should cause no problems for administrators with prior Unix administration. GIAC Enterprises has decided to run OpenBSD on Intel x86 hardware.

Although the following installation was performed on a Sparc-based Axil 245 in our test lab, the installation process is no different for an x86 machine. For security, the system is not attached to a network during installation. Although the OpenBSD install and setup program is as security-conscious as the rest of the system and does not enable services, we feel that it is good practice to install systems unattached to networks.

Booting from the OpenBSD CDROM:

```
>> OpenBSD BOOT 2.2
Booting 3.2/sparc/bsd.rd @ 0x4000
3359728+182416
console is ttya
Copyright (c) 1982, 1986, 1989, 1991, 1993
    The Regents of the University of California. All rights
reserved.
Copyright (c) 1995-2002 OpenBSD. All rights reserved.
http://www.OpenBSD.org
OpenBSD 3.2 (RAMDISK) #153: Thu Oct  3 21:13:51 MDT 2002
deraadt@sparc.openbsd.org:/usr/src/sys/arch/sparc/compile/RAMDISK
real mem = 66727936
avail mem = 58183680
using 200 buffers containing 3334144 bytes of memory
bootpath:
/iommu@0,10000000/sbus@0,10001000/espdma@4,8400000/esp@4,8800000/sd@6,0
:d
mainbus0 (root): SUNW,Axil-245
cpu0 at mainbus0: MB86904 @ 85 MHz, on-chip FPU
cpu0: 16K instruction (32 b/l), 8K data (16 b/l) cache enabled
obio0 at mainbus0
clock0 at obio0 addr 0x71200000: mk48t08 (eeprom)
timer0 at obio0 addr 0x71d00000 delay constant 40
zs0 at obio0 addr 0x71100000 pri 12, softpri 6
zstty0 at zs0 channel 0 (console i/o)
zstty1 at zs0 channel 1
zs1 at obio0 addr 0x71000000 pri 12, softpri 6
zskbd0 at zs1 channel 0: keyboard, type 0
wskbd0 at zskbd0
zstty2 at zs1 channel 1: mouse
slavioconfig at obio0 addr 0x71800000 not configured
auxreg0 at obio0 addr 0x71900000
```

³⁶ Orders can be made online at <http://www.openbsd.org/orders.html>

```

power0 at obio0 addr 0x71910000
fdc0 at obio0 addr 0x71400000 pri 11, softpri 4: chip 82077
fd0 at fdc0 drive 0: 1.44MB 80 cyl, 2 head, 18 sec
iommu0 at mainbus0 addr 0x10000000: version 0x4/0x0, page-size 4096,
range 64MB
sbus0 at iommu0: clock = 21.250 MHz
dma0 at sbus0 slot 4 offset 0x8400000: rev 2
esp0 at dma0 offset 0x8800000 pri 4: ESP200, 40MHz, SCSI ID 7
scsibus0 at esp0: 8 targets
sd0 at scsibus0 targ 0 lun 0: <SEAGATE, ST15230N, 0638> SCSI2 0/direct
fixed
sd0: 4095MB, 3992 cyl, 19 head, 110 sec, 512 bytes/sec, 8386733 sec
total
cd0 at scsibus0 targ 6 lun 0: <SONY, CDU 561 SUNMSCD, 1.9k> SCSI2
5/cdrom removable
SUNW,bpp at sbus0 slot 4 offset 0xc800000 not configured
audio at sbus0 slot 4 offset 0x1300000 not configured
ledma0 at sbus0 slot 4 offset 0x8400010: rev 2
le0 at ledma0 offset 0x8c00000 pri 6: address 00:00:3b:80:2b:d7
le0: 16 receive buffers, 4 transmit buffers
hme0 at sbus0 slot 0 offset 0x8c00000 pri 7: address 08:00:20:93:75:b8
rev 34
qsphy0 at hme0 phy 1: QS6612 10/100 media interface, rev. 1
hme1 at sbus0 slot 0 offset 0x8c10000 pri 7: address 08:00:20:93:75:b9
rev 34
qsphy1 at hme1 phy 1: QS6612 10/100 media interface, rev. 1
hme2 at sbus0 slot 0 offset 0x8c20000 pri 7: address 08:00:20:93:75:ba
rev 34
qsphy2 at hme2 phy 1: QS6612 10/100 media interface, rev. 1
hme3 at sbus0 slot 0 offset 0x8c30000 pri 7: address 08:00:20:93:75:bb
rev 34
qsphy3 at hme3 phy 1: QS6612 10/100 media interface, rev. 1
SUNW,DBRIe at sbus0 slot 2 offset 0x10000 not configured
cgsix0 at sbus0 slot 3 offset 0x0: , 1152x900, rev 11
wsdisplay0 at cgsix0
wsdisplay0: screen 0 added (std, sun emulation)
root on rd0a
rd0: fixed, 4352 blocks
rootdev=0x1100 rrootdev=0x6a00 rawdev=0x6a02
WARNING: clock gained 127 days -- CHECK AND RESET THE DATE!
erase ^?, werase ^W, kill ^U, intr ^C, status ^T

```

(I)nstall or (S)hell? i

Welcome to the OpenBSD/sparc 3.2 install program.

This program will help you install OpenBSD in a simple and rational way. At ny prompt except password prompts you can run a shell command by typing '!foo', or escape to a shell by typing '!'. Default answers are shown in []'s and are selected by pressing RETURN. At any time you can exit this program by pressing Control-C and then RETURN, but quitting during an install can leave your system in an inconsistent state.

Specify terminal type: [sun] vt100

IS YOUR DATA BACKED UP? As with anything that modifies disk contents, this program can cause SIGNIFICANT data loss.

It is often helpful to have the installation notes handy. For complex disk configurations, relevant disk hardware manuals and a calculator are useful.

Proceed with install? [n] y

Cool! Let's get to it...

You will now initialize the disk(s) that OpenBSD will use. To enable all available security features you should configure the disk(s) to allow the creation of separate filesystems for /, /tmp, /var, /usr, and /home.

The installer now allows us to partition the system's drive(s). Users coming from the Microsoft or Linux world may want to run through this part of the install a few times, as it is a bit strange when first used. Don't forget to add a swap partition!

Available disks are: sd0.

Which one is the root disk? (or done) [sd0]

This platform requires that partition offsets/sizes be on cylinder boundaries.

Partition offsets/sizes will be rounded to the nearest cylinder automatically.

Initial label editor (enter '?' for help at any prompt)

> ?

Available commands:

- p [unit] - print label.
- M - show entire OpenBSD man page for disklabel.
- e - edit drive parameters.
- a [part] - add new partition.
- b - set OpenBSD disk boundaries.
- c [part] - change partition size.
- d [part] - delete partition.
- D - set label to default.
- g [d|b] - Use [d]isk or [b]ios geometry.
- m [part] - modify existing partition.
- n [part] - set the mount point for a partition.
- r - recalculate free space.
- u - undo last change.
- s [path] - save label to file.
- w - write label to disk.
- q - quit and save changes.
- x - exit without saving changes.
- X - toggle expert mode.
- z - zero out partition table.
- ? [cmdnd] - this message or command specific help.

Numeric parameters may use suffixes to indicate units:

'b' for bytes, 'c' for cylinders, 'k' for kilobytes, 'm' for

megabytes, 'g' for gigabytes or no suffix for sectors (usually 512 bytes).

Non-sector units will be rounded to the nearest cylinder.
Entering '?' at most prompts will give you (simple) context sensitive help.

```
> a a
offset: [0]
size: [8386733] 8000000
Rounding to nearest cylinder: 8000520
FS type: [4.2BSD]
mount point: [none] /
> a b
offset: [8000520]
size: [386213]
Rounding to nearest cylinder: 384560
FS type: [swap]
> p
device: /dev/rsd0c
type: SCSI
disk: SCSI disk
label: ST15230N
bytes/sector: 512
sectors/track: 110
tracks/cylinder: 19
sectors/cylinder: 2090
cylinders: 3992
total sectors: 8386733
free sectors: 1653
rpm: 3600

3 partitions:
#      size  offset  fstype  [fsize bsize  cpg]
a:  8000520      0   4.2BSD   1024  8192    16 # /
b:   384560 8000520    swap
c:  8386733      0   unused          0      0
> q
Write new label?: [y] y
The root filesystem will be mounted on sd0a.
sd0b will be used for swap space.
Done - no available disks found.
```

You have configured the following partitions and mount points:
sd0a /

The next step creates a filesystem on each partition, ERASING existing data.

Are you really sure that you're ready to proceed? [n] y
/dev/rsd0a: 8000520 sectors in 3828 cylinders of 19 tracks, 110 sectors
3906.5MB in 240 cyl groups (16 c/g, 16.33MB/g, 3968 i/g)
/dev/sd0a on /mnt type ffs (rw, asynchronous, local, ctime=Sat Feb 8 23:04:29 2003)

Enter system hostname (short form, e.g. 'foo'): danio
Configure the network? [y] n
Password for root account (will not echo):

Password (again):

You will now specify the location and names of the install sets you want to load. You will be able to repeat this step until all of your sets have been successfully loaded. If you are not sure what sets to install, refer to the installation notes for details on the contents of each.

Sets can be located on a (m)ounted filesystem; a (c)drom, (d)isk or (t)ape device; or a (f)tp, (n)fs or (h)ttp server.

Where are the install sets you want to use? (m, c, f, etc.) c

Available CD-ROMs are: cd0.

Which one contains the install media? (or done) [cd0]

Enter the pathname where the sets are stored (or '?') [3.2/sparc]

The following sets are available. Enter a filename, 'all' to select all the sets, or 'done'. You may de-select a set by prepending a '-' to its name.

```
[X] base32.tgz
[X] etc32.tgz
[X] misc32.tgz
[X] comp32.tgz
[X] man32.tgz
[X] game32.tgz
[ ] xbase32.tgz
[ ] xshare32.tgz
[ ] xfont32.tgz
[ ] xserv32.tgz
[X] bsd
```

For some inexplicable reason, the BSD games collection (which includes fortune, snake, etc.) is included by default in the OpenBSD base install. At best they are a waste of disk space and at worst, a potential liability. Since we certainly will not be using these, we will remove the set from our installation.

For much the same reason, we will not be installing any of the X Windows packages. If this system were a workstation, installation of the X packages would of course be recommended.

File Name? (or 'done') [xbase32.tgz] -game32.tgz

The following sets are available. Enter a filename, 'all' to select all the sets, or 'done'. You may de-select a set by prepending a '-' to its name.

```
[X] base32.tgz
[X] etc32.tgz
[X] misc32.tgz
[X] comp32.tgz
[X] man32.tgz
[ ] game32.tgz
[ ] xbase32.tgz
[ ] xshare32.tgz
```

```

[ ] xfont32.tgz
[ ] xserv32.tgz
[X] bsd

File Name? (or 'done') [game32.tgz] done
Ready to install sets? [y]

Getting base32.tgz ...
100% |*****| 24255 KB
03:52
Getting etc32.tgz ...
100% |*****| 19880 KB
03:23
Getting man32.tgz ...
100% |*****| 5659 KB
01:04
Getting bsd ...
100% |*****| 2620 KB
00:13
Extract more sets? [n] n
Saving configuration files.....done.
Generating initial host.random file .....done.
What timezone are you in? ('?' for list) [US/Pacific] US/Central
You have selected timezone 'US/Central'.
Making all device nodes...done.
Installing boot block...
boot: /mnt/boot
proto: /mnt/usr/mdec/bootxx
device: /dev/rsd0c
architecture: sun4c
/mnt/usr/mdec/bootxx: entry point 0x340000
/mnt/usr/mdec/bootxx: a.out header left on
proto bootblock size 7680
room for 256 filesystem blocks at 0x341810
/mnt/boot: block numbers: 6432 6448 6464 6480 6496 6512 6528 1068

CONGRATULATIONS! Your OpenBSD install has been successfully completed!
To boot the new system, enter halt at the command prompt. Once the

system has halted, reset the machine and boot from the disk.
# halt
syncing disks... done
halted

```

Now that the system is installed, it's time to reboot and perform post-installation configuration.

```

OpenBSD 3.2 (GENERIC) #36: Thu Oct  3 20:47:45 MDT 2002
deraadt@sparc.openbsd.org:/usr/src/sys/arch/sparc/compile/GENERIC
real mem = 66727936
avail mem = 58773504
.
.
.
.

```

Booting the system for the first time will result in the generation of several cryptographic keys:

```
ssh-keygen: generating new DSA host key... done.  
ssh-keygen: generating new RSA host key... done.  
ssh-keygen: generating new RSA1 host key... done.
```

Once completed, the bootup procedure continues:

```
starting network daemons: sendmail inetd sshd.  
starting local daemons:.  
standard daemons: cron.  
Sat Feb  8 17:54:50 CST 2003
```

```
OpenBSD/sparc (danio.my.domain) (console)
```

```
login: root  
Password:  
Feb  8 17:55:34 danio login: ROOT LOGIN (root) ON console  
Feb  8 17:55:34 danio login: ROOT LOGIN (root) ON console  
OpenBSD 3.2 (GENERIC) #36: Thu Oct  3 20:47:45 MDT 2002
```

Welcome to OpenBSD: The proactively secure Unix-like operating system.

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest version of the code. With bug reports, please try to ensure that enough information to reproduce the problem is enclosed, and if a known fix for it exists, include that as well.

You have mail.
Terminal type? [sun] vt100

Don't login as root, use su
danio#

The system is now ready. For newcomers to OpenBSD, there is a manpage that helps familiarize them with the system, afterboot(8)³⁷. Before going any further, it is helpful to read this document, as it truly makes the transition from another Unix-like system easier.

Hardening

The system is now installed, and is ready for hardening. Hardening will consist of disabling unnecessary services, configuring essential services, restricting access to essential services, and patching.

Like in any good, BSD system, upon bootup, the rc script checks the /etc/rc.conf file, and configures the system according to the settings there. This

³⁷ <http://www.openbsd.org/cgi-bin/man.cgi?query=afterboot&apropos=0&sektion=0&manpath=OpenBSD+3.2&arch=i386&format=html>

differs from the SysV-style scripts familiar to RedHat Linux and Sun Solaris administrators. Once recovered from the shock of the absence of runlevels, these administrators should adapt quite easily to editing a single file to control services started at boot. Settings in the rc.conf file determine which daemons are started at boot-time, as well as what options to pass them. As with any general-purpose OS, OpenBSD contains programs and services not required for normal firewall operation. Keeping with good security practice, we will disable all such services, and limit access to those that are still required.

Disabling unnecessary daemons

Some daemons that are set to run by default in rc.conf include inetd, sendmail, syslogd, and sshd. We will disable inetd and sendmail, as a firewall really should not be running these services:

```
inetd=NO
sendmail_flags=NO
```

Securing necessary services

Syslogd and sshd, however, will be needed for logging and administrative access. Therefore, they should be examined for possible hardening.

Syslogd

Although OpenBSD's syslogd rather annoyingly appears to be listening on a port despite a system administrator's best efforts, it is in fact not listening by default. It would seem that the developers decided to replace one, real headache with a fake one! Syslogd may be configured normally through /etc/sysylog.conf.

OpenSSH

Since OpenSSH is a "side project" of the OpenBSD team, it is not surprising that OpenBSD includes OpenSSH. OpenSSH is an open source suite containing an SSH server, sshd, as well as related clients, including ssh and sftp. OpenSSH has a large installed base and has a good track record for security. The OpenSSH server also interoperates with ssh clients from other vendors, including SSH Communications, PuTTY, MacSSH, and others.

The OpenSSH server, sshd, reads its configuration upon startup from /etc/ssh/sshd_config. Modifying this file allows us to increase the security of our firewall.

By default, OpenSSH supports multiple versions of the SSH protocol, including SSH 1, 1.5, and 2. Since SSHv1 is vulnerable to an insertion attack³⁸, we will only allow connections using SSHv2.

³⁸ Detailed at <http://www.corest.com/common/showdoc.php?idxseccion=10&idx=131>

Protocol 2

Since all legitimate connections to the firewall will be initiated from the internal segment of the GIAC Enterprises network, the daemon should only listen on that interface:

```
ListenAddress      10.255.255.254
```

The included `sshd` is compiled with support for Wietsa Venema's TCP Wrappers³⁹. As such, access control is configured via the `/etc/hosts.allow` and `/etc/hosts.deny` files. Access control will also be enforced through packet filtering for a layered approach.

```
/etc/hosts.deny
```

```
ALL:ALL
```

```
/etc/hosts.allow
```

```
sshd:10.10.254.
```

Configuring logging

Keeping correct time

Since having a correct timestamp is essential to good logging, we will configure our firewall to periodically synchronize itself with an NTP server on our internal network. OpenBSD's `rdate(8)`⁴⁰ utility, included in the base system, supports NTP and clock skewing:

```
danio# crontab -e

58 * * * * /usr/sbin/rdate -nca time.server | logger \
    -t rdate
```

Logging packets

When `pf` logs a packet, it does so literally – it copies the packet to a virtual network interface named `pflog0` in `pcap` format. Logged packets may be viewed in real-time with `tcpdump`:

```
danio# ifconfig pflog0 up
danio# tcpdump -e -n -tttt -i pflog0 -s 96 -X
```

³⁹ Available for download at [ftp://ftp.porcupine.org/pub/security/index.html](http://ftp.porcupine.org/pub/security/index.html).

⁴⁰ manpage for `rdate` here....

While this is great for tcpdump-fluent network administrators, we must be slightly inventive to get the information to our central syslog server. Since tcpdump can read from the interface, we invoke it and send its output to the logger(1)⁴¹ utility, which then forwards the log to a syslog server.

```
tcpdump -e -n -tttt -i pflog0 -s 96 -X -v| logger
```

Both tcpdump and logger are part of the base OpenBSD system. To have this begin at startup, simply add the above line to the /etc/rc.local script (backgrounded with output to /dev/null, of course).

Networking

Configuring interfaces

In OpenBSD, interfaces are configured by placing ifconfig(8)⁴² parameters in a "hostname.if" file. For example, to set the interface dc0 to be set to 192.168.0.1 at boot time, with a netmask of 255.255.255.0:

```
echo "inet 192.168.0.1 netmask 255.255.255.0" > /etc/hostname.dc0
```

Default Route

Similarly, the default route is set in the file, /etc/mygate:

```
echo "192.168.0.254" > /etc/mygate
```

Enabling Firewalling

Pf may be enabled by setting

```
pf=YES
```

in /etc/rc.conf, and routing is enabled by setting

```
net.inet.ip.forwarding=1
```

in /etc/sysct.conf and rebooting the system.

⁴¹ <http://www.openbsd.org/cgi-bin/man.cgi?query=logger&apropos=0&sektion=0&manpath=OpenBSD+3.2&arch=i386&format=html>

⁴² <http://www.openbsd.org/cgi-bin/man.cgi?query=ifconfig&apropos=0&sektion=0&manpath=OpenBSD+3.2&arch=i386&format=html>

Administrivia

Adding users

User accounts can be added to the system by using the supplied 'adduser' script. The script inserts the account into the master.passwd file, creates the home directory, and optionally copies files into the home directory and emails a welcome message to the new user. To add a new user named 'jack':

```
danio# adduser
Use option ``-silent'' if you don't want to see all warnings and
questions.

Reading /etc/shells
Check /etc/master.passwd
Check /etc/group

Ok, let's go.
Don't worry about mistakes. I will give you the chance later to
correct any input.
Enter username [a-z0-9_-]: jack
Enter full name []: Jack D. Ripper
Enter shell csh ksh nologin sh [sh]:
Uid [1001]:
Login group jack [jack]: users
Login group is ``users''. Invite jack into other groups: guest no
[no]:
Enter password []:
Enter password again []:

Name:      jack
Password:  ****
Fullname:  Jack D. Ripper
Uid:       1001
Gid:       10 (users)
Groups:    users
HOME:      /home/jack
Shell:     /bin/sh
OK? (y/n) [y]: y
Added user ``jack''
Copy files from /etc/skel to /home/jack
Add another user? (y/n) [y]: n
```

Before users on an OpenBSD system may 'su' to the root account, they must first be explicitly placed in the 'wheel' group. For example, to allow our new user 'jack' to become the superuser, the /etc/group file must contain the line:

```
wheel:*:0:root,jack
```

Applying Patches

Before applying patches, the complete system source code must be present on the host. Also, verify that the system has enough disk space to allow

for the building of compiled programs or libraries. It may be desirable to have two identical systems, patch the standby, and then move it into production in place of the production system.

A tarball containing all patches for a release are downloadable from the OpenBSD website. For the 3.2 release, this location is:

<ftp://ftp.openbsd.org/pub/OpenBSD/patches/3.2.tar.gz>

Unpack the tarball. Patches required for the i386 platform will be located in 3.2/i386, while patches common to all OpenBSD releases will be located in 3.2/common. Patches are released in diff format, and are numbered numerically in order of release. It is best to apply the patches in order, as later patches may depend on changes made by prior patches. Comments at the beginning of the patch explain how to apply the patch and what other steps may be necessary.

For further information, Jacek Artymiak has an excellent guide to patching OpenBSD in his January 16, 2003 ONLamp article⁴³.

⁴³ "Patching OpenBSD." http://www.onlamp.com/pub/a/bsd/2003/01/16/ssn_openbsd.html

Appendix B: Network and System Information

Function	System	Version	Platform
Firewall	OpenBSD	3.2	i386
Router	IOS	12.1.12c.EA1	3550-EMI
SMTP Relay	Sendmail	8.12.18	OpenBSD/i386
DNS	BIND	9.2.1	OpenBSD/i386
Mail	Outlook 2000	SP2	Windows 2000SP2
Webservers	Apache	2.0.44	OpenBSD/i386
Servlet Engine	Tomcat	4	RedHat 8/i386
Database	MySQL	3.23.55	RedHat 8/i386
VPN	Cisco 3015	3.6.7	Cisco 3015

© SANS Institute 2003, Author retains full rights.

References

Albitz, Paul and Cricket Liu. DNS and BIND. 3rd ed. Sebastapol: O'Reilly, 1998.

Analysis Console for Intrusion Databases. 8 Jan. 2003. Roman Danyliw. 2 Feb. 2003. <<http://www.andrew.cmu.edu/~rdanyliw/snort/snortacid.html>>

Artymiak, Jacek. "Patching OpenBSD" 16 Jan. 2003.
<http://www.onlamp.com/pub/a/bsd/2003/01/16/ssn_openbsd.html>

Balaban, Murat. "Designing Shellcode Demystified" Date unknown.
<<http://www.enderunix.org/docs/en/sc-en.txt>>

Bernstein, D. J. "Internet Host SMTP Server Survey" 4 Oct 2001.
<<http://cr.yp.to/surveys/smtpsoftware6.txt>>

Big Brother System and Network Monitor. May 2002. BB4 Technologies.
<<http://www.bb4.com/>>

Center for Internet Security. 2002. Center for Internet Security. 20 January 2003. <<http://www.cisecurity.org>>

CERT Coordination Center. Feb. 2003. Carnegie-Mellon Software Engineering Institute. 28 Jan. 2003. <<http://www.cert.org>>

Cisco Connection Online. Jan. 2003. Cisco Systems, Inc. 22 January 2003.
<<http://www.cisco.com>>

Digital Offense. 25 Jan. 2003. Digital Offense. 8 Feb. 2003.
<<http://www.digitaloffense.com>>

Dittrich, David A. "Analysis of SSH crc32 compensation attack detector exploit". 15 Nov. 2001. <<http://staff.washington.edu/dittrich/misc/ssh-analysis.txt>>

Hartmeier, Daniel. "Design and Performance of the OpenBSD Stateful Packet Filter (pf)." Presented at Usenix 2002. 23 Jan. 2003.
<<http://www.benzedrine.cx/pf-slides.pdf>>

ISC BIND. Jan. 2003. Internet Software Consortium. 18 January 2003.
<<http://www.isc.org/products/BIND>>

Internet Explorer. January 2003. Microsoft Corporation. 23 January 2003.
<<http://www.microsoft.com/windows/ie/default.asp>>

Internet Protocol v4 Address Space. 19 Jan. 2003. Internet Assigned Numbers Authority. 10 Dec. 2002. <<http://www.iana.org/assignments/ipv4-address-space>>

Last Stage of Delirium (author). "Proof of Concept Code for Remote Sendmail Vulnerability" Mar 2003. http://www.security.nnov.ru/files/linx86_sendmail.c

Lokesh, B. K. "Denial of Service Attacks - DDOS, SMURF, FRAGGLE, TRINOO". 1 Mar. 2001. <http://www.sans.org/rr/threats/dos_attacks.php>

Microsoft. "Exchange 2000 Windows 2000 Connectivity Through Firewalls". 3 Oct 2002. <<http://support.microsoft.com/?kbid=280132>>

Nmap. 10 Aug 2002. Fyodor. 18 Jan. 2003. <<http://www.insecure.org/nmap>>

OpenBSD. 26 Dec. 2002. OpenBSD Project. 22 Jan. 2003. <<http://www.openbsd.org>>

Packet Storm. Feb. 2003. Packet Storm. 8 Feb. 2003. <<http://packetstorm.decepticons.org/>>

PuTTY. 1 Feb. 2003. Simon Tatham. 2001. <<http://www.chiark.greenend.org.uk/~sgtatham/putty/>>

RFC 1918: Address Allocation for Private Internets. February 1996. Network Working Group. 18 Jan. 2003. <<http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1918.html>>

Robinson-Wells, Janice. "Security Architecture and Policy" 2002. <http://www.giac.org/practical/Janice_Robinson-Wells_GCFW.doc>

SSH Insertion Attack. 11 June 1998. Core Security Technologies. 24 Jan. 2003. <<http://www.corest.com/common/showdoc.php?idxseccion=10&idx=131>>

Skoudis, Ed. Counter Hack. Upper Saddle River: Prentice Hall, 2002.

Stunnel – Universal SSL Wrapper. 12 Jan. 2003. Brian Hatch. 29 Jan. 2003. <<http://www.stunnel.org>>

TCPDump. Vers. 3.7.1. 21 Jan. 2002. <<http://www.tcpdump.org/release/tcpdump-3.7.1.tar.gz>>

TFN2K. 17 Dec. 1999. <<http://packetstorm.decepticons.org/distributed/tfn2k.tgz>>

Wash, Rick. "Bind 9 in –current." Online posting. 21 Jan. 2003. OpenBSD Journal. 21 Jan. 2003.
<<http://www.deadly.org/article.php3?sid=20030121022208&mode=flat>>

© SANS Institute 2003, Author retains full rights.