Global Information Assurance Certification Paper

Building a More Secure Network

George Rosamond
March 8, 2004
GSEC Practical Requirements (v.1.4b) (August 2002)

- Abstract

Securely designed network topography is a common theme among vendors and end-user administrators. However, secure network design is not just about enormously expensive hardware and software. It can be accomplished with a well-thought out strategy and freely available open source software. This document will look at a template for secure network design, and how it can be integrated into both perimeter regions and internal segments with little outlay in capital expenditures.

When firms, including less capital-flush small and mid-sized entities, look to increase the level of security on their networks, they frequently look to expensive hardware and software solutions. The associated expenditures strain their meager budgets, particularly in these difficult economic times. Additionally, the costs of administration, training and installation are also beyond the much decreased IT budgets.

Table of Contents

• Introduction

With a goal of building a more secure network topography, this piece will look to breakdown some elements that can easily be integrated into a network or used as a template for building from scratch. Expenditures for this approach do not need to exceed the means of most firms.

Much of the necessary hardware is easily obtainable. Even in the difficult economic conditions of New York City today, Intel-based Pentium II boxes are frequently found on the street on “garbage nights.”

In terms of software, we will use the free, open source BSD's: Berkeley Software Distribution.

The role of BSD Unix, the open source operating system originating at 1970's Berkeley, will be central to our design. The variants of BSD, including FreeBSD[1], NetBSD[2] and OpenBSD[3], have a strong tradition of providing high-performance with stability and security. OpenBSD, in particular, holds an impressive record with only having one remote vulnerability in almost eight years of existence. There is a large array of software available for the BSD family, in addition to the ability to run Linux packages via emulation. Numerous native packages deal with syslog outputs, firewalling, radius servers for authentication and much more.

The starting point for developing a secure network topography is a well conceived security policy, particularly one that categorizes users based upon rights and privileges.

Such organization is important in deciding how and where to segment your network.

There are often users at firms that are contracted or do not have a direct stake in maintaining the operation's integrity. We will look to isolate these users as they most likely only need a limited number of services available from the private network.

Security policy is also critical for categorizing the firm's technical resources. For instance, if your firm utilizes the internet for relations with clients, vendors, and other firms, we categorize them separately from the various monitoring and surveillance systems that the internal IT staff relies upon for the maintenance of systems.

The New Riders' book entitled Inside Network Perimeter Security sums up the concept, which is labeled as “security zones”:

1 FreeBSD website. http://www.freebsd.org (March 6, 2004).
2 NetBSD website. http://www.netbsd.org (March 6, 2004).
3 OpenBSD website. http://www.openbsd.org (March 6, 2004).

a logical grouping of resources, such as systems, networks or processes, that are similar in the degree of acceptable risk.[4]

Below is a basic template with examples for understanding these categories:

Resource	*Role*	*Audience/User*	*Usual Location*
MRTG Server[5]	charting traffic on firewall and perimeter router	IT Staff	DMZ
WWW Server	maintaining web presence on the internet, providing current and potential customers with firm data	∞planet Earth∞ Clients	DMZ
DNS Server	providing dns information to firm and providing primary dns functions for firm's publically addressed servers	internal firm users, other dns servers for replication	DMZ
Mail Gateway	transferring external mail into the network and scrub for viruses, etc and also transferring internal mail to remote users	∞planet Earth∞ every mail sender or recipient interacting with the firm's users	DMZ
File Server	providing home and shared directories to internal users	internal firm users	Private Network
Mail Server	providing mail services to internal users	internal firm users	Private Network

There are many variations on the above listed services, and there are likely to be many more to add. There could, for example, certainly be one or more DNS server in the private network. But that table should give some idea as to categorizing the data.

4 Northcutt, Stephen; Zeltser, Lenny; Winters, Scott; Kent Frederick, Karen; Ritchey, Ronald W. Inside Network Perimeter Security. Indianapolis: New Riders, 2003. Page 324.

5 Multi Router Traffic Grapher. http://people.ee.ethz.ch/~oetiker/webtools/mrtg/ (March 6, 2004). MRTG is an application that monitors network traffic and generates web pages based on the output.

- What this Document Doesn't Cover

The scope of this document is quite general, but we tackle a number of specific problems and propose some very specific solutions.

However, there is more to securing a network than technical solutions. There are a number of features that are not addressed, but are crucial to that goal.

First, and most importantly, is physical security. No physically insecure network is safe. You could be using Blowfish[6] encryption on an OpenBSD server's password file, yet if someone slaves the drive on another machine, the encryption algorithm just doesn't matter, since root access is unnecessary. Certainly, if someone rips out a few cables here and there, or even takes a sledge hammer to your hardware, secure network topography, software and operating systems are irrelevant since you lose access.

Second, we aren't going to approach the full details of an implementable security policy. We have touched one of the roles of the security policy, but much more is necessary. Having a security policy is also vital for having enforceable standards that allow punitive action to be taken in the event of computer misuse by a staff member. That is why it helps if all staff are required to sign-off on the security policy as part of any pre-hire documentation. The security policy also assists the firm in understanding how to deal with virus outbreaks, intrusions, etc.

Finally, user awareness and education may be the most critical element in network security. If users are writing their passwords down on little stickies like a lion's mane around their monitor, or if they have no problem telling remote “tech support” their password, beware. Many successful security intrusions come from simple variations of the basics: social engineering and user complacency.

With that covered, let's move onto network topography, and the standard approach of IT staffs.

6 Schneier, Bruce. “The Blowfish Encryption Algorithm.” http://www.schneier.com/blowfish.html (March 6, 2004).

- General Network Layout: The Current Model

The standard layout for most networks today is the public-demilitarized-private setup. There's nothing magical about this equation. Its logic starts with categorizing types of activity, and positioning resources according to the scale of security and accessibility.

The public, or internet, section of this layout, is simultaneously a Wild West-like environment and also the location for valuable resources, both information and human. The open internet, despite various attempts at regulation, remains a place of extremes and dangers. Child pornography and political terrorism roam happily in the chaos, infrequently hindered by legal systems. But your firm's business also relies on vendors, partners, remote staff, etc., who also dwell on the internet. The internet is critical to any firm's basic needs.

Secondly, we have the demilitarized zone (DMZ) which contains data that needs to be publically accessible for certain necessary services. This could include mail, web-serving, ftp, etc. Frequently the most misconfigured part of network layouts, DMZ should not have direct or unrestricted access to the private network. For the purposes of this document, we'll call this the “public DMZ,” as it contains data that is served to the internet.

The private section of a network is where the protected day-to-day operations of a firm occur. It needs to be isolated from the internet mayhem, with an eye on cracking attempts, viruses, etc. By no means should all hardware, software and people within this section be considered trusted. The greatest vulnerability most firms face is within their network, from conscious or unconscious problems arising from user actions. This could include incoming, infected email files, malicious attempts to remove intellectual property, infected data brought from the outside on floppy disks, etc.

This model as a general guide works well for most firms. But there are some additional changes we can make to this configuration that will greatly increase its robustness.

- Some Basic Additions to the Public-DMZ-Private Architecture

There are a number critical functions which are often placed in this DMZ that we want to isolate from the publically accessible servers and functions.

One fundamental change we are going to look at with this standard network layout is an additional network interface, that we'll call the "administrative DMZ." These additional DMZs are common with larger hosting and telecommunications firms, but can be useful for smaller operations as well. There is also some reference to this concept in Inside Network Perimeter Security, where it is labeled a "management console."[7]

Additionally, we will look at adding a "hidden network" to DMZ servers for radius authentication, monitoring/intrusion detection and backup purposes. Positioning these functions with private, non-routable addresses, allows for unencumbered surveillance to occur, without any incoming traffic being aware unless the intruders are directly on the DMZ servers.

Finally, we will also strategically locate some packet filtering bridges, so that traffic we want to filter is dropped without the incoming host being aware of where the packet loss is occurring.

Bridges act as a mere connecting device for two of the same networks. Bridges do not have internet protocol (IP) addresses, so that they are not remotely mappable or accessible. They can only be administered locally.

For this, it's recommended to utilize OpenBSD's packet filter, PF[8]. When there were licensing disputes with the standard packet filtering software IPFilter[9], the OpenBSD developers began creating an alternate port starting with OpenBSD version 3.0[10]. Its development since 2001 has been remarkable[11]. PF's coding language is simple, yet its capabilities are great. It is a fully functional stateful firewall with extras such as operating system fingerprinting of ingress traffic, load balancing and network address translation (NAT). For our purposes, there's no need to utilize NAT as there's no address translation occurring with a bridge.

---

7 Northcutt, Stephen; Zeltser, Lenny; Winters, Scott; Kent Frederick, Karen; Ritchey, Ronald W. Inside Network Perimeter Security. Indianapolis: New Riders, 2003. Page 176, 344.

8 OpenBSD web site. "PF: The OpenBSD Packet Filter." http://www.openbsd.org/faq/pf/ (March 6, 2004).

9 IP Filter – TCP/IP Firewall/NAT Software. http://coombs.anu.edu.au/~avalon/ (March 6, 2004).

10 Hartmeier, Daniel. "OpenBSD Packet Filter – How it Started (June 2001)." June 2001. http://www.benzedrine.cx/pf-beginning.html (March 6, 2004).

11 Hartmeier, Daniel. "OpenBSD Packet Filter." September 9, 2003. http://www.benzedrine.cx/pf.html (March 6, 2004).

- OpenBSD: Secure by Default

It's difficult to overstate the foresight of the OpenBSD developers in their security sophistication.

In November 2002, Steven M. Bellovin of AT&T Labs Research wrote an article entitled "A Technique for Counting NATed Hosts."[12] As so many hosts are hidden by NAT, it's not only hard to actually count the total number of devices on the internet, but it's seemingly impossible for internet service providers (ISPs) to limit the number of hosts accessing the internet if behind a device providing NAT services. He states that "the IP header's ID field is a simple counter. By suitable processing of trace data, packets emanating from individual machines can be isolated, and the number of machines determined."[13]

Bellovin notes that "(r)ecent versions of OpenBSD and some versions of FreeBSD [actually, it's a simple kernel hack in FreeBSD-GR] use a pseudo-random number generator for the IPid field." He goes on to warn, however, that "(a) keyed generator, as is used in OpenBSD and FreeBSD, provides some protection, but one needs to be careful to avoid duplication if the generator is rekeyed periodically."[14]

But to add to the case for PF, there's a simple setting that's included in its configuration file (/etc/pf.conf by default) that takes this defensive measure a step further.

Michael Lucas, in his May 2003 No Starch Press book entitled Absolute OpenBSD explains the PF option of "State Modulation":

> it provides additional security enhancements for poor TCP/IP stacks. . . State modulation replaces the ISN [the TCP/IP initial sequence number-GR] sent by every machine involved in a connection by a highly random ISN, and translates as needed for each client. While it only works for TCP connections, it provides a much greater level of TCP protocol security than the native TCP stack found in the most common desktop operating systems.[15]

In other words, instead of worrying about possible TCP sessions being hijacked due to ISN hijacking, a PF firewall can address the issue for the devices it's protecting.

12 Bellovin, Steven M. "A Technique for Counting NATed Hosts." November 2002. http://www.research.att.com/~smb/papers/fnat.pdf (March 6, 2004). Story also referenced on Slashdot.org at http://slashdot.org/article.pl?sid=03/02/05/2129218&mode=thread.
13 Ibid., page 267.
14 Ibid., pages 267, 271.
15 Lucas, Michael. Absolute OpenBSD. San Francisco: No Starch Press, 2003. Page 380.

We'll look to utilize OpenBSD as the core operating system in these PF bridges, as we will refer to them throughout this document. OpenBSD's approach to security is to be “secure by default”, thereby minimizing network administrator configuration errors and those application vulnerabilities that are posted so regularly on security lists like Bugtraq[16].

The default install for an OpenBSD system is quite bare. Run the Unix command “top” at the command prompt, and you will find a handful of processes running when you first boot the machine. Note that these statistics include my access to the server via OpenSSH.

```
load averages:  0.09,  0.10,  0.08                                  15:50:51
19 processes:  1 running, 15 idle
CPU states:  0.0% user,  0.0% nice,  0.2% system,  0.0% interrupt, 99.8% idle
Memory: Real: 6964K/24M act/tot  Free: 97M  Swap: 0K/300M used/tot

  PID USERNAME PRI NICE  SIZE   RES STATE WAIT   TIME    CPU COMMAND
33323 gman     2    0   348K  1148K sleep select 0:00  0:00% sshd
 3326 root     2    0   376K  1360K sleep netio  0:00  0.00% sshd
 4706 root     2    0   804K   932K sleep select 0:00  0.00% sendmail
 7851 _syslogd 2    0   176K   472K sleep select 0:00  0.00% syslogd
11347 root     18   0   348K   284K sleep pause  0:00  0.00% csh
11724 root     2    0   136K   444K sleep netio  0:00  0.00% syslogd
24368 root     28   0   164K   848K run   -      0:00  0.00% top
    1 root     10   0   336K   224K idle  wait   0:00  0.00% init
 3262 root     3    0    60K   464K idle  ttyin  0:00  0.00% getty
 1299 root     2    0   132K   448K idle  select 0:00  0.00% inetd
28803 root     3    0    76K   472K idle  ttyin  0:00  0.00% getty
25778 root     3    0    76K   472K idle  ttyin  0:00  0.00% getty
31027 root     3    0    80K   472K idle  ttyin  0:00  0.00% getty
10471 root     3    0    96K   476K idle  ttyin  0:00  0.00% getty
15410 root     2    0   292K   548K idle  select 0:00  0.00% cron
```

Run the Unix command “ps -ax | more” and you'll find you don't even need to pipe the command, since there is only more service than can be displayed by the “top” command, and it all fits on one displayed screen.

```
brick# ps -ax | more
  PID  TT  STAT      TIME COMMAND
    1  ??  Is    0:00.01 /sbin/init
11724  ??  Is    0:00.02 syslogd: [priv] (syslogd)
 7851  ??  I     0:00.03 syslogd -a /var/empty/dev/log
 1299  ??  Is    0:00.01 inetd
 5906  ??  Is    0:00.01 /usr/sbin/sshd
 4706  ??  Is    0:00.04 sendmail: accepting connections (sendmail)
15410  ??  Is    0:00.01 cron
```

16 Bugtraq on the SecurityFocus Home Page. http://www.securityfocus.com/ (March 6, 2004).

```
 3326   ??   Is    0:00.07   sshd: gman [priv] (sshd)
32223   ??   I     0:00.08   sshd: gman@ttyp0 (sshd)
11347   p0   I     0:00.02   -csh (csh)
 2627   p0   R+    0:00.00   ps -ax
 3262   C0   Is+   0:00.01   /usr/libexec/getty Pc ttyC0
28803   C1   Is+   0:00.01   /usr/libexec/getty Pc ttyC1
31027   C2   Is+   0:00.01   /usr/libexec/getty Pc ttyC2
25778   C3   Is+   0:00.01   /usr/libexec/getty Pc ttyC3
10471   C5   Is+   0:00.01   /usr/libexec/getty Pc ttyC5
```

The major drawback, however, is that OpenBSD is currently incapable of fully utilizing symmetric multiprocessing (smp)[17]. While OpenBSD's present ability to scale to larger network roles is hindered, a PF bridge's performance is not likely to improve with the addition of SMP functionality.

PF bridges can assist our network in a number of ways.

We can invisibly restrict traffic moving in and out of our network are various choke points, so that packets are merely dropped if they don't match our ingress firewall rule sets.

These PF bridges don't have to be only for potentially hostile incoming traffic. We can also place them in locations which restrict outgoing traffic from certain categories of users.

There is nothing 100% secure in having a network connected to the internet, unless one disconnects the public interface. However we are going to move in that direction as closely as possible.

17 OpenBSD web site. "The OpenBSD SMP Project." http://www.openbsd.org/smp.html (March 6, 2004).

- Creating an Administrative DMZ: Where Control Lies

Currently, many firms have a DMZ that is for end users to interact act with the "outside world" and for external users to interact with the firm, such as with a web server.

There are, however, a variety of administration related functions that we don't want sitting in the same segment as these publically accessible servers.

If you have the extra interface to create a second DMZ, you can dedicate it solely to managerial functions, namely monitoring your DMZ and external network hardware. This segment would be solely for some or all of the technical staff, and would include all tasks unrelated to providing outside services.

Applications such as Multi Router Traffic Grapher (MRTG)[18], HP's OpenView[19], an additional logging server, etc., could be administered from this second DMZ. This would allow you to maintain regular, real-time logging of the external hardware, such as the firewall, routers, switches and servers, yet restrict the traffic going into this second DMZ to only SNMP or syslog information.

This log server would focus primarily on archiving data from the other two administrative servers, plus accept logs from the firewall.

This is also an excellent location to place some type of sniffing software, to watch and maybe chart network traffic flowing in and out of your network. A sniffing program monitors all packets passing by its promiscuous mode network card.

Having an administrative DMZ doesn't
negate the possibility of having the relevant users access these servers remotely. While virtual private networking and Smart cards/SecureID's, along with regularly changing complex passwords is unmanageable for large numbers of people, it's a possibility for at least some technical staff. Restricting this access to only a secure shell package, such as OpenSSH[20], would increase the strength of this arrangement.

Such measures of security are difficult to administer and enforce with the majority of users in a firm, and often even with the majority of a technical staff. But for the restrictive purposes of the administrative DMZ, these complicated security measures are only applicable to the few who can handle the responsibility and parameters.

18 Ibid.
19 Hewlett-Packard OpenView. http://www.openview.hp.com/ (March 6, 2004).
20 OpenSSH. http://www.openssh.org/ (March 6, 2004).

- "Hidden" Private Network in the Public DMZ

The O'Reilly and Associates book Incident Response[21] provides a basic diagram of a "hidden" interface for placement of an intrusion detection server. The scheme is simple. In this DMZ, all servers have two sets of addresses: one set to a public address, the other set to a private address.

Through the public interface, the network services are handled with the internet.

On the private IP end of the servers, there's one way, read-only NFS connections to a Tripwire IDS system, which has a line to a modem for page the technical staff.

It's a brilliant idea for the purpose of security surveillance outside the curious probes of an intruder. As soon as there's an anomaly with any of the servers that the Tripwire system detects, the appropriate technical staff is notified via pager. And the intruder has no way of knowing about this separate network, short of gaining direct access to the relevant servers.[22]

But we can take this concept a step further. Let's create a "hidden" network where some critical functions are performed for the DMZ that you don't want anyone to know that it even exists.

For each server in the DMZ, you will have an additional network interface card with a private, non-routable address, such as the 172.16.0.0/12 network as is illustrated in the network diagram in the appendix.

On the inside interface should be a switch with the servers you want hidden. This could include a logging server to handle the DMZ logs, a radius authentication server for an ftp server, a backup server with fresh images to quickly rebuild a compromised server in the DMZ and an intrusion detection server to monitor the servers.

Each of the boxes that are running some variant of Unix (such as a BSD) on the hidden network should be tcp-wrapped[23], which can limit the acceptable traffic to only the DMZ machines' IP addresses.

The OpenBSD daemon TCPD[24] contains queries to services listed in the

---

21 van Wyck, Kenneth R. and Forno, Richard. Incident Response. Sabastopol, California: O'Reilly & Associates, July 2001. Pages 137-138.
22 Ibid. Page 137.
23 OpenBSD web site: TCPD (8) manual page. http://www.openbsd.org/cgi-bin/man.cgi?query=tcpd&apropos=0&sektion=0&manpath=OpenBSD+Current&arch=i386&format=html (March 6, 2004).
24 Ibid.

configuration file /etc/inetd.conf, and can determine access according to host addresses in the
/etc/hosts.allow and /etc/hosts.deny files.

Let's look at an example from the OpenBSD manual pages[25].

The service finger enables a user to gain information about a remote user on a remote host. While less and less frequently enabled on any device today, its role is certainly useful in troubleshooting network issues.

Therefore instead of merely enabling the finger daemon, FINGERD, in the /etc/inetd.conf file, we'll replace the configuration with TCPD:

finger stream tcp nowait nobody /usr/libexec/fingerd fingerd

becomes:

finger stream tcp nowait nobody /usr/libexec/tcpd fingerd[26]

We would first create an /etc/hosts.deny file that would include the following:

ALL: ALL

Meanwhile, our /etc/hosts.allow file might state the following for the network diagram in the appendix, if we were restricting only those within our hidden network off the public DMZ:

fingerd: 172.16.0.

Only machines with the IP addresses from 172.16.0.1 through 172.16.0.255 can utilize the finger daemon on the relevant machine.

For our purposes, we might look to use TCPD to limit the hosts that are allowed to send requests to UDP port 54 on the logging server, the port at which syslog output is gathered.

Additionally, if both the servers in the DMZ and the servers on your hidden network have non-routable addresses, such as the 172.16.0.0/12 and 192.168.0.0/16 networks as we use in the network diagram appendix, then routing packets between

25 Ibid.
26 OpenBSD web site: FINGERD (8) manual page. http://www.openbsd.org/cgi-bin/man.cgi?query=fingerd&apropos=0&sektion=0&manpath=OpenBSD+Current&arch=i386&format=html (March 6, 2004).

the DMZ and the hidden network is impossible.

The hidden network is only accessible if one of the DMZ servers is directly compromised. The remote cracker must be authenticated on the server to find the additional, non-routable interface.

You can access this hidden network if you have OpenSSH running on the DMZ servers, then also run it on the hidden network servers. Additionally, you may want modems on the hidden network either for remote access into them, or for notification in the event of an anomaly with the IDS as done in the Incident Response example. In the later case, you can use non-Direct Inward Dialing numbers, which also impedes, to some extent, war-dialing attempts.

- OpenBSD PF Bridges: Dropping Bad Traffic Without a Culprit

We have already covered some details of the OpenBSD PF bridges. But its general topographical role in network is also critical.

OpenBSD PF bridges can drastically increase security for any network architecture.

In the words of one of my close collaborators,

> A firewall is a doorway that everyone knows they want to get through. A packet-filtering bridge is more like a secret agent that picks off the bad guys from the shadows whom nobody can attack back.[27]

It's a simple concept that easily fits into any network configuration.

This extra network security device's role in secure network topography is credited to the Openly Secure website, according to one of the authors of Building Linux and OpenBSD Firewalls[28] and the recently published Network Security Illustrated[29], Wes Sonnenreich.

Sonnenreich's plug for this “invisible firewall” is useful in further understanding the concept. Note that he refers to IPFilter, which was still the default packet filter for OpenBSD at the time of publication:

> From inside the network, everything works fine. But inexplicably, your mailserver simply is unreachable on any port other that 25. There's simply no response... in fact, your packets seem to vanish into the ether.
>
> You're experiencing the power and ease of a transparent bridging firewall. It doesn't require any reconfiguration of your network (other than plugging it between your router and your network) yet allows you to completely control network access with the full power of the IPFilter firewalling system. And the best part? The firewall is invisible and inaccessible from anywhere other than the console. This means a hacker can't possibly break into your firewall. It just doesn't exist.[30]

---

27 Matteo Ames, networking consultant

28 Sonnenreich, Wes and Yates, Tom. Building Linux and OpenBSD Firewalls. New York: John Wiley & Sons, Inc., 2000.

29 Sonnenreich, Wes and Albanese, Jason. Network Security Illustrated. New York: McGraw-Hill, 2004.

30 Sonnenreich, Wes. “Memoirs of an Invisible Firewall.” September 6, 2001. http://openlysecure.org/openbsd/how-to/invisible_firewall.html (March 5, 2004).

To understand the function of this hardware, we should differentiate between a bridge and a router. Routers connect traffic between two different networks, bridges merely pass packets from one interface to another. An important consequence of being a bridge is that it is not assigned an IP address, and therefore is not remotely mappable. But bridges don't have to mindlessly move the packets from one interface to another, they can also "mysteriously" drop unwanted packets. By building a two-interfaced OpenBSD server with PF, we can drop unwanted packets going in and out and remain invisible to the outside world.

When configuring the packet-filter bridge, figure out only the essential traffic you want coming into and going out of your network.

As a side note, for those who want to have relatively secure access to the bridge remotely, you can add a third interface with an IP address exclusively for OpenSSH access.

However, as this could give clues to the actual existence of a device between the perimeter router and the firewall, think about your decision carefully.

If you do setup the third, OpenSSH-only interface, configuration should be rock-solid. Beside only allowing version 2 of OpenSSH, you should enable tcp wrappers, and other basic security rules. A somewhat dated document I wrote in September 2002 entitled “A FreeBSD Operating System Security Checklist”[31] still provides some details on secure configuration of OpenSSH. Remember, you are not only opening a potential hole into your network by adding this third interface, you are also providing clues as to the real nature of your "mysterious" network topography.

Let's now refer to the appendix's network diagram to further illustrate the role of the PF bridge.

Coming toward your network from the internet, the step after your perimeter router but before your firewall sits a PF bridge (**A**).

An outside packet would encounter the PF bridge after the router but before the firewall. Make a list of the services and their respective ports you are providing to the public internet.

This might include TCP port 25 for SMTP mail traffic, TCP ports 80 and 443 for unencrypted and encrypted traffic to your web server, TCP and UDP port 53 for DNS requests and zone transfers, TCP port 22 for secure shell, and so on[32]. Return to the

31 Rosamond, George. “A FreeBSD 4.8-RELEASE Operating System Security Checklist.” June 12, 2003. http://www.sddi.net/FBSDSecCheckList.html (March 6, 2004).
32 Internet Assigned Numbers Authority. “Port Assignments.” March 4, 2004.

list of resources we created earlier in the document, and determine the applicable services.

From this list, you can then compile the appropriate PF rules as stated in the OpenBSD /etc/pf.conf file.

For instance, to allow secure shell access to your web server with the public IP address of 1.2.3.4/24, the PF rule would state the following:

```
pass in proto tcp from any to 1.2.3.4 port 22
```

There is certainly a case for using access control lists on perimeter (and even internal) routers, but a routers job should be to route. We can limit the router to just blocking the non-routable addresses coming into the network, such as external machines claiming to be on the 192.168.0.0/16 network and dropping them. Limiting the load on this hardware is critical. The PF bridge would be the first true obstacle to unwanted traffic.

Undesired packets would “mysteriously”disappear. To an intruder, creating an accurate network mapping would be difficult, even if the perimeter router was compromised. No addressable device could be viewed as a culprit.

http://www.iana.org/assignments/port-numbers (March 5, 2004).

- Protecting Your Servers & Restricting Users in the Private Network

If the majority of security issues are related to internal users, then why not add a PF bridge as a corral for the less trusted section of users?

This second instance of the PF bridge (**B**) would be placed for internal segmentation reasons. Many firms' industries, such as finance and telemarketing, have sales or related staff who only need limited access to the outside world, and are often the most liable for security issues. They may be retail brokers who need basic internet access, email and possibly intranet access to a leads database. Such staff are frequently independent to varying extents of the basic integrity of the firm, in that they operate on commission and not salary or wages. This could also be the case with some independent contractors. So a PF bridge hindering their access to the rest of the internal network, and the outside world, may be critical.

The third instance in which we are inserting a bridge (**C**) is for the protection of the firm's servers. As this represents at least a portion of the intellectual value of the firm, whether in the form of leads, research, etc., it's important that an additional layer of security exists. As noted earlier, the majority of security issues originate inside a network–in the private interface–therefore this segment of the network should also be considered hostile territory.

Local area network users could have access to the necessary ports, such as pop TCP port 110 for mail, TCP port 80 for an intranet server access, etc., but the majority of traffic would be blocked.

We can assume users are dangerous for what they don't know, probably more than what they do know. Yes, you always get the users who try to run l0pht crack off a floppy in your network, but the most likely day-to-day security problem arises from ignorance on the part of users.

Placing a PF bridge between the users and the backed servers of your network will restrict much of the unnecessary traffic from entering the server realm.

Start with listing the services your users need. Intranet access on TCP port 80, pop mail on TCP port 110, etc. This measure will also increase the performance on your servers, as they will not be dealing with unnecessary requests.

The source IP addresses could also be predetermined for traffic allowed into the server realm. It's common today for an outside party to need internet connectivity from your internal network for sales presentations, etc. Limit the scope of addresses coming into the server realm via the /etc/pc.conf file, and include an extra range of addresses that could be used by these outside parties. From their alien hardware,

they would be unable to access your internal services that reside in the server realm.

There are obviously a number of vulnerabilities that are exploitable over the ports that are left open, but we are at least dealing with illegitimate request to unused ports, particularly those greater than 1024.

- Assessing the Security of this Network Design with Free & Open Source Tools

Firms use a wide array of tools to monitor and secure their network. It's not uncommon to spend thousands of dollars on software and services from firms such as Internet Security Systems[33] and Foundstone[34] to assess the strengths and weaknesses of a network topography and content.

Bruce Schneier, generally considered one of the top electronic security gurus of the world, paints an almost humorous picture of this game, which ultimately caused him to shift the focus of his firm, Counterpane[35]

> (A) company would come to us with an already designed system that purported to be secure against a list of threats, and we would poke holes in the solution and then fix them. We could invoice as many hours as we could stay awake. (p 396)[36]

That was during the boom times of the late 1990's. It's unlikely that many chief financial officers and controllers are willing to allow such expenditures today. And it's particularly the case when the task is comparable to Sisyphus' struggle of eternally pushing a rock to the top of a mountain, only to have it roll down again.

The quest for a decent vulnerability assessment can be destructive to your firm's coffers and morale. Particularly when you discover that the majority of crackers in the world are utilizing free or open source tools that are available to all.

We can't perform an exhaustive review of the free and open source tools that are useful in assessing a network's vulnerabilities, but we'll consider a few of the more important ones.

The first is ported to practically every operating system today, and is one of the classics of the TCP/IP suite: trace route[37].

The trace route command maps the number of hops between the originating IP and a destination IP address.

33 Internet Security Systems. http://www.iss.net
34 Foundstone. http://www.foundstone.com
35 Counterpane. http://www.counterpane.com
36 Schneier, Bruce. Secrets and Lies. New York: Jon Wiley & Sons, Inc. 2000. page 396.
37 OpenBSD web site: TRACE ROUTE (8) manual page. http://www.openbsd.org/cgi-bin/man.cgi?query=traceroute&apropos=0&sektion=0&manpath=OpenBSD+Current&arch=i386&format=html (March 6, 2004).

For instance, if your mail server is publically viewable to the world, via a DNS mail exchange record (MX) and its IP address is 1.2.3.4, we can deduce some basic but useful information.

At a minimum, trace route will illustrate who provides your internet connectivity and the relevant routers along the way, in addition to your own perimeter router(s) and ultimately your mail server.

And unfortunately, due to the standard device naming conventions, much is revealed by the trace route command.

Nmap's[38] role in the world of network security and auditing can not be overestimated.

Fyodor, a respected security hacker who develops Nmap, is a legend to many. He describes Nmap as a:

> utility for network exploration or security auditing. It supports ping scanning (determine which hosts on a network are up), many port scanning techniques (determine what services the hosts are offering), and TCP/IP fingerprinting (remote host operating system identification). Nmap also offers flexible target and port specification, decoy/stealth scanning, sunRPC scanning, and more. Most Unix and Windows platforms are supported in both GUI and command-line modes.[39]

Of note is the mention of operating system fingerprinting by TCP/IP. Mentioned briefly when earlier the listing of PF features, OS fingerprinting allows the intruder (or auditor) to remotely determine the operating system. While never perfect in its output, this is an enormously useful ability, particularly as new operating system and application vulnerabilities are a regular occurrence, and it's the rare systems administrator that keeps their servers patched to the moment.

The last utility to mention is EtherApe[40], possibly the most enjoyable tool to actually watch in operation, as long as you're not watching an attack of your own systems.

It runs on any of the free Unix-like operating systems such as the BSDs or Linux in X Windows, and provide as a graphical representation of the various connections. If you're not allowing any OpenSSH connections to your web server, yet you continually notice a remote host attempting to connect via TCP port 22, you know

38 Insecure.org Web Site. "Nmap – Free Security Scanner for Network Exploration & Security Audits." http://www.insecure.org/nmap/ (March 5, 2004).
39 Insecure.org Web Site. "Who is Fyodor?." http://www.insecure.org/myworld.html (March 5, 2004).
40 EtherApe Web Page on Sourceforget.net. "EtherApe, a graphical network monitor." http://etherape.sourceforge.net/(March 5, 2004).

immediately someone is doing something wrong. And EtherApe lists those particular connections in yellow, as opposed to white for HTTP connections.

EtherApe running on a console is useful to gaining a picture of the various network connections to your servers. It is an excellent first line of defense that works well when integrated with a good logging system.

Some firewalls, such as Watchguard[41], have similar monitoring tools which provide you with the same picture, but of course you are required to purchase their product to utilize the function.

41 Watchguard Web Site. http://www.watchguard.com/ (March 5, 2004).

- Problems with This Architecture

Through basic inference, we can note a number of possible problems with this network topography.

There are the obvious issues that we can reiterate. In the private interface of the appendix's network diagram, the possibility of social engineering and trojans residing among the unrestricted clients or servers could certainly be an issue.

Also noted earlier, this topography doesn't deal with potential physical security issues. They may be separate from the thrust of this piece, but we can not negate the impact.

Next, even though we are dropping a number of packets before they even reach the firewall, we are expecting our firewall to deal with large amounts of specific traffic between the interfaces. Certainly, a four interface firewall is not uncommon, but many firewalls today were quickly purchased a few years ago when senior management realized that perimeter security was important, or inexperienced administrators thrust into their positions by the technical boom were quick to move with the basic options provided by eager vendors.

As we have positioned the PF bridges in a number of critical locations, their hardware demands a high mean-time between failures. It is true that the BSD's are noted for their stability and impressive uptimes. The Netcraft.com's uptime surveys consistently show the long uptime of BSD web servers, to the extent that the top fifty in the survey are either all BSD or more infrequently only one other server may eek its way into the list.

However infrequent, there are sometimes vulnerabilities in the BSD operating systems and even more often in the applications that they run.

And software stability doesn't equal hardware reliability. There have been huge strides in hardware reliability over the past few years, even with basic desktops we are utilizing for the PF bridges, but improvements don't equal 100%.

So in the event of hardware failure, there is no redundancy or backup in this setup. That isn't a problem if the appropriate technical staff are onsite, but as PF bridges aren't addressed, they can't be remotely accessed. Great for security, but not great for being fail-safe.

• Conclusions

It should be clear from this piece that raising the level of security on a network is within the abilities of most firms without driving the operation into Chapter 11.

While security itself is rightly viewed as a process, there are a number of topographical approaches that can greatly improve your defensive standing.

This paper has covered various infrastructure-related improvements that could assist many firms, particularly small and mid-sized businesses. None of them requires hardware or software expenditures as the necessary hardware considered "junk" by many, and the software is free and open source.

The main constraint would clearly be both the willingness of senior management to to allow these changes, and most importantly, having a technically proficient IT staff capable of implementation.

The war for secure networks has no end point. Vulnerabilities in operating systems and applications have no end in sight. There are no magic bullets in security.

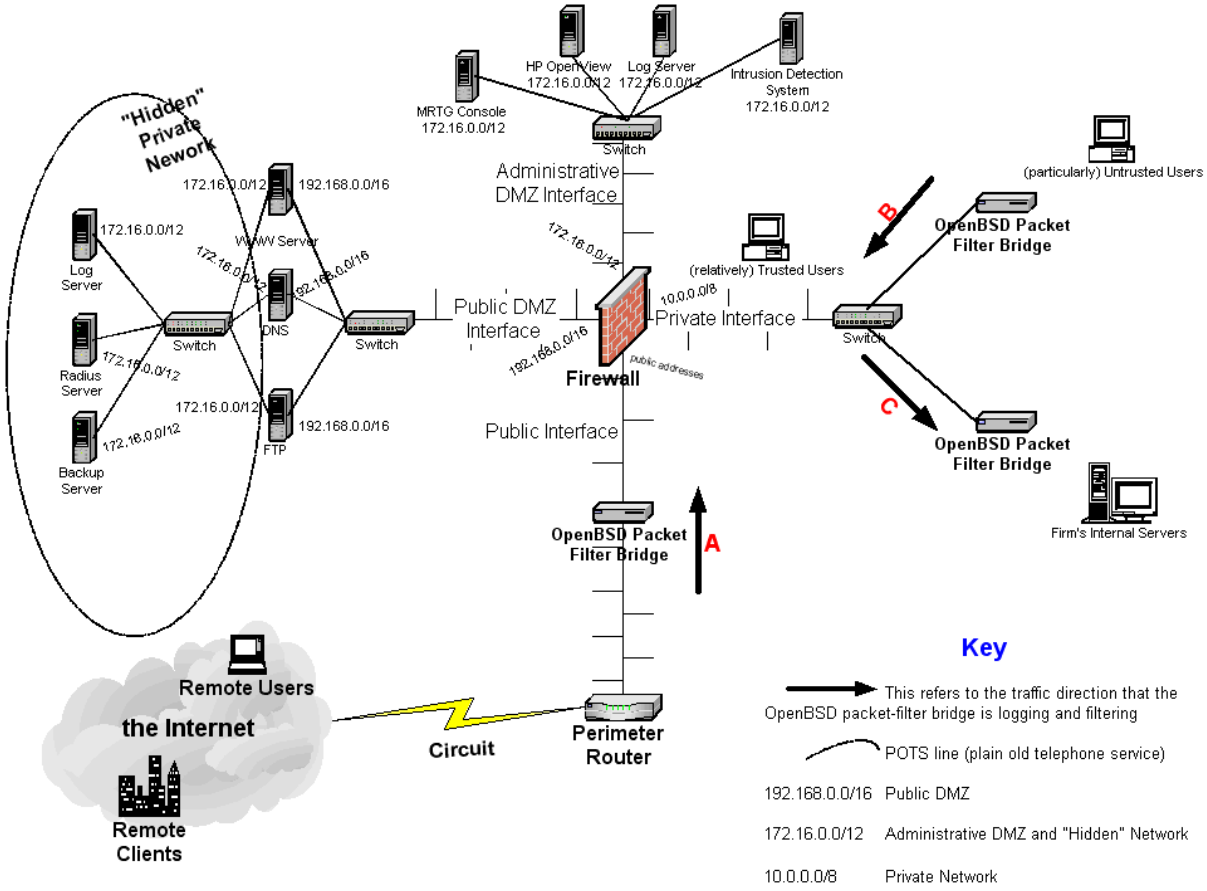
While Bruce Schneier is speaking specifically about the role of encryption in this citation, we can clearly generalize to the larger technical security picture.

> We draw boxes around different players and lines between them. We define different attackers—eavesdroppers, impersonators, thieves—and their capabilities. We use preventive countermeasures like encryption and access control to avoid different threats. If we can avoid the threats, we've won. If we can't, we've lost.
>
> Imagine my surprise when I learned that the world doesn't work this way.[42]

Schneier's ultimate conclusion is not to give-up the effort. But we can certainly begin to strengthen our network models as a first step.

42 Schneier, Bruce. Secrets and Lies. New York: Jon Wiley & Sons, Inc. 2000. page 397.

- Appendix



- References

Bellovin, Steven M. "A Technique for Counting NATed Hosts." November 2002. http://www.research.att.com/~smb/papers/fnat.pdf (March 6, 2004). Story also referenced on Slashdot.org at http://slashdot.org/article.pl?sid=03/02/05/2129218&mode=thread.

Bugtraq on the SecurityFocus Home Page. http://www.securityfocus.com/ (March 6, 2004).

Counterpane. http://www.counterpane.com (March 6, 2004).

EtherApe Web Page on Sourceforget.net. "EtherApe, a graphical network monitor." http://etherape.sourceforge.net/(March 5, 2004).

Foundstone. http://www.foundstone.com (March 6, 2004).

FreeBSD web site. http://www.freebsd.org (March 6, 2004).

Hartmeier, Daniel. "OpenBSD Packet Filter – How it Started (June 2001)." June 2001. http://www.benzedrine.cx/pf-beginning.html (March 6, 2004).

"OpenBSD Packet Filter." September 9, 2003. http://www.benzedrine.cx/pf.html (March 6, 2004).

Hewlett-Packard OpenView. http://www.openview.hp.com/ (March 6, 2004).

Insecure.org Web Site. "Nmap – Free Security Scanner for Network Exploration & Security Audits." http://www.insecure.org/nmap/ (March 5, 2004).

Insecure.org Web Site. "Who is Fyodor?." http://www.insecure.org/myworld.html (March 5, 2004).

Internet Assigned Numbers Authority. "Port Assignments." March 4, 2004. http://www.iana.org/assignments/port-numbers (March 5, 2004).

Internet Security Systems. http://www.iss.net (March 5, 2004).

IP Filter – TCP/IP Firewall/NAT Software. http://coombs.anu.edu.au/~avalon/ (March 6, 2004).

Lucas, Michael. Absolute OpenBSD. San Francisco: No Starch Press, 2003.

Multi Router Traffic Grapher. http://people.ee.ethz.ch/~oetiker/webtools/mrtg/. (March 6, 2004).

NetBSD web site. http://www.netbsd.org (March 6, 2004).

Northcutt, Stephen; Zeltser, Lenny; Winters, Scott; Kent Frederick, Karen; Ritchey, Ronald W. Inside Network Perimeter Security. Indianapolis: New Riders, 2003.

OpenBSD web site: FINGERD (8) manual page. http://www.openbsd.org/cgi-bin/man.cgi?query=fingerd&apropos=0&sektion=0&manpath=OpenBSD+Current&arch=i386&format=html (March 6, 2004).

OpenBSD web site: TCPD (8) manual page. http://www.openbsd.org/cgi-bin/man.cgi?query=tcpd&apropos=0&sektion=0&manpath=OpenBSD+Current&arch=i386&format=html (March 6, 2004).

OpenBSD web site: TRACE ROUTE (8) manual page. http://www.openbsd.org/cgi-bin/man.cgi?query=traceroute&apropos=0&sektion=0&manpath=OpenBSD+Current&arch=i386&format=html (March 6, 2004).

OpenBSD web site. "PF: The OpenBSD Packet Filter." http://www.openbsd.org/faq/pf/ (March 6, 2004).

OpenBSD web site. "The OpenBSD SMP Project." http://www.openbsd.org/smp.html (March 6, 2004).

OpenBSD web site. http://www.openbsd.org. (March 6, 2004).

OpenSSH. http://www.openssh.org/ (March 6, 2004).

Rosamond, George. "A FreeBSD 4.8-RELEASE Operating System Security Checklist." June 12, 2003. http://www.sddi.net/FBSDSecCheckList.html (March 6, 2004).

Schneier, Bruce. "The Blowfish Encryption Algorithm." http://www.schneier.com/blowfish.html. (March 6, 2004).

Schneier, Bruce. Secrets and Lies. New York: Jon Wiley & Sons, Inc. 2000.

Sonnenreich, Wes and Albanese, Jason. Network Security Illustrated. New York: McGraw-Hill, 2004.

Sonnenreich, Wes and Yates, Tom. Building Linux and OpenBSD Firewalls. New York: John Wiley & Sons, Inc., 2000.

Sonnenreich, Wes. "Memoirs of an Invisible Firewall." September 6, 2001. http://openlysecure.org/openbsd/how-to/invisible_firewall.html (March 5, 2004).

van Wyck, Kenneth R. and Forno, Richard. Incident Response. Sabastopol, California: O'Reilly & Associates, July 2001.

Watchguard Web Site. http://www.watchguard.com/ (March 5, 2004).