Global Information Assurance Certification Paper

SOHO OpenBSD Intranet IMAP Server

by Donald Pitts

Option 1 - Securing UNIX Step by Step

Description of System

Background

With electronic mail there are different categories of software involved:

Category	Description
Mail User Agent (MUA)	End user client software presenting contents of mail and permitting sending mail. Examples include Eudora, Pegasus, Netscape Messenger, and Microsoft Outlook.
Mail Transport Agent (MTA)	Sends mail between machines. MTAs are typically not used directly by end users. Examples include Sendmail, Courier, Exim, Postfix, and Qmail.
Mail Delivery Agent (MDA)	Actually inserts mail into the mailbox of the end user. Examples include Procmail, Maildrop, and /bin/mail.

Role/Purpose (Intranet IMAP Server)

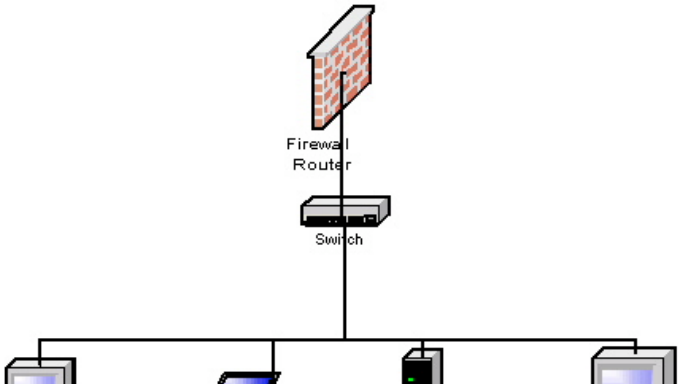
The system to be designed will service a Small Office/Home Office (SOHO) with mail via IMAP. The system will therefore run an IMAP server and will host email persistently. Users can access mail from various machines within the SOHO by accessing a central server and therefore will not need to be limited to a particular machine to have access to older mail. The number of mail users is targeted to be two to four people with distinct mail accounts. Many ISPs provide Post Office Protocol (POP) service as their default mail solution and charge additional money for IMAP service. The server being designed will provide an IMAP service locally while retrieving mail from the ISP via POP and saving the SOHO an added recurring expense. The IMAP server is designed only for intranet accessibility. External accessibility to mail could be added in the future as an enhancement with VPN technology or by migrating the server to a service network and considering running a web interface capability at that time, but is not part of the immediate system being designed.

Approach

A design goal is to attempt to provide the majority of the functionality using freeware components due to the lack of significant budgets in typical home or SOHO environments. The overriding principles are simplicity and inexpensiveness in addition to security.

Architecture

It is assumed that the installation has a constant Internet connection (i.e. DSL or cable modem) protected by a residential hardware firewall/router device which uses Network Address Translation (NAT). A static IP address will be used for the IMAP server from the private address subnetwork. The ISP provides incoming email via POP services and outgoing mail via SMTP.



Hardware (Intel compatible PC with Ethernet Access)

A generic PC compatible built from available components will be constructed or procured as follows:

Equipment	Quantity
Intel compatible CPU (i386) (Athlon XP 1700+ in this case)	1
Motherboard (First International Computer (FIC) AN11 in this case)	1
RAM Memory	256 Mb
20 Gb Hard drive	1
10/100 Mbps Network Interface Card	1
VGA display controller	1
VGA monitor for console access	1
PS/2 Keyboard	1
3.5" Floppy drive	1
CD Reader/Writer	1
Keyboard Video Mouse (KVM) switch	1

Table 1 - Hardware Equipment Inventory

Since no remote access is being provided initially, console access will be relied on when maintenance is required. The KVM switch will be useful in sharing the keyboard, monitor, and mouse between multiple systems since it won't be frequently that an administrator needs to access this machine.

The disk space required will depend primarily upon the habits of the users of the system with regard to the amount and size of mail they typically recieve and how much of it they retain on an ongoing basis for future reference. Due to the occasionally large files that people recieve as mail and may fail to remove, a rough estimate of 1Gb per person is made. It is recommended that at least 3 Gb be set aside for installation of the OS and building and installing third party add-on products from source as well as loading the OS sources for applying security patches and rebuilding the kernel. If space becomes overly critical, then the whole OS source tree does not have to be loaded to apply patches, but more effort will be necessary to only pull in the required source files. A rough overall estimate of 10 Gb should be more than enough for the normal SOHO environment. Therefore, even the smallest hard disks commonly being sold today would be more than adequate and likely older existing systems with smaller drives could be readily utilized.

Software (OpenBSD 3.1, Courier IMAP Server, Fetchmail, etc.)

OpenBSD 3.1 will be utilized as the operating system and these instructions assume that the official set of CDs has been purchased and available for this installation. This has the multiple benefits of giving back to the community to develop future versions, reducing the chance of compromise of the integrity of the OS by a third party, and the ability to reliably recreate the installation in short order. OpenBSD comes by default after installation in the strongest security posture of any OS I have worked with. This makes it a good platform to build a secure system on and reduces the amount of security work required to adequately secure the system. Many of the more elaborate lengths gone to in this setup would not be possible due to time considerations if more fundamental security work was required with the OS.

The software to be installed is as follows:

Software Package	Purpose	Version	Obtained From	Distribution file	Verification of Authenticity
OpenBSD	Operating System	3.1	Official CDs purchased - see http://www.openbsd.org/orders.html for more information		
GNU Make (gmake)	Required by Courier IMAP	3.79.1	http://ftp.gnu.org/pub/gnu/make/	make-3.79.1.tar.gz	http://ftp.gnu.org/pub/gnu/make/md5sum http://ftp.gnu.org/pub/gnu/make/md5sum.asc
Courier IMAP server (imapd)	IMAP Server	1.5.3	http://www.courier-mta.org/download.php#imap	courier-imap-1.5.3.tar.gz	Not known to be available
GNU DBM (gdbm)	Database for virtual mailboxes	1.8.0	http://ftp.gnu.org/pub/gnu/gdbm/	gdbm-1.8.0.tar.gz	http://ftp.gnu.org/pub/gnu/gdbm/md5sum http://ftp.gnu.org/pub/gnu/gdbm/md5sum.asc

Fetchmail	Retrieve mail from ISP via POP	5.9.0 (gold)	http://www.tuxedo.org/~esr/fetchmail	fetchmail-5.9.0.tar.gz	Not known to be available for gold version
Procmail	Deliver mail locally into user's mailboxes	3.22	http://www.procmail.org	procmail-3.22.tar.gz	http://www.procmail.org/procmail-3.22.tar.gz.sig http://www.procmail.org/pgp-key.html
H+BEDV AntiVir for Server for OpenBSD	Scans for viruses in incoming mail	2.0.4	http://www.hbedv.com/download/download.htm#AVServer	avobsrv.tgz	PGP Fingerprint: http://www.hbedv.com/index.html Within distribution: pgp/README pgp/antivir.gpg
NTP	Network Time Protocol server. Keeps the machine clock accurately set for use in log lines.	4.1.1a	http://www.eecis.udel.edu/~ntp/download.html	ntp-4.1.1a.tar.gz	Not known to be available
OpenSSH	Secure Shell for remote shell sessions and file transfers. Updated version with recent vulnerabilities corrected	3.4	http://www.openssh.com/openbsd.html	openssh-3.4.tgz	Not known to be available
Protector	Eliminates all but a few approved types of files as attachments within mail	1.00.6	http://www.lowth.com/protector/bin/view/Protector/DownloadVersion1006	protector-1.00.6.tgz	MD5 and sum information provided on download web page.
Gnu Privacy Guard (GnuPG)	Free replacement for Pretty Good Privacy (PGP)	1.0.7	http://www.gnupg.org/download.html	gnupg-1.0.7.tar.gz	MD5 information provided on download web page.
Advanced Intrusion Detection Environment (AIDE)	File integrity checker (i.e. Tripwire replacement)	0.9	http://www.cs.tut.fi/~rammer/aide.html	aide-0.9.tar.gz	ftp://ftp.cs.tut.fi/pub/src/gnu/aide-0.9.tar.gz.asc
Gnu Bison	Generates a program that will parse a defined language (Used in compilation of other packages)	1.35	http://ftp.gnu.org/gnu/bison/	bison-1.35.tar.gz	http://ftp.gnu.org/gnu/bison/md5sum.asc
Libmhash	Provides a uniform interface to many hash algorithms (Used by AIDE)	0.8.16	http://sourceforge.net/project/showfiles.php?group_id=4286	mhash-0.8.16.tar.gz	Not known to be available

mmencode (part of MetaMail)	Supports base64 encoding and decoding for mailing of binary data	2.7	ftp://thumper.bellcore.com/pub/nsb/	mm2.7.tar.Z	Not known to be available

Table 2 - Software Inventory

Risk Analysis

This system is not anticipated to be particularly exposed to attack because it is to be located within the interior of the network behind the firewall and will not need to run many network services. That is one of the advantages to using this type of architecture to provide mail to a small number of users. No static IP address is necessary for external visibility on the Internet and no Mail Transfer Agent (MTA) such as sendmail, Postfix, qmail, Exim, or Courier will need be run as a daemon listening on the SMTP network port. The most likely issue involving this server is its storage and propagation of mail which may contain viruses. Each user of the intranet is anticipated to have a dedicated IMAP accouint and this machine will run an IMAP server. This will not correspond to an operating system account for each user, so no one but the administrator will have shell access to the machine. Programs requiring to manipulate files relating to mail on the system will be run as a dedicated "mail" user which is not permitted to login normally to reduce the odds of an attacker gaining access to "root" should they compromise the system through a mail service.. The other potential exposure is revealing POP or IMAP user account names or passwords to unauthorized third parties.

The services required are as follows:

Incoming (Listening) Services	Source IP (Remote)	Source Port (Remote)	Destination Port (Local)	Valid Users	Remarks
IMAPS (IMAP over SSL)	Local subnets	Ephemeral ports (> 1023)	TCP port 993	All internal SOHO users	Versus the standard non-SSL IMAP which uses TCP port 143
Syslog	localhost over the loopback interface	Ephemeral ports (> 1023)	UDP port 514	Local processes within the IMAP server	Not necessary if central log server is already available and running within the SOHO.
Network Time Protocol (NTP)	Local subnets	UDP port 123	UDP port 123	All internal SOHO machines	The time of this machine will be made available to the rest of the SOHO.
Console OS Login	N/A	N/A	N/A	SOHO system administrators	

Table 3 - Incoming/Listening Network Services

Outgoing (Client) Services	Source Port (Local)	Destination IP (Remote)	Destination Port (Remote)	Valid Users	Remarks
Post Office Protocol (POP) version 3	Ephemeral ports (> 1023)	POP Servers (i.e. *<pop1.sans.org>*, *<pop2.sans.org>*)	TCP port 110	Fetchmail process running as the automated user "mail"	Retrieve mail from ISP for each local user. If accounts are serviced on different servers then more than one server will need to be authorized.
Network Time Protocol (NTP)	UDP port 123	NTP Servers (i.e. *<ntp1.sans.org>*, *<ntp2.sans.org>*, and *<ntp3.sans.org>*)	UDP port 123	NTP daemon process running as user "root"	Reqeust time from a group of pre-chosen time servers from those available within the local time zone.
Hyper Text Transfer Protocol (HTTP)	Ephemeral ports (> 1023)	H+BEDV Servers	TCP port 80	Anti-virus process running as user "mail"	Dynamically update virus engine and signatures. The IP address was determined by activating the logging feature of the packet filter for HTTP traffic and then determining the block of network addresses assigned to HBEDV.
Domain Name Service (DNS)	Ephemeral ports (> 1023)	Name Server(s) (i.e. *<192.168.0.1>*)	UDP port 53	Local processes within the IMAP server	Hostnames must be resolved for various server names. This example depends upon the local firewall router providing redirected DNS service. In any case, IP addresses must be provided because of the chicken and egg

				requiring resolution of host names.	problem with resolving a hostname for a DNS server.
Simple Mail Transfer Protocol (SMTP)	Ephemeral ports (> 1023)	SMTP Servers (i.e. <*smtp.sans.org*>)	TCP port 25	Sendmail or any other process needing to deliver mail externally.	These would likely correspond to servers provided by the ISP, but could be local network machines, if already present.

Table 4 - Outgoing/Client Network Services

Risks within the system are assessed as follows:

Risk	Mitigation
Viruses embedded in arriving mail	• Anti-virus software will be deployed to scan incoming mail as it arrives with automatically updated virus signatures and all but a few types of attachments are not permitted to be received.
ISP POP passwords are intercepted by unauthorized users	• SSL enabled POP access will be used if available from the ISP. • Switched architecture will be fielded to limit the ease of snooping by insiders. • Only POP accounts within the immediate ISP will be accessed and not those from a remote second ISP to limit the chance of snooping by outsiders.
Outside SOHO threats of compromise	• Boundary firewall is required. • Host-based firewall is implemented for local protection. • Minimal number of network services will be available. • Employed network services are stripped of obvious identifying characteristics with potential fingerprinting uses.
Insiders within the SOHO accessing each others mail without authorization	• Separate IMAP accounts will be issued to each SOHO user. • The IMAP service will be SSL protected. • Users will be encouraged to not store their IMAP passwords within their IMAP clients to reduce the odds of password compromise.
Insiders within the SOHO compromise the integrity of the IMAP server	• Normal users will not be provided any OS accounts on the IMAP server. • Mail related activity within the server will be implemented with normal user accounts which lack any root privilege. • The IMAP server will listen on a non-privileged port and the redirection capability of the host-based firewall will be employed to accept IMAP requests on the standard TCP port.
Rogue system administrator of SOHO	• Some measure of background investigation and scrutiny of personal references to include criminal history is warranted.
Physical compromise of the IMAP server	• BIOS protections will be activated to limit electronic access - add password protection and boot only from hard drive. • Locate the IMAP within a limited access area of the facility if such an area is available.

Table 5 - Risks and Mitigation

Approach

Source or Binary (Ports/Packages)?

OpenBSD has the concept of ports and packages where different software has been tailored to work within OpenBSD specifically. Packages can be thought of as similar to packages within Solaris or RPMs within Linux. Ports will usually automatically download the relevant source code from the Internet, compile, link, and install the relevant files on the system.

We will be building all of the software we install from source, when this is possible. This effort is undertaken with the following benefits in mind:

- Can be assured of obtaining the most recently available version of the software when binaries and pre-packaged solutions may lag

somewhat
- Custom tailoring of package during compilation process
  - Omitting unnecessary components for this specific circumstance
  - Embedding critical or constant options to limit insecure/undesired functionality
  - Modification of source code to reduce the likeliehood of fingerprinting
- Improves understanding of components and options available
- Increases confidence in trustworthiness of the resultant package
- Archiving the obtained source code provides the option of self sustinance should a software package become unmaintained or, even worse and unlikely, unavailable.

This is not to say we have completely avoided the possibility of dangerous trapdoors or other malicious embedded content. Therefore we will be verifying any provided checksums and other available safeguards to limit this risk.

Taking this approach requires the availability of an experienced systems administrator with some minimal programming knowledge. Taking the alternate approach of going with pre-built packages or binaries or even giving up and paying the ISP for equivalent service may be necessary should someone of this caliber not be available to the SOHO. In our circumstances, we assume the availability of someone of sufficient technical ability to adequately support the SOHO.

Risk assessment should be performed to help determine whether a particular capability is worthy of any potential vulnerabilities it may introduce.

Which Version?

Most packages offer two or more "current" versions. One is often marked the "stable", "production", or "gold" version. This is usually meant to denote that while all of the neatest features may not be contained therein, it has been tested extensively and thought to be reliable. Many packages offer a "beta" or "development" version which is targeted toward early adopters and developers. Choosing the most appropriate version can be a very subjective decision and one that is not made easier by the variations with which these names are utilized by the various development teams. Stability should be factored into the decision, but sometimes features and even security issues can make the "beta" the only way to go. This is the age old "it depends" answer that means you will need to decide for yourself based upon the state of the available releases at the time of installation.

Step by Step Guide

Information Needed Beforehand

Many pieces of information will be required as part of the setup of the server that will likely vary between different installations. The administrator performing the installation will need to acquire this information and substitute it for placeholders throughout the procedure. Information expected to vary between installations is denoted such as this with angled brackets and italics: <*example*>. The domain name of <*sans.org*> will be used when a domain is needed, but this should be obviously substituted with the domain name procured by the SOHO.

Placeholders	Explanation
<*192.168.0.0/24*>	Local SOHO network
<*192.168.0.255/32*>	Local broadcast address for SOHO network
<*192.168.0.1*>	At least one, but preferably two servers providing Domain Name Service (DNS). These can be within the local network, if available, or correspond to servers supplied by the ISP. This value is also used for the default gateway, since it corresponds to the address of the firewall router.
<*192.168.0.78*>	IP address assigned to the single network interface on the IMAP server being setup. This should be outside the range of IP addresses being dynamically assigned via DHCP by the firewall router, assuming that has been activated.
<*ntp1.sans.org*>, <*ntp2 sans.org*>, <*ntp3.sans.org*>	3 or more Network Time Protocol (NTP) servers. If this is the first machine to use NTP, then they will likely be various public NTP servers within the local region. Once many machines within the domain need to use NTP, then local NTP servers within the domain will need to be setup.
<*pop1.sans.org*>, <*pop2.sans.org*>	One or more POP servers corresponding to the collective set of POP user accounts that are to be periodically queried from this new IMAP server.
<*smtp.sans.org*>	The SMTP server likely provided by the ISP to send email to be delivered. Because the mail on this system is for virtual users, all will be delivered via this method.
<*YourAdmin@sans.org*>	The email address of the system administrator.
<*YourState*>	Two letter abbreviation for the state where the SOHO is located.
<*YourCity*>	Name of the city where the SOHO is located.
<*YourOrganizationName*>	Name of the SOHO organization.
<*popusern*>	The username of a user *n*'s POP account

<realpassn>	The password of a user *n*'s POP account
<imapusern>	The username of a user *n*'s IMAP account

Table 6 - List of Placeholders

Obtaining Software and Latest Patches

Retrieve the software shown in Table 2 on a separate machine which is already adequately secured and networked, if possible. Also, retrieve the OpenBSD 3.1 patches combined within a tar file from http://www.openbsd.org/errata.html. Verify the fingerprint of the software and the authenticity of the fingerprint, where possible. Most commonly an MD5 one-way hash is used to create and verify a fingerprint. Known places to download fingerprints for the packages is shown in Table 2. Typically a separate file can be downloaded which contains a fingerprint. Then the person obtaining the software can independently generate a corresponding fingerprint for the software distribution downloaded and then verify it matches that indicated by the supplier. The cksum and sum methods use simple Cyclical Rendundancy Checks (CRCs) which can be easily matched by engineered versions of the distribution file engineered with filler to provide the same checksum as the original. Therefore, MD5 would be the preferred method of verification, but all available options should be employed within reason. Another, stronger alternative is to sign the distribution file with GnuPG or Pretty Good Privacy (PGP).

```
md5 distribution.tar.gz
cksum distribution.tar.gz
sum distribution.tar.gz
```

Some sites go to the extra step of signing their fingerprints or even the distribution file itself which can then be checked for authenticity using GnuPG or PGP.

```
gpg --verify distfile.tar.gz.asc distfile.tar.gz
gpg --verify pkg-md5.asc
```

If you receive an error during verification indicating that the public key was not found, then:
Retrieve the public key from a public key server or other source (i.e. http://www.us.pgp.net/pgpnet/pks-commands.html)
Load the public key into the local key ring: gpg --import developer.key
gpg --sign-key 0x*<fingerprint hex string>* (indicate that casual checking has been done)

Should no method of verification be available for a given package, then common sense should be used to be on the lookout for obvious signs of tampering. For stable versions that are a few months old or more, the dates on files would not normally be recent. Dates on files can be adjusted by hand, so this is not a foolproof check. Any other characteristics such as size of the distribution file should be compared against any independent corroborating information from the source to lend credence that the file is likely legitimate.

Starting Installation

Network cables should not be initially connected to the system being constructed to protect it against attack until sufficient defenses are erected and corrective action for known vulnerabilities has been taken.

On an existing system connected to the Internet with a CD burner, retrieve the software packages and OS patches indicated earlier, verify any checksums, scan for viruses, and burn a CD with the contents. Alternatively, the authenticity can be checked within the system being built as long as the needed data to verify are stored on the CD as well.

Importing files from DOS can cause extra carriage returns which are shown as "^M" at the end of each line if you view the file with the vi editor. These can cause problems such as for scripts. To remove the extra carriage returns try something similar to the following example:

Action	Explanation
cat dos_file | tr -d "\r" >unix_file	Delete carriage returns from standard input which contains the DOS file and redirect the corrected version of the file to a new filename. Now the file is prepared for use within Unix.

Whenever a configuration file is edited, it is very helpful, at least in the beginning, to make a backup copy of the file before beginning to modify it. This helps to provide traceability to exactly what changes have been made to help with troubleshooting and understanding. Additionally any edits to a file can be started over without the need to reinstall the system. So, if the /etc/rc.local file was to be modified, for example:

```
cp  /etc/rc.local   /etc/rc.local.dist
vi  /etc/rc.local
```

Root can typically overcome any restrictions, but when saving a file within the vi editor an error can be issued when the permissions do not allow a write operation by root ("Read-only file, not written; use ! to override."). The vi editor allows the "w!" or "wq!" operation to be requested which will overcome the permissions limitation. This can be useful when it is desirable to have a very restrictive set of permissions on a file, but some changes must be made. Since root can change permissions on any file, this is not circumventing any security, but merely making things more convenient.

BIOS - Booting from CD

Some systems can boot from CD. Insert OpenBSD 3.1 CD #1 into the CD drive and begin the system. Should it not boot from the CD, then restart it and press the proper key code to enter the BIOS (sometimes the Delete key). Set the system to boot from the CD drive if this is possible. Save the new BIOS settings and reboot. Should the system not support booting from CD, see the instructions with the OpenBSD jewel case concerning how to build a bootable floppy disk. The exact steps vary significantly due to different BIOS manufacturers and versions. See the documentation for your particular BIOS to resolve any issues

Disk layout:

The system being built will exclusively run OpenBSD and will NOT need to dual boot or have an alternate OS available. Therefore the available disk space only needs to be assigned for use by OpenBSD. Partitions are typically mounted at a particular location within the system's directory structure and encompass all files from that location, called the mount point, and down to include the entire subtree to only exclude other partitions which may have a mount point within this subtree. If the original partition holding the mount point.for a subsequently mounted partition contains files or even a more involved directory structure from that point down, then those files will not be available unless the subsequent partition is ever unmounted. This is the normal behavior and will correspond to the way this system will function. There is a kernel option to make these "hidden" files somewhat accessible, but due to the complexity and the non-standard aspect of this it will not be utilized.

The chosen disk layout is an area where the System Administrator can be creative and has some license to provide disk space in many different ways as long as enough space is provided for all necessary activities. Some administrators prefer to limit the number of partitions to a minimal number and others like to create many more specialized ones. Partitions work such that if one is filled, then other partitions are not affected. One tradeoff is that fewer partitions lack the ability to segregate data to avoid affecting other potentially unrelated services. Another is the ability to set special options on a partition restricting the actions permitted with files within the partition such as making its files read-only or preventing set-UID functionality. Typically the swap space is made twice the amount of memory, although this becomes less advised once a considerable amount of memory is present (i.e. Gigabytes). The following are recommendations for allocating disk space for partitions:

Partition	Mount Point	Suggested Size
a	/	256 Mb
b	swap	256 Mb
d	/var	512 Mb
e	/usr	2 Gb
f	/mail	5 Gb (or about 1 Gb per user)

Action	Explanation
d a	Delete the existing partitions (note that for historical reasons, c is a reserved partition always corresponding to the whole disk).
d b	
d d	
d e	
a a	Add our own partitions of the sizes shown above.
a b	
a d	
a e	
a f	
w	Write out the disk label to disk to make our partitions permanent.
q	Quit editing partitions.

Continuing installation:

Action	Prompt	Explanation
no	config network?	This will be setup by hand and connected up later.
<admin pass>	initial pass:	Choose a strong password by incorporating special symbols, numbers, as well as a combination of upper and lower case letters.
no	X Window	Because this is a server machine with no normal end users, we will not be running a graphical Windowing environment to avoid the vulnerabilities that come with that.

	System?	
C	Install from?	CD-ROM disc
cd0	which cdrom:	Device name of CD drive. May vary based upon hardware.
3.1/i386	directory within CD:	Directory within the CD to load the OS from.
base31, etc31, misc31, comp31, man31, bsd	which packages:	Follow the instructions to select the packages not relating to X11 or games. Not enough granularity is provided here to be able to significantly exclude non-applicable elements of the system through the package process.
no	extract more sets?	
GMT	timezone:	Using GMT as the timezone can be useful as a standard if ever fielding systems in multiple timezones, which is not likely in the short term for our installation. It can also be useful when tracking down anomalous events involving external equipment of other organizations which may not be in the same timezone. Enter a local timezone if you prefer that.
halt	#	Remove CD from drive and restart the system

Begin to tailor the system:

Action	Explanation
Login as root	Use the password setup during installation..
crontab -e	Edit the crontab for the root user.
Comment out (prepend with a pound sign "#") or remove the "/usr/libexec/atrun" line.	This is just another effort to simplify the system. The at system is similar to cron and is unnecessary at this time.

Create the file /var/cron/allow to contain only the following so only root can start processes from cron:

```
root
```

Create the file /etc/myname with the following to define the name of this machine:

```
mail
```

Create the file /etc/mygate with the IP address of the firewall router for the local subnet. This defines the route of last resort where network packets are sent when there is no specific rule indicating where it should go:

```
192.168.0.1
```

Setup the network interface, which in this case is dc0, with the IP address we assign and the corresponding subnet mask for the local network. Notice that the extension of the filename is the name of the network interface. Performing an "ifconfig -a" can help to determine which network interfaces are available. Creat the file /etc/hostname.dc0 with the following:

```
inet 192.168.0.78 255.255.255.0
```

Ensure an entry is in the /etc/hosts file for this machine:

```
192.168.0.78  mail  mail.<sans.org>
```

Edit the /etc/rc.conf with the following changes:

```
portmap=NO
inetd=NO
sendmail_flags=NO
check_quotas=NO
sshd_flags=NO
cron=YES
ntpd=YES
imapd=YES
pf=YES
```

Even though inetd is being disabled within the rc.conf, it is still useful as a defense in depth move to limit what would be started if it was ever run. Edit the /etc/inetd.conf and comment out all lines. The ones that should need to be commented are: daytime, time, rstatd, rusersd, comsat, ident

Edit /etc/resolv.conf to contain this one line which causes the system to send DNS lookups to the address shown. This has to be an IP address and not a hostname that would require name resolution.

```
nameserver 192.168.0.1
```

Activate the new network settings by running the command "sh /etc/netstart" or you can reboot the machine with "sync; reboot".

Removable media

Add entries to the end of the /etc/fstab (file system table) corresponding to the CD and floppy drives to simplify mounting of the corresponding media without the need for additional flags. The commands "mount floppy", "umount /floppy", "mount /cdrom", and "umount /cdrom" can be used to mount and unmount media.

```
/dev/cd0a /cdrom cd9660 ro,noauto 0 0
/dev/fd0a /floppy msdos rw,noauto 0 0
```

Then create the mount points needed to mount CD or floppy media:

Action	Explanation
mkdir /cdrom	Create a mount point where the CD can be mounted.
mkdir /floppy	Create a mount point where the floppy can be mounted.

Verify that the path for the root user (and other users subsequently created) does not contain the current directory indicator ('.' - period). This helps to prevent an attacker from staging executable files of their creation of a commonly used name that could possibly be run instead of the intended program.
echo $PATH

Add /usr/local/sbin and /usr/local/bin to PATH within /root/.cshrc for the root user. Also remove /usr/X11R6/bin from PATH since the X11 windowing system will not be loaded or utilized.

Original Lines	Modify Original Lines as shown
set path=(/sbin /usr/sbin /bin /usr/bin **/usr/X11R6/bin**)	set path=(/sbin /usr/sbin /bin /usr/bin **/usr/local/bin /usr/local/sbin**)

Set-UID / Set-GID files

Executables are at times marked with special permission indicators that do more than just decide who can access or manipulate a file or directory. Two such indicators are the Set-UID and Set-GID flags. These flags are used to indicate to the OS that when such a file is executed, it runs as if the owner of the file ran it - in the case of SetUID, or as if the user running the executable is a member of the group marked on the file. This is used to permit system commands to perform some action that the actual user will likely not have the needed permissions to achieve the objective. This mechanism has been taken advantage of by attackers in the past. The danger is that a user can modify the environment in which one of these binaries run in and mislead it into performing some action not in the better interest of the integrity of the system.

File permissions are often handled in octal. An octal digit can range from 0 - 7. The first digit is a 0 to signify the number is in octal form. The second digit contains the Set-UID and Set-GID indicators as well as the sticky bit. The high-order bit (4 decimal) is the Set-UID, the middle bit (2 decimal) indicates Set-GID, and the low-order bit (1 decimal) is the sticky bit. The last three octal digits correspond to the access permissions: read, write, and execute for the owner (user), group, and public (other) collections.

Action	Explanation
find / -perm -04000 -exec ls -l {} \; >suid_owner_progs	Locate all of the Set-UID programs within the system irrespective of the other file permission settings.
find / -perm -02000 -exec ls -l {} \; >sgid_owner_progs	Locate all of the Set-GID programs within the system irrespective of the other file permission settings.

Look through each of the programs listed and perform "chmod u-s,g-s <*file*>" for those files whcih don't appear to require these permissions.

Change the default Time-To-Live (TTL) for packets originated by this machine from the OpenBSD of 64 to the Solaris default of 255 as an effort to thwart OS fingerprinting efforts. Note that many characteristics of a system can be used by an attacker in OS fingerprinting efforts, so this will

not prevent someone from determining our OS. Hopefully this will slow someone down or add doubt to their prognosis.

Add to /etc/sysctl.conf:

```
net.inet.ip.ttl=255  # Modify default TTL from 64 to 255 to emulate Solaris
```

Scrub Users and Groups

Remove unneeded users and groups from the system. Being as tidy as possible helps to limit the avenues through which the system may be attacked.

Create the following script: clean_accts:

```
#!/bin/sh
# clean_accts - Remove user accounts and groups which are not needed for this installation.

for users in popa3d uucp www named proxy
do
  userdel $users
done

for groups in games popa3d www named proxy dialer news
do
  groupdel $groups
done
```

Perform the following steps to remove the unneeded elements:

Action	Explanation
chmod u+x clean_accts	Make the script executable.
./clean_accts	Run the script to remove unnecessary users and groups from the system.
rm clean_accts	Remove the script since it is only necessary during installation.

Apply Patches to OS and Rebuild Kernel

Patches are not supplied in binary form as they are on most systems. This means that the original source code must first be available on the system and then patches to the source code are applied. Then the modified source code is compiled and the newly built binaries installed. Typically some patches must modify the core part of the OS known as the kernel. This requires recompiling and installing a new kernel binary executable.

Action	Explanation
Insert OpenBSD 3.1 CD #3 into the CD drive	This CD contains the OpenBSD source.
mount /cdrom	Make the contents of the inserted CD media available within the /cdrom directory.
cd /usr/src	This is the recommended location to store the OS source.
tar xzf /cdrom/src.tar.gz	This will take some time depending upon the system! Unbundle the OS source code on the CD into /usr/src. This will create a number of subdirecories containing source code.
umount /cdrom	The CD is no longer required.
Eject the CD from the drive	
tar xvzf 3.1.tar.gz	Unbundle the tar file (should have been downloaded previously from http://www.openbsd.org/errata.html) containing the source patches to the OS within /usr/src. A new directory "3.1" will be created containing various subdirectories and files.

There are 14 patches to OpenBSD 3.1 at the time of this writing. Patches can be common to all architectures or specific to a particular one. All of the current patches are applicable to all architectures and are located within the /usr/src/3.1/common directory. Each patch contains comments at the beginning showing the steps required to successfully apply and install the patch onto a system. These instructions have been taken and used to create a script (apply_patches) to automate this effort. This reduces the chance of a manual mistake and simplifies this task which is especially useful should a system need to be rebuilt from scratch.

Create the script apply_patches within /usr/src as follows:

```
#!/bin/sh
# Derived heavily from the instructions included within each of the 3.1 patches.
```

```
SRC=/usr/src
PATCH_DIR=$SRC/3.1
COMMON_PATCH_DIR=$PATCH_DIR/common
KERNEL=GENERIC
KHOME_DIR=$SRC/sys/arch/i386
KCONF_DIR=$KHOME_DIR/conf
KCOMP_DIR=$KHOME_DIR/compile/GENERIC

if [ ! -d $PATCH_DIR ]; then
  echo "Error: Directory $PATCH_DIR does not exist"
  exit 1
fi

# Patch 1
cd $SRC
patch -p0 <$COMMON_PATCH_DIR/001_sshafs.patch
cd usr.bin/ssh
make obj
make cleandir
make depend
make && make install

# Patch 2
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/002_sudo.patch
cd usr.bin/sudo
make -f Makefile.bsd-wrapper obj
make -f Makefile.bsd-wrapper cleandir
make -f Makefile.bsd-wrapper
make -f Makefile.bsd-wrapper install

# Patch 3
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/003_fdalloc2.patch

# Patch 4
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/004_sshbsdauth.patch
cd usr.bin/ssh
make obj
make cleandir
make depend
make && make install

# Patch 5
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/005_httpd.patch
cd usr.sbin/httpd
make -f Makefile.bsd-wrapper obj
make -f Makefile.bsd-wrapper cleandir
make -f Makefile.bsd-wrapper
make -f Makefile.bsd-wrapper install

# Patch 6
# Load latest SSH package

# Patch 7
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/007_resolver.patch
cd lib/libc
make obj cleandir depend
make && make install

# Patch 8
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/008_mod_ssl.patch
cd usr.sbin/httpd
make -f Makefile.bsd-wrapper obj
```

```
make -f Makefile.bsd-wrapper cleandir
make -f Makefile.bsd-wrapper depend
make -f Makefile.bsd-wrapper
make -f Makefile.bsd-wrapper install

# Patch 9
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/009_ktrace.patch

# Patch 10
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/010_isakmpd.patch
cd sbin/isakmpd
make obj
make cleandir
make depend
make
make install

# Patch 11
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/011_pppd.patch
cd usr.sbin/pppd
make obj cleandir depend
make && make install

# Patch 12
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/012_xdr.patch
cd lib/libc
make obj cleandir depend
make && make install

# Patch 13
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/013_ssl.patch
rm -fr /usr/obj/lib/libssl
cd lib/libssl
make obj cleandir
make -f Makefile.bsd-wrapper prereq
make -f Makefile.bsd-wrapper includes
make -f Makefile.bsd-wrapper
make -f Makefile.bsd-wrapper install

# Patch 14
cd $SRC
patch -p0 < $COMMON_PATCH_DIR/014_scarg.patch

echo "Now rebuilding your kernel..."
cd $KCONF_DIR
config $KERNEL
cd $KCOMP_DIR
make
cp /bsd /bsd.old
cp bsd /bsd

echo
echo "Reboot your system now."
```

Now we are ready to actually apply the patches, rebuild the kernel and reboot.

Action	Explanation
Follow the directions in any new patch files manually or update the apply_patches script for each new patch and backup the script.	
chmod u+x apply_patches	Make the new script executable.
./apply_patches	Actually apply the latest patches to the OS and kernel.

	This will also take quite some time to complete on most systems!
ls -l /bsd*	Examine the sizes of the new and old kernel file to verify the new kernel is a reasonable size and likely valid.
sync; reboot	Reboot the system to pick up the new kernel.

Create a User Account for an Administrator

Be sure to create any administrator accounts to be in the wheel group within the /etc/group file so they become root with "su root". This automatically creates a home directory within /home and assigns a User ID. Then assign a password. Note that <*Name of User*> typically corresponds to the full name of the user and that <*username*> is the name of the account to be created.

useradd -m -c "<*Name of User*>" -G wheel <*username*>
passwd <*username*>

Warning Banner for Logins

OpenBSD does not support the standard /etc/issue for a statutory warning as used on Solaris or similar file-based mechanisms to control the message shown to the user prior to receiving a login prompt. The /etc/gettytab file has an "im" setting on the default terminal class that can be modified to control the initial message. Some other BSD flavors appear to support an "if" setting which can define a particular file to contain the initial message. This capability is not currently documented to exist within OpenBSD and without it, a full warning screen cannot be provided. Therefore a more limited warning is generated.

By default, the initial message or banner on the console login screen reveals that OpenBSD is the OS. While does not pose a remote threat, it does reveal more than necessary to anyone who has gained physical access to the server. The "im" setting corresponds to the initial message. The original setting looks something like this:

OpenBSD/i386 (mail) (ttyC0)

login:

Change /etc/gettytab as follows:

Original Lines	Modify Original Lines as shown
default:\ :np:im=\r\n**%s/%m (%h) (%t)**\r\n\r\n:sp#1200:	default:\ :np:im=\r\n***WARNING* Only authorized users may proceed.**\r\n\r\n:sp#1200:

Note that various special variables are available that when used are replaced with an associated dynamic value such as the name of the host. The gettytab special variables referenced in the default initial message equate to the descriptions shown below. To learn more about the various settings, consult the gettytab man page.

Gettytab Special Variable	Description	Likely Value
%s	Name of the OS	OpenBSD
%m	Type of machine architecture	i386
%h	Hostname of machine	mail
%t	Name of tty	ttyC0

The modified login screen will look like:

WARNING Authorized administrators may only proceed

login:

Notify Users of Failed Logins

Users are normally unaware of invalid attempts made by other individuals to gain unauthorized access to their account. OpenBSD has the capability to display information once a user account is properly logged into indicating the time and quantity of failed logins since the previous valid login session.

Action	Explanation
touch /var/log/failedlogin	Activate the failed login notification feature.

An example of what this will look like during the next valid login:

```
Password:
There have been 2 unsuccessful login attempts to your account.
Last unsuccessful login: Sat Aug 24 21:11:54 on ttyC0
Last login: Sat Aug 24 21:11:29 on ttyC0
```

The indication of the failed logins is then removed and a subsequent valid login with no further interspersed failed logins will return to looking normal:

```
Password:
Last login: Sat Aug 24 21:11:59 on ttyC0
```

Packet Filter (pf)

OpenBSD ships with a built-in stateful firewalling capability called pf (packet filter). OpenBSD switched to this product from IPFilter with OpenBSD 3.0, so it is still relatively new within the OS. This facility can be used to build a dedicated firewall or to effectively function as a host-based firewall as is the case here. For those more familiar with Linux, it can be thought of as similar to IP Tables and for those of more of a PC orientation, Zone Alarm and Tiny Peronal Firewall would be similar products to the intended usage.

The main purpose is to provide a mechanism by which packets can be examined to decide whether to permit given packets to pass based upon various characteristics such as which interface they arrived in on, the direction they are travelling, the protocol, TCP flags when applicable, and the source and destination IPs as well as the related ports. Network Address Translation (NAT) is also supported where IP addresses are modified as the pass through.

Packet filter also supports the ability to redirect packets arriving on one port to a different IP address and port. For this system, IMAP is accepted at the external address on the dc0 interface at the standard IMAPS port of 993 and redirected to the localhost loopback address and a port 11993 based upon nat.conf. The loopback interface is only accessible within a machine, so this effectively keeps access of this port unavailable to others on the network. This capability is utilized to provide a generic mechanism that can be applied to any application to avoid the need to run as root for the sole purpose of opening a privileged port (below 1024). Many applications provide a means of starting as root opening privileged ports and then switching to a non-administrative user later and handing off the port as another way around the problem. Using packet filter is useful since it centralizes any sensitive code which might introduce vulnerabilities instead of trusting many different applications to properly implement this. Additionally it is very useful in cases where the application does not provide such a mechanism itself.

Note that translations and redirections indicated in /etc/nat.conf are applied before the rules within /etc/pf.conf are utilized. Therefore, any ports referenced in the pf.conf that are dealt with in nat.conf should be the redirected port values and not the original ones arriving at the relevant network interface.. A similar situation applies should any IP address translation be indicated by nat.conf.

Several different mechanisms are available to self-document the rules and also support centralizing any references to promote flexibility for possible future changes. Hostnames are resolved via the standard mechanisms such as /etc/hosts and DNS, although networks will not resolve from /etc/networks. Hostnames are used instead of hardcoded IP addresses. There are some pros and cons to this. The negative aspect is oriented toward the potential of hostnames resolved through DNS to be corrupted with some alternate, incorrect IP address. While this is a possibility, it is far outweighed by the fact that the most viable alternative is to not restrict by IP address. If that approach is taken, then the new server is open to attack from anyone. The other alternative is to manually pre-resolve any IP addresses and hardcode the addresses within the packet filter rules. This is more secure, but is inflexible. It particularly becomes a problem when referencing public servers which are controlled by external entities which may have their IP addresses changed periodically.

One more critical issue with using DNS resolved names within the packet filter capability is that OpenBSD as it ships initially loads pf with special rules blocking all incoming and outgoing traffic before the network is activated. Right after the network is activated, then the final rules are loaded from /etc/pf.conf and /etc/nat.conf. This means that no network traffic can normally take place while the final rules are loaded. This is corrected by altering the initial pre-network rules to permit outgoing DNS lookups to be performed. The extra rule added is not as strictly locked down as is implemented in the final rules. This is a tradeoff to prevent this initial rule from impacting maintenance should the valid name servers or other characteristics change. The initial rules are very quickly replaced with the final rules, so this is a very small window and it is only permitting outgoing traffic in this manner with related stateful incoming traffic permitted.

The protocol abbreviations defined within /etc/services can be used in lieu of the actual port numbers. Additional customized entries can be easily made.

Lastly, symbols can be defined with a corresponding value within the packet filter definition files and then referenced later within the same file by prepending a dollar sign to the symbol name.

Normally the last rule that matches the characteristics of the current packet is applied. The quick keyword is utilized heavily to cause any rule that matches containing this keyword to be applied to the packet even though more rules may follow that match.

Packet filter can be stateful and subsequently related packets are permitted based upon its own internal state table for packets matching a rule with the "keep state" phrase. Use of this feature simplifies the rules while more strictly limiting the holes which have to be opened. TCP flags can be specified and these rules specify that only the SYN flag will appear out of all of the possible flags to correspond with the typical initial packets for a starting TCP connection ("flags S"). Most rules are specified to only be valid for a particular interface. This helps to protect against new

interfaces being added later without updating the filter rules.

Rules can be further simplified by collapsing multiple rules which are very similar into one rule. This can be accomplished in many cases by specifying a list of hosts or ports separated by commas and bordered by braces on both sides (i.e. "{ one, two, three }"). The following use of packet filter attempt to implement the network services and capabilities identified earlier in Tables 3 and 4.

Logging all bad packets is a good start, but can be refined as experience is garnered with the network to reduce the occurence of false negatives. Rules were added to block, but not log some Windows broadcast traffic within the local network. This may or may not be appropriate depending upon the presence of Windows machines and their behavior within the SOHO network.

A commented out rule is also provided to permit outgoing HTTP and HTTPS web requests. This can be useful to retrieve updated software and patches using the textual web browser lynx, but can be easily activated only when needed. The command "pf -R /etc/pf.conf" will reload the rules. By leaving this off as a default, it makes it much harder for any malicious software that might somehow penetrate this machine to afterwards contact anyone or anything else externally.

Create a script in /usr/bin for convenience called "pflog" to permit access to the packet filter logs. This can be fed into the tail command (i.e. "pflog | tail") to see the last few recent packets that were logged.

```
#!/bin/sh
# pflog - display the packet filter logs
tcpdump -n -e -ttt -r /var/log/pflog
```

Finish setting up the pflog script:

Action	Remarks
chgrp wheel /usr/bin/pflog	Fix the group so any administrator can run this.
chmod 550 /usr/bin/pflog	Make the script executable
rehash	Cause shell to rebuild internal structures to find the new script.

Modify /etc/rc as shown to permit DNS lookups to be performed to resolve hostnames when loading the final version of the packet filter rules:

Original Lines	Modify Original Lines as shown
if ["X${pf}" != X"NO"]; then RULES="block in all\nblock out all" case `sysctl vfs.mounts.nfs 2>/dev/null` in	if ["X${pf}" != X"NO"]; then RULES="block in all\nblock out all" **RULES="$RULES\npass out proto udp from any to any port = domain keep state"** case `sysctl vfs.mounts.nfs 2>/dev/null` in

Define any needed IP addresses that are not available via DNS within the /etc/hosts. For example:

```
<192.168.0.1>      namesvr                 # Firewall router providing redirecting DNS name services
<192.168.0.78>     mail mail.<sans.org>    # This IMAP server
```

Add the following lines to /etc/services:

```
imaps           993/tcp                 # Secure IMAP (SSL)
limaps          11993/tcp               # Non-privileged (> 1023) local loopback port for secure imap
```

Change /etc/pf.conf to contain the following:

```
# Network interface on the IMAP server connected to the local network switch
# Can determine with command "ifconfig -a | more"
if = "dc0"

# This server
imap_svr = "mail"

# Defines the network corresponding to the local set of machines within the SOHO
localnet = "<192.168.0.0/24>"
local_broadcast = "<192.168.0.255/32>"

# Definition of authorized machines to provide various services to this machine
ntp_svrs = "{ <ntp1.sans.org>, <ntp2 sans.org>, <ntp3.sans.org> }"
name_svrs = "{ namesvr }"
pop_svrs = "{ <pop1.sans.org>, <pop2.sans.org> }"
smtp_svrs = "{ <smtp.sans.org> }"
hbedv_svrs = "213.x.x.0/28"     # Note: IP sanitized

# Private addresses that are not routable on the Internet
```

```
# Since our local network uses 192.168, we need to be careful about where we use this.
private_addrs = "{ 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/24, 255.255.255.255/32 }

# Ephemeral, non-privileged ports are those above 1023 which are typically used for the source port on client
connections
ephemeral = "> 1023"

# Scrutinize  and cleanup packets for oddities and remove fragmentation
scrub in all

# Don't constrain loopback traffic (internal to the IMAP server)
pass in log quick on lo0 all
pass out log quick on lo0 all

# Authorized incoming traffic
pass in quick on $if inet proto tcp from $localnet port $ephemeral to localhost port = limaps flags S keep state
pass in quick on $if inet proto tcp from $localnet port = ntp to $imap_svr port = ntp keep state

# Authorized outgoing traffic to specific destinations
pass out quick on $if inet proto udp from $imap_svr port = ntp to $ntp_svrs port = ntp keep state
pass out quick on $if inet proto udp from $imap_svr port $ephemeral to $name_svrs port = domain keep state
pass out quick on $if inet proto tcp from $imap_svr port $ephemeral to $pop_svrs port = pop3 flags S keep state
pass out quick on $if inet proto tcp from $imap_svr port $ephemeral to $smtp_svrs port = smtp flags S keep state

# Prevent attempts to talk to any unused private addresses
# No private address restrictions are necessary prior to here because all IP addresses are limited severely
block out log quick inet from any to $private_addrs

# Authorized outgoing traffic to unrestricted destinations
# These must appear after the private address restrictions
#
# Can uncomment this out temporarily and run "pfctl -R /etc/pf.conf" to use lynx web browser
# *COMMENTED* pass out quick on $if inet proto udp from $imap_svr port $ephemeral to any port { www, https
} flags S keep state

pass out quick on $if inet proto udp from $imap_svr port $ephemeral to $hbedv_svrs port www flags S keep state

# Prevent logging for known, frequent network traffic that cannot be prevented from occurring that will otherwise
# fill the packet logs with mostly useless information. This is a tradeoff and there could be a malicious packet in
# the future that will not be logged because of this. In the case of the network built, Windows machines the local
network
# make frequent queries to the broadcast address for the local network.
block in quick on $if inet proto udp from $localnet port = netbios-ns to $local_broadcast port = netbios-ns
block in quick on $if inet proto udp from $localnet port = netbios-dgm to $local_broadcast port = netbios-dgm

# Reject all remaining traffic
block in log quick all
block out log quick all
```

Change /etc/nat.conf: to contain the following:

```
# Network interface
if = "dc0"

rtr on $if proto tcp from any to mail port imaps -> localhost port limaps
```

After making those edits, cause them to take effect and verify they load OK by doing the following:

Action	Remarks
pfctl -R /etc/pf.conf	Reload filter rules.
	Look for any errors reported, such as, "syntax error" and make corrections.
pfctl -s rules	Display current filter rules.
pfctl -N /etc/nat.conf	Reload translations.
	Look for any errors reported, such as, "syntax error" and make corrections.
pfctl -s nat	Display latest translations.

Edit /etc/rc.conf to modify the following line to be as follows so that packet filtering will be started upon reboot:

pf=YES # Packet filter / NAT

OpenSSH

This package provides a more secure alternative to telnet, rlogin, ftp, rcp, etc. for remote shell sessions and file copying. This already ships with OpenBSD, but a newer version must be installed to correct vulnerabilities (OpenBSD 3.1 patch #6). This installation does not use OpenSSH, but the most recent version is installed as a defense in depth move to replace the less secure older version. The other option would be to remove the program entirely.

Action	Explanation
tar xvzf openssh-3.4.tgz	Unbundlle the distribution tar file.
cd ssh	Change into the distribution directory.
make obj	
make cleandir	Clean distribution directory of any previous results of complications.
make depend	Constructs makefile dependency lists using mkdep.
make	Build the executables.
make install	Install the package within /usr/local hierarchy.
cp ssh_config sshd_config /etc/ssh	

Setting up NTP

An NTP daemon is run to synchronize the local clock with the proper time for use in mail headers and logs. Additionally time is provided to other machines within the local network to reduce the burden on public time servers.

Action	Explanation
tar xvzf ntp-4.1.1a.tar.gz	Unpack the distribution tar file.
cd ntp-4.1.1a	Change into the directory of the distribution.
setenv LIBS "-lcurses -ltermcap"	Avoids an error with running ntpq where the symbol "_tgetent" is undefined from a reference in libreadline
./configure	Tailor the makefile to the nuances of this particular system.
make	Build the executables.
make install	Install the executables and related files.
cp conf/baldwin.conf /etc/ntp.conf	Start with an existing template file for the NTP configuration file.
mkdir -p /usr/local/html/ntp	Create a directory to install the HTML based help pages for NTP.
cd html	Move to the HTML help page directory within the distribution.
cp -R * /usr/local/html/ntp	Install the HTML help pages.
mkdir /etc/ntpstats	Create a directory where NTP statistics can be stored.

If something should go wrong, the software can be uninstalled with "make uninstall" and the distribution directory can be cleaned back up to its initial state with "make distclean".

There are likely no local NTP servers already in place within the SOHO. If not, then examine the lists of public NTP servers (http://www.eecis.udel.edu/~mills/ ntp/clock1.htm) and locate three servers as close to your location as possible. Note that many NTP servers have placed limitations on who is permitted to use them that is enforced with the honor system. Access control within NTP is utilized. The local machine is permitted all activities over the loopback interface. No remote machine is permitted to use NTP facilities to query or change its configuration. The local network is permitted to query the system for time and the chosen public time servers are permitted to provide time information which is utilized in adjusting the local clock. Note that the IP addresses of the NTP servers must be entered to provide this granularity of protection. NTP does not support the use of a hostname in the NTP configuration which makes this a maintenance issue should an NTP server IP address change.

Edit the /etc/ntp.conf as follows:

Original Lines	Modify Original Lines as shown
# stratum-1 chimes instended as backup should the external clock croak. #	# stratum-1 chimes instended as backup should the external clock croak. #

server 127.127.1.0 prefer # local clock driver **broadcast 224.0.1.1 key 6 ttl 127** **peer rackety.udel.edu** **peer barnstable.udel.edu** **peer mizbeaver.udel.edu** **peer pogo.udel.edu** # # Miscellaneous stuff	**server *<ntp1.sans.org>*** **server *<ntp2 sans.org>*** **server *<ntp3.sans.org>*** # # Miscellaneous stuff
driftfile /etc/ntp.drift # path for drift file **statsdir /baldwin/ntpstats/ # directory for statistics files** filegen peerstats file peerstats type day enable	driftfile /etc/ntp.drift # path for drift file **statsdir /etc/ntp/ntpstats/ # directory for statistics files** filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable **#** **# Authentication stuff** **#** **keys /usr/local/etc/etc/ntp.keys # path for keys file** **trustedkey 3 4 5 6 14 15** **requestkey 15 # key (7) for accessing server variables** **controlkey 15 # key (6) for accessing server variables**	filegen clockstats file clockstats type day enable **#** **# Access control** **#** **# By default, don't permit anything via NTP** **restrict default ignore** # **Don't restrict the server from accessing itself** **restrict 127.0.0.1** **# Permit the local network to receive time, but not** **# query or modify the configuration.** **restrict 192.168.0.0 mask 255.255.255.0 noquery** **# Permit authorized time servers to update the time of** **# this machine, but not query or modify this NTP** **# configuration or receive time from this server.** **# Note: IPs are sanitized here** **restrict x.x.34.2 noquery noserve** **restrict x.x.28.134 noquery noserve** **restrict x.x.98.21 noquery noserve**

Verify manually that ntp will work:

Action	Explanation
ntpdate <*ntp1.sans.org*>	Manually initialize the clock to something reasonable
/usr/sbin/ntpd -d	Test the ntp daemon manually in debug mode to verify it is working properly.

Edit /etc/rc.local adding the following section:

```
if [ X"${ntpd}" == X"YES" -a -x /usr/local/sbin/ntpd \
    -a -e /etc/ntp.conf ]; then
        echo -n ' ntpd';     /usr/local/sbin/ntpd -p /var/run/ntpd.pid
fi
```

GNU Make (gmake)

Make is a capability to define an orderly relationship between source files and the various output components required to eventually build the final end products which often corresponds to binary executables. This package is required to successfully compile the Courier IMAP server with no errors. Courier IMAP will report make errors if a build is attempted with the make shipped with OpenBSD.

Action	Explanation
tar xvzf make-3.79.1.tar.gz	Unbundlle the distribution tar file.
cd make-3.79.1	Change into the distribution directory.
./configure	Tailor the makefile for this platform and its inherent capabilities.
make	Compile the binaries.
make check	Verify proper operation.
make install	Install the package within /usr/local hierarchy.
mv /usr/local/bin/make /usr/local/bin/gmake	Rename the GNU Make binary to gmake so that it can coexist with the OpenBSD make which resides in /usr/bin. The OpenBSD make can still be useful for building

	OS patches and kernels where we would rather be completely sure all is well as it normally would be.
mv /usr/local/man/man1/make.1 /usr/local/man/man1/gmake.1	Rename the corresponding man page as well. Note that we did not correct the contents of the man page which will still refer to make.

Gnu DBM (GDBM)

Gnu DBM provides a minimal database capability using hashes. This package is required to support virtual mailboxes within Courier IMAP.

Action	Explanation
tar xvzf gdbm-1.8.0.tar.gz	Unbundle the distribution tar file.
cd gdbm-1.8.0	Change into the distribution directory.
./configure	Creates a customized makefile for this system
make	Compile and link the source code
make install	Puts resultant library files within /usr/local/lib

Gnu Privacy Guard (GnuPG)

GnuPG enables the encryption and signing of data. Some software distributions are signed by GnuPG or PGP which can be verified afterwards as authentic.

Action	Explanation
md5 gnupg-1.0.7.tar.gz	Verify the MD5 hash against that shown on http://www.gnupg.org/download.html if this has not already been done.
tar xvzf gnupg-1.0.7.tar.gz	Unbundlle the distribution tar file.
cd gnugp-1.0.7	Change into the distribution directory.
./configure --disable-asm	Creates a customized makefile for this system
make	Compile and link the source code
make check	Verify proper operation of the built package. Warning, may take an extremely long time depending upon the system!
make install	Puts resultant gpg executable within /usr/local/bin
gpg --gen-key	Generate a private key to be able to sign other public keys or alternatively, you can load an existing key should one already exist elsewhere
Select (1) DSA and ElGamal	Permits both signing and encryption
Select keysize at the default of 1024	This is a tradeoff of speed versus difficulty to break the key. Increase this should you feel uncomfortable up to a maximum of 2048 bits.
Select (0) Do not expire key	
Enter your full name (i.e. Joe Smith)	Enter information that attempts to describe the person owning the private key to be generated.
Enter your email address	
Enter a comment defining the purpose of this key.	
Enter a passphrase	This is similar to a password, but should be longer. Ideally this will be a phrase you can easily remember, but not a common quote or saying.

Courier IMAP Server

The Courier IMAP server is a component of the Courier mail system which includes the MTA, web mail, and POP server components. This is available separately from those other Courier components as well as embedded in the MTA package.

The Courier system is a good fit due to its wide range of features with security factored into the design and a modular layout with the ability to pick and choose components of interest - both within the IMAP server and also the web mail, POP server, and MTA components. This will permit our installation to grow into this solution and potentially evolve into a larger and more sophisticated solution as the needs of users change. In our case, none of these components will be utilized except for the IMAP server.

Because of the small nature of the system being constructed, the recommendation is to not implement mail quotas and leave it to the honor system. Quotas can always be instituted in the future should the need present itself. Similarly we will assume that such a small installation does not have

or necessarily need an LDAP or SQL database pre-existing to serve as an authentication database. Therefore these capabilities will not be compiled into the resultant product. The capability to send outgoing mail via IMAP will not be enabled to keep the installation as simple and focused as possible to the core capability and purpose. Users will configure their MUA products to send via the ISP's SMTP server.

The documentation indicates that Courier will not currently compile with gcc 3.x, but OpenBSD 3.1 comes wth gcc 2.95.3, so this is not a problem unless the standard gcc is not used.

Maildirs that originate from the qmail package will be used in preference to the more traditional mailbox structures. Maildirs store each message in a separate file which helps to eliminate the need for file locking and reduces the odds of corruption of all mail with a single error.

Modify /etc/login.conf as follows to increase the number of processes that can be created (i.e. forked) by a given user. Without this change the check phase will fail on a test of waitlib within the testwait program with an error "fork: Resource temporarily unavailable".

Original Lines	Modify Original Lines as shown
default:\ :path=/usr/bin /bin/ /usr/sbin /sbin **/usr/X11R6/bin** /usr/local/bin:\ :umask=022:\ :datasize-max=256M:\ :datasize-cur=64M:\ :maxproc-max=**128**:\ :maxproc-cur=**64**:\	default:\ :path=/usr/bin /bin/ /usr/sbin /sbin /usr/local/bin **/usr/local/sbin**:\ :umask=**027**:\ :datasize-max=256M:\ datasize-cur=64M:\ :maxproc-max=**200**:\ :maxproc-cur=**200**:\

Create a special group and user to run mail processes and then compile the package as this special user. The package is ultimately built twice. Once to check that the platform is working properly and then a second to build the final version to be used in production use. Use of some options cause the check phase to fail which is the reason for two builds.

Action	Remarks
groupadd mail	Create a special group for mail operations.
useradd -b / -c "Mail User" -g mail mail	Create normal user account needed to build Courier. This will also be used for all mail operations once the system is put into production.
chown mail:mail /mail	Change the top level mail directory to be owned by the special mail user.
chmod 750 /mail	Limit access to this directory to mail related users.
su -l mail	Very critical according to the instructions with the package that this be a non-root user!
tar xvzf courier-imap-1.5.3.tar.gz	Unbundle the distribution
cd courier-imap-1.5.3	Change to the package distribution directory.
setenv CPPFLAGS "-I/usr/local/include"	Required for --with-db=gdbm to work
setenv LDFLAGS "-L/usr/local/lib"	Required for --with-db=gdbm to work
./configure --prefix=/usr/local --without-authdaemon --without-authcram --without-authcustom --without-authpwd --with-db=gdbm --without-ipv6	Tailor the makefile to this particular system and its inherent capabilities while specifying our own preferences for this installation.
gmake	Compile and link the source code into binary executables. **The use of gmake is critical!** Strange errors will be received by the standard OpenBSD make.
gmake check	We have to do this before we modify the source code ourselves or it will fail the check due to our modifications
gmake distclean	Cleanup the just built objects and executables since we need to modify some files before proceeding.

One method used by persons seeking unauthorized access to equipment is reconnaissance. Fingerprinting can be used by an attacker to determine what type of equipment and/or software comprises the target system. This helps to make the attack more succinct and likely for undetected success. Once the attacker knows what comprises the defenses, they choose among any known vulnerabilities or even setup their own test system to discover new vulnerabilities to then attempt against the target system should the attacker be extremely interested in this particular system. In the case of applications, the fingerprint information currently sought is fairly blatant and freely available in most applications of today by boldly announcing the name of the product as well as the particular version in many cases.

Illustrative commands, no need to perform these.	Results

telnet 127.0.0.1 143	Trying 127.0.0.1... Connected to 127.0.0.1. Escape character is '^]'. * OK **Courier**-IMAP ready. **Copyright 1998-2001 Double Precision, Inc.** See COPYING for distribution information
CAPABILITY	* CAPABILITY IMAP4rev1 CHILDREN NAMESPACE THREAD=ORDEREDSUBJECT THREAD=REFERENCES SORT QUOTA STARTTLS CAPABILITY OK CAPABILITY completed
LOGOUT	* BYE **Courier**-IMAP server shutting down LOGOUT OK LOGOUT completed

To make fingerprinting of our IMAP server a bit more difficult, we will attempt to obscure its true name and origin for those accessing the system externally. Since the system will be setup so only users within the local network can access IMAP, it is arguable that this is beyond the call of duty. Nonetheless, being paranoid we recommend a little more effort just in case. When we look within the distribution for lines with "Courier" in them with a command such as "find . -exec grep -l Courier -print", we find the following relevant change helpful:

Source Filename	Original Lines	Modify Original Lines as shown
imap/imaplogin.c	if (nexttoken()->tokentype != IT_EOL) return (-1); writes("* BYE **Courier-IMAP** server shutting down\r\n");	if (nexttoken()->tokentype != IT_EOL) return (-1); writes("* BYE **IMAP** server shutting down\r\n");

imap/imaplogin.c	if (authmoduser(argc, argv, 60, 5))
	{
	writes("* OK **Courier-IMAP** ready. **Copyright 1998-2001 Double Precision, Inc. See COPYING for distribution information**.\r\n");
		if (authmoduser(argc, argv, 60, 5))
		{
		writes("* OK **IMAP** ready.\r\n");

imap/imapd.c	fetch_free_cache(); writes("* BYE **Courier-IMAP** server shutting down\r\n");	fetch_free_cache(); writes("* BYE **IMAP** server shutting down\r\n");
imap/mainloop.c	static RETSIGTYPE sigexit(int n) { static char byemsg[]="* BYE **Courier-IMAP** server shut down by signal\r\n";	static RETSIGTYPE sigexit(int n) { static char byemsg[]="* BYE **IMAP** server shut down by signal\r\n";

Actually build the software now that our modifications have been made. These steps should be begun while still the special "mail" user:

Action	Remarks
cd /mail/courier-imap-1.5.3	If you are not already here from earlier
setenv CPPFLAGS "-I/usr/local/include"	Required for --with-db=gdbm to work
setenv LDFLAGS "-L/usr/local/lib"	Required for --with-db=gdbm to work
./configure --prefix=/usr/local --without-authdaemon --without-authcram --without-authcustom --without-authpwd --without-ipv6 --with-db=gdbm	Tailor the makefile to this particular system and its inherent capabilities while specifying our own preferences for this installation. May need to add the --enable-workarounds-for-imap-client-bugs flag here if using an IMAP client that is not implemented properly.
gmake	Compile and link the source code into binary executables. **The use of gmake is critical!** Strange errors will be received by the standard OpenBSD make.
exit	Go back to being the root user before installing the software.
cd /mail/courier-imap-1.5.3	Change to the package distribution directory.
gmake install-strip	Install the binaries in stripped form.
gmake install-configure	Install configuration files.
chown mail:mail /usr/local/etc/imapd*	Fix the IMAP configuration files to be owned by the special mail user.
./authlib/authinfo	Verify that AUTHENTICATION_MODULES only has authuserdb.

Should a problem be experienced and there be a need to start over, then "gmake uninstall" as root will remove any files installed previously and "gmake distclean" will remove any object, executable, and other files created within the source distribution directories as part of the original compilation and linking process.

As root, create a script to simplify creating a mail account for a new user. This script still could use some error checking, but it gets the job done if you give it proepr input. The maildirmake command is used to create a new maildir for the new user. Create the file /usr/local/bin/museradd containing:

```
#!/bin/sh
# Script: museradd
# Usage: museradd <imapusern>

MUSER=mail
MGRP=mail
MHOME=/mail
MUID=`id -u $MUSER`
MGID=`id -g $MUSER`

su -l $MUSER -c "maildirmake $MHOME/$1"
userdb $1 set home=$MHOME mail=$MHOME/$1 uid=$MUID gid=$MGID
chmod 700 /etc/userdb
userdbpw -md5 | userdb $1 set imappw
makeuserdb
chown $MUSER:$MGRP /etc/userdb*
```

Action	Remarks
chmod 500 /usr/local/bin/museradd	Make the custom script executable by the root user.
rehash	Cause shell to rebuild internal structures to find the new script.
museradd <*imapuser1*>	Use the custom script to add a new IMAP account for each user. Note that this user will not have a corresponding OS user account (virtual mailbox). A corresponding password will need to be provided when requested.

The IMAP server will be run by the special "mail" user, so it will not have any special permissions or abilities. Some adjustments will be necessary from the standard installation to account for this. The IMAP server will be setup to listen on a non-privileged port (11993) instead of the standard port for SSL enabled IMAP (993). Special directories will be used to hold a file storing the current process ID of the Courier process and another directory to store a cache file.

Action	Remarks
cd /usr/local	
mkdir imaprun	Create a dedicated directory for storage of the current Courier process ID.
chown mail:mail imaprun	Make the directory owned by the mail user.
chmod 700 imaprun	Restrict access to the mail user.
mkdir mail	Create a dedicated directory for storage of a Courier cache file.
chown mail:mail mail	Make the directory owned by the mail user.
chmod 700 mail	Restrict access to the mail user.

Edit /usr/local/etc/imapd-ssl as follows:

Original Lines	Modify Original Lines as shown
# The SSLADDRESS setting is a default for ports that do not have # a specified IP address. SSLPORT=993	# The SSLADDRESS setting is a default for ports that do not have # a specified IP address. SSLPORT=**11**993
# That's the SSL IMAP port we'll listen on. # Feel free to redefine MAXDAEMONS, TCPDOPTS, and MAXPERIP. SSLPIDFILE=**/var/run**/imapd-ssl.pid	# That's the SSL IMAP port we'll listen on. # Feel free to redefine MAXDAEMONS, TCPDOPTS, and MAXPERIP. SSLPIDFILE=**/usr/local/imaprun**/imapd-ssl.pid
# problems with SSL clients. Disable SSL caching by commenting out the # following settings: TLS_CACHEFILE=/usr/local/**var**/couriersslcache	# problems with SSL clients. Disable SSL caching by commenting out the # following settings: TLS_CACHEFILE=/usr/local/**mail**/couriersslcache

A server certificate will be needed to be able to make Courier use IMAPS. A self-signed certificate will be used in the following steps. A more

officially sanctioned certificate can be purchased from a commercial Certificate Authority such as VeriSign instead. Use of a self-signed certificate within such a constrained environment is not considered to be a major issue.

Edit /usr/local/etc/imapd.cnf before generating a certificate:

Original Lines	Modify Original Lines as shown
ST=**NY**	ST=***<YourState>***
L=**New York**	L=***<YourCity>***
O=**Courier Mail Server**	O=***<YourOrganizationName>***
CN=**localhost**	CN=**mail**
emailAddress=**postmaster@example.com**	emailAddress=***<YourAdmin@sans.org>***

Action	Remarks
cd /usr/local/etc	
mkimapdcert	This creates an IMAP server certificate file /usr/local/share/imapd.pem using the inputs provided in /usr/local/etc/imapd.cnf..
chown mail:mail /usr/local/share/imapd.pem	Change the IMAP server certificate to be owned by the special mail user.
openssl x509 -in /usr/local/share/imapd.pem -noout -text | more	Verify the contents of the resultant certificate look appropriate by dumping its contents.

Edit /etc/rc.local adding the following section:

```
if [ X"${imap}" == X"YES" -a -x /usr/local/libexec/imapd-ssl.rc ]; then
        echo -n ' imapd';    su -l mail -c '/usr/local/libexec/imapd-ssl.rc start'
fi
```

Edit /etc/rc.conf adding the following section:

```
imapd=YES                       # run imapd if it exists
```

Procmail

This package is in its basic form an MDA, but has a large set of capabilities to include mailing lists, delivering mail into different folders, and running programs once mail arrives. This system will depend upon procmail to simply deliver mail provided within a named local maildir.

Action	Remarks
tar xvzf procmail-3.22.tar.gz	Unpack the tar bundle for the package.
cd procmail-3.22	Change into the product distribution directory.

Edit the Makefile in the distribution directory to force the man pages to go into the more standard /usr/share/man within OpenBSD instead of /usr/man. Also avoid compiling and installing other unused binaries just to keep things minimalized.

Original Lines	Modify Original Lines as shown
MANDIR = $(BASENAME)/man	MANDIR = $(BASENAME)**/share**/man
BINSS = procmail **lockfile formail mailstat**	BINSS = procmail

Now actually build and install the procmail package.

Action	Remarks
make install	Build and install the binaries and related files.

The administrator can undo and start over in the case of a problem with "make deinstall".

For each end user (<*imapusern*> in this case):
Create a procmailrc file in /mail named <*imapusern*>_procmail.rc owned by the special "mail" user (i.e chown mail /mail/<*imapusern*>_procmail.rc) containing:

```
SHELL=/bin/sh
MAILDIR=/mail
ORGMAIL=$MAILDIR/<imapusern>/
DEFAULT=$ORGMAIL
```

Fetchmail

This package can retrieve mail over the POP protocol for one or more accounts from one or more ISPs and even forward it via SMTP (which this installation will not do).

Action	Remarks
tar xvzf fetchmail-5.9.0.tar.gz	Unpack the distribution tar bundle.
cd fetchmail-5.9.0	Change into the distribution directory to build the package.
./configure --with-ssl --enable-fallback=procmail --disable-ETRN --disable-IMAP	Have the package attempt to tailor itself to this particular platform and its inherent capabilities. Can add the "--ssl" option if one of the ISPs providing POP service supports SSL enabled POP.
make	Compile the fetchmail source code.
make install	Install the built binaries and related files.

Create the file .fetchmailrc in the home directory of the special "mail" user (/mail) owned by the special "mail" user (i.e chown mail /mail/.fetchmailrc):

```
set syslog  # Send any messages to syslog
set invisible  # Suppress fetchmail from adding any new Received header line
defaults protocol pop3  # Retrieve messages via the Post Office Protocol version 3
          keep  # for testing - Do not remove any messages from the POP server after retrieval
          fetchall # for testing - Retrieve all messages including old ones
poll  <pop1.sans.org>  username <popuser1> password  "<realpass1>"
      mda "/usr/bin/procmail  /mail/<imapuser1>_procmail.rc"
poll  <pop2.sans.org>  username <popuser2> password  "<realpass2>"
      mda "/usr/bin/procmail  /mail/<imapuser2>_procmail.rc"
```

Add an entry in cron to run fetchmail every 5 minutes to retrieve any new mail for users. Adjust the polling interval as needed. Use the command "crontab -e" to add the entry to the cron.:

```
*/5   *   *   *   *   su -l mail -c /usr/local/bin/fetchmail
```

H+BEDV AntiVir

It is difficult to find a free anti-virus solution for any platform, but this is particularly true for UNIX. This particular product is available free for private non-commercial use on some platforms, but a license can be purchased if the SOHO is a profit-based venture. The linux and freebsd versions are shown as being available free under the previously stated constraints, but the OpenBSD version is not explicitly mentioned in this vein. A free license must be requested by registering at http://www.hbedv.com/private/. Note that a valid license is required to get the proper return codes returned from the antivir command to distinguish a file containing a virus from one without. A return code of 214 indicates that a license file was not found. Other anti-virus solutions are also available and could likely be easily accomodated should it have a command line interface and some sort of return code or feedback available indicating if a virus was detected.

AntiVir provides:

- antivir - command line virus scanner
- avguard - intercepts in the background any disk accesses and scans the files to be accessed for viruses (requires Dazuko from http://www.dazuko.org which currently only runs on Linux)
- avupdater - simple program which runs as a daemon and periodically retrieves any virus updates

This installation will only be utilizing the antivir program.

Action	Prompt	Explanation
Purchase or request a non-commercial license at http://www.hbedv.com/private/		Get a key file which is necessary to use this product in an legally automated fashion.
tar xvzf avobsrv.tgz		Unpack the contents of the archive
cd antivir-2.0.4-server		Move into the distribution directory
cp <*key dir*>/hbedv.key .		Copy the key file from where you have it into the distribution directory
./install		Begin installing

y	Would you like to create a link in /usr/bin? [y]	Avoids need to update path settings.
n	Would you like to install the automatic internet updater? [n]	Cron will be used instead to periodically poll and retrieve any available updates.
y	Would you like to configure AntiVir now? [y]	Helps minimize any manual editing of /etc/antivir.conf.
y	Would you like email notification of viruses? [n]	
email_of_admin@sans.org	What email address will receive notifications? []	
n	Would you like AntiVir to log to a custom file? [n]	
n	Does this machine use an HTTP proxy server? [n]	
mail	Which syslog FACILITY should AntiVir use? [user]	
notice	Which syslog PRIORITY should AntiVir use? [notice]	
y	Save configuration settings? [y]	Verify the settings shown to be proper and accept if valid.
rehash	#	Cause shell to rebuild internal structures to find new executables for AntiVir.
cp <*key dir*>/hbedv.key /usr/lib/AntiVir	#	If you did not have a key file earlier, then you can copy the key file from where you have it directly into the program directory.
Modify /etc/antivir.conf, if needed, to customize.		
antivir --help	#	Shows full detailed description of options.
/usr/lib/AntiVir/antivir --update	#	Manually update the program and virus signatures, if needed. Verify the update works properly with no error messages. Vendor says that full path is absolutely needed and that it must be run as root.

Add an entry in cron to nightly check for new updates to virus signatures or the anti-virus program itself. Adjust the time as needed. This example has it run at 3:17am each morning. Consider entering in the script within the README file of the distribution so that feedback can be logged as to the success of the attempt. Use the command "crontab -e" to add the entry to the cron.:

```
17   3  *   *   *    /usr/lib/AntiVir/antivir --update -q
```

Here are the main antivir options this installation is utilizing with a corresponding description:

Antivir Option	Option Description
-z	uncompress files to look within archives
-e	repair infected files
--update	get a newer version of AntiVir or the signatures
-q	quiet mode, do not print out any messages

Protector

By default, the protector software severely limits which mail attachments are permitted and closely scrutinizes those which potentially are allowed with custom analysis programs. This is achieved by substituting the use of protector where procmail would traditionally suffice. The new program is a script called protector located within /usr/bin which first calls a protector binary executable stored in /usr/lib/protector/bin and once complete the script then calls procmail with the options originally supplied to the script. This package is used with modifications to utilize our anti-virus software to additionally scan mail attachments to provide a second layer of protection. Active development continues with this package and more capabilities and features are likely to appear soon making its use even more viable for such an installation.

Action	Explanation
tar xvzf protector-1.00.6.tgz	Unbundle the distribution file.
cd protector-1.00.6	Enter the directory of the unbundled distribution.

There is no reason for us to run the save_reject program as set-UID root or set-GID for this installation. This program is called from within the part_filter script which is in turn run from the main protector script. We go ahead and tailor the Makefile so that the various elements are accessible only by the special mail user.

The smrsh program is a restricted shell utility for sendmail that ships with some Linux distributions as a replacement for /bin/sh in the program mailer definition. On Linux, it limits sendmail to only be able to run the programs listed within the /etc/smrsh directory. Instead of /etc/smrsh, OpenBSD uses the directory /usr/libexec/sm.bin by default. Since we do not need this capability as we are not using sendmail to deliver incoming email and are already editing the Makefile, we will eliminate the command and avoid the error generated otherwise during installation. If this is not corrected, then the following error will be displayed:

```
ln -s /usr/bin/protector /etc/smrsh/protector
ln: /etc/smrsh/protector: No such file or directory
gmake: *** [install] Error 1
```

Edit the Makefile within the distribution directory as follows:

Original Lines	Modify Original Lines as shown
ROOT_GROUP = **root**	**PR**OT_USER = **mail** **PROT_GROUP = mail**
chown **root** $(LIB_DIR)/bin/save_reject chgrp $(**RO**OT_GROUP) $(LIB_DIR)/bin/save_reject chmod **6755** $(LIB_DIR)/bin/save_reject	chown **$(PROT_USER)** $(LIB_DIR)/bin/save_reject chgrp $(**PR**OT_GROUP) $(LIB_DIR)/bin/save_reject chmod **500** $(LIB_DIR)/bin/save_reject
chown **root** $(REJ_DIR) chgrp $(**RO**OT_GROUP) $(REJ_DIR)	chown **$(PROT_USER)** $(REJ_DIR) chgrp $(**PR**OT_GROUP) $(REJ_DIR)
chmod 700 $(REJ_DIR) **ln -s /usr/bin/protector /etc/smrsh/protector** install -o root -g mail -m 07**55** daily.sh /etc/cron.daily/protector	chmod 700 $(REJ_DIR) install -o **$(PROT_USER)** -g **$(PROT_GROUP)** -m 07**00** daily.sh /etc/cron.daily/protector
save_reject: save_reject.o ($CC) $(CFLAGS) save_reject.o -o $@ chmod **6755** $@	save_reject: save_reject.o ($CC) $(CFLAGS) save_reject.o -o $@ chmod **500** $@

Modifications are provided below which cause all attachments to undergo an added step to check for viruses. The output from the virus scanner are substituted for any attachment which is identified as containing a virus. Edit part_filter within the distribution directory as follows:

Original Lines	Modify Original Lines as shown
#! /bin/sh	#! /bin/sh **# Altered version: mods by Don Pitts Sept. 9, 2002** **# Mods must be clearly disclosed to others to abide by the license**
ZIPPED=$TMP/zipped	ZIPPED=$TMP/zipped **# *** BEGIN MOD Sept. 9, 2002** **# Store output from virus checking to be substituted for any** **# attachments found to contain a virus.** **VIRUS_OUTPUT=$TMP/virus_scan_output** **# H+BEDV virus scanner**

	VSCAN=/usr/lib/AntiVir/antivir **# -z = uncompress archives** **# -e = repair infected files, when possible** **VSCAN_OPTS="-z -e"** **# *** END MOD Sept. 9, 2002**
# Pre-allow basic text types - these are normally moderately safe, ## although I guess someone will prove me wrong in this one day. case "$CONTENT_TYPE" in message/disposition-notification \|\ message/delivery-status \|\ text/* } echo "$HEADERS"; cat; return 0;; esac	# Pre-allow basic text types - these are normally moderately safe, ## although I guess someone will prove me wrong in this one day. case "$CONTENT_TYPE" in message/disposition-notification \|\ message/delivery-status \|\ text/* } **# *** BEGIN MOD Sept. 9, 2002** **cat >$ORIGINAL** **cp $ORIGINAL $DECODED** **virus_check** echo "$HEADERS"; cat **$ORIGMAIL**; return 0;; **# *** END MOD Sept. 9, 2002** esac
## Decode the attachment and get it's file type by inspecting it's ## "magic" numbers - etc. cat >$ORIGINAL decode <$ORIGINAL >$DECODED CONTENT_TYPE="`classify $DECODED`"	## Decode the attachment and get it's file type by inspecting it's ## "magic" numbers - etc. cat >$ORIGINAL decode <$ORIGINAL >$DECODED **# *** BEGIN MOD Sept. 9, 2002** **virus_check** **# *** END MOD Sept. 9, 2002** CONTENT_TYPE="`classify $DECODED`"
esac exit 0 } ## # Dangerous extension checking - a recent trick is to give files in MS mail # attachments two extensions, the last being hidden by the mail client, but	esac exit 0 } **# *** BEGIN MOD Sept. 9, 2002** **##** **# virus checking - Invoke the H+BEDV command line virus** **# scanner to see if the attachment has a known virus.** **# Should a virus be found, the textual output from the virus** **# scanner will be substituted for the original attachment.** **virus_check()** **{** **if [-x $VSCAN]; then** **$VSCAN $VSCAN_OPTS $DECODED >$VIRUS_OUTPUT; stat=$?** **[$stat -eq 1] && replace_with $VIRUS_OUTPUT** **fi** **}** **# *** END MOD Sept. 9, 2002** ## # Dangerous extension checking - a recent trick is to give files in MS mail # attachments two extensions, the last being hidden by the

	mail client, but
if [-f $LIB_DIR/messages/$1.txt]; then cat $LIB_DIR/messages/$1.txt else echo "WARNING CODE: $1" cat $LIB_DIR/messages/general.txt fi	if [-f $LIB_DIR/messages/$1.txt]; then cat $LIB_DIR/messages/$1.txt **# *** BEGIN MOD Sept. 9, 2002** **elif [-f $1]; then** **cat $1** **# *** END MOD Sept. 9, 2002** else echo "WARNING CODE: $1" cat $LIB_DIR/messages/general.txt fi

You can make other edits to the part_filter script to let other file types through as described at http://www.lowth.com/protector/bin/view/Protector/PartFilterEdits100 even though the author suggests that it is dangerous.

Action	Explanation
gmake build	Compile the executables.
gmake install	Install the executables, scripts, and associated data files within /usr/lib/protector and /var/protector/rejects and /usr/bin/protector with links in /usr/bin and /usr/lib.

In case of an error, the installed program may be removed with "gmake uninstall" and the distribution directory can be cleaned up to remove the built object and executables with "gmake clean".

Bison

This package is used to generate a program which parses a defined language and is required by the AIDE file integrity tool..

Action	Explanation
tar xvzf bison-1.35.tar.gz	Unbundle the distribution file.
cd bison-1.35	Enter the directory of the unbundled distribution.
./configure	Tailor the package to this machine.
make	Build the executables.
make check	Verify proper operation of built binaries.
make install	Install the binaries and associated data files.
rehash	Trigger the shell to reread all executable files within directories listed in the path.

Libmhash

This package provides a uniform method of access to many different hash algorithms which is required by the AIDE file integrity tool.

Action	Explanation
tar xvzf mhash-0.8.16.tar.gz	Unbundle the distribution file.
cd mhash-0.8.16	Enter the directory of the unbundled distribution.
./configure	Tailor the package to this machine.
make	Build the executables.
make check	Verify proper operation of built binaries.
make install	Install the binaries and associated data files.
rehash	Trigger the shell to reread all executable files within directories listed in the path.

AIDE

AIDE is a replacement package for Tripwire that provides a method to verify file integrity. Bison and libmhash is required for AIDE to compile. This package has some interesting features that must be dealt with to get a working version on OpenBSD. In hind sight, perhaps the OpenBSD version which only has 0.7 should be considered as an option. The most current version available of AIDE in general is 0.9. We will valiantly

struggle on here because we are too far to go back at this point. A warning to others that it might not be worth it, although we have hopefully already tackled the biggest issues.

There are some missing semicolons at the end of rules within the src/conf_yacc.y for which warnings (i.e. conf_yacc.y:185: warning: previous rule lacks an ending ';') are generated by bison. Since these are warnings and they are a syntax issue where the parser understands it is missing, they can be safely ignored.

While compiling src/conf_yacc.c, the compiler finds an error in the included include/db_config.h on line 357 indicating "syntax error before `blkcnt_t'. In tracking down the origin, there is a reference to "blkcnt_t" within the configure script. This is supposed to be the type of the field containing the number of blocks which is an attribute returned from a stat() system call. This type is not defined within the OpenBSD OS like it is in some others. Looking at /usr/include/sys/stat.h shows that it should correspond to int64_t instead according to the st_blocks field within the stat struct. Correcting this in the configure script will resolve the problem.

There are some definitions within src/util.c that attempt to ensure the aide binary is statically linked. These are just empty dl functions which are supposed to trick the linker into not linking into the real functions at link time. With OpenBSD these extra functions only end up generating an error of multiple symbols
Removing the definitions of dlopen, dlsym, dlclose, and dlerror within src/util.c as shown below ends up resolving this problem. It is common to wish to have particularly sensitive security programs statically linked. The issue is that a dynamically linked executable will run executable instructions pulled at run-time from shared library files. There are too many ways in which an attacker might subvert either the library files themselves or the process of finding the library files and thus inject unintended code of a potentially malicious nature to risk this. For normal programs there are many advantages to dymanic executables such as sharing common code and simplifying bug fixes and upgrades, but even this is not without its own pain similar to DLL files within the Windows environment.

Begin the adventure by getting into the distribution:

Action	Explanation
tar xvzf aide-0.9.tar.gz	Unbundle the distribution file.
cd aide-0.9	Enter the directory of the unbundled distribution.

Make the corrections discussed on the configure and src/util.c files before attempting compilation:

Filename	Original Lines	Modify Original Lines as shown
configure	cat >>confdefs.h <<EOF #define AIDE_BLKCNT_TYPE **blkcnt**64_t EOF	cat >>confdefs.h <<EOF #define AIDE_BLKCNT_TYPE **int**64_t EOF
configure	cat >>confdefs.h <<EOF #define AIDE_BLKCNT_TYPE **blkcnt**_t EOF	cat >>confdefs.h <<EOF #define AIDE_BLKCNT_TYPE **int32**_t EOF

src/util.c	/* If we get this far no match was found so we return NULL */
	return NULL;
	}
	
	**/* We need these dummy stubs to fool the linker into believing that
	we do not need them at link time */**
	
	void* dlopen(char*filename,int flag)
	{
	return NULL;
	}
	
	void* dlsym(void*handle,char*symbol)
	{
	return NULL;
	}
	
	void* dlclose(void*handle)
	{
	return NULL;
	}
	
	const char* dlerror(void)
	{
	return NULL;
	}
		/* If we get this far no match was found so we return NULL */
		return NULL;
		}
		
		const char* aide_key_2=CONFHMACKEY_02;
		const char* db_key_2=DBHMACKEY_02;

	const char* aide_key_2=CONFHMACKEY_02; const char* db_key_2=DBHMACKEY_02;	

Complete the process of creating the package:

Action	Explanation
setenv LDFLAGS "-L/usr/local/lib"	Make sure that the libmhash library can be located during linking
./configure	Tailor the compilation to this machine.
make	Compile the binaries.
ldd aide	Verify that the binary is indeed statically linked (not dynamic). Should see "ldd: aide: not a dynamic executable" as output. The ldd program shows any dynamic references within an executable.
make install	Install the binaries and related files onto the system.
rehash	Cause the shell to find any new binaries within the directories within the current path.

Create the configuration file (/usr/local/etc/aide.conf). This is based heavily upon the default configuration file shown on the AIDE manual available at http://www.cs.tut.fi/~rammer/aide/manual.html:

```
#AIDE conf

  # Here are all the things we can check - these are the default rules
  #
  #p:      permissions
  #i:      inode
  #n:      number of links
  #u:      user
  #g:      group
  #s:      size
  #b:      block count
  #m:      mtime
  #a:      atime
  #c:      ctime
  #S:      check for growing size
  #md5:    md5 checksum
  #sha1:   sha1 checksum
  #rmd160: rmd160 checksum
  #tiger:  tiger checksum
  #R:      p+i+n+u+g+s+m+c+md5
  #L:      p+i+n+u+g
  #E:      Empty group
  #>:      Growing logfile p+u+g+i+n+S

  # Define a rule to which checks most attributes except for the modification time
  #
  Most = p+i+n+u+g+s+b+m+c+md5+sha1

  # Next decide what directories/files you want in the database

  /etc p+i+u+g     #check only permissions, inode, user and group for etc
  /etc/ntp.drift p+u+g
  /bin Most      # apply the custom rule to the files in bin
  /sbin Most     # apply the same custom rule to the files in sbin
  /var Most
  /var/cron/log >
  !/var/log/.*     # ignore the log dir it changes too often
  !/var/spool/.*   # ignore spool dirs as they change too often
  !/var/protector/rejects  # ignore rejected mail attachments
  !/var/run   # ignore process IDs
```

There is still one remaining issue and it is evidently affecting others as well which you can read about at then end of http://www.mail-archive.com/aide@cs.tut.fi/msg00127.html. Basically, it looks like there is an error in the parsing of the integrity database. If you manually edit the database following its generation and remove the last line which will contain "@@end_db" then the error message will not be displayed when the

filesystem is checked against the database. Further refinement of the rules will be necessary as the system is utilized.

```
# aide --check
Not implemented in db_readline_file 310
"@@end_db"
```

Action	Explanation
aide --install	Creates an initial database from the current state of the filesystem into /usr/local/etc/aide.db.new
cp /usr/local/etc/aide.db.new /usr/local/etc/aide.db	Accept this as the current database.
aide --check	Check the current state of the filesystem against the database in /usr/local/etc/aide.db. A different database can be consulted by specifying the --config=<*configfile*> option.

The database (/usr/local/etc/aide.db) and the configuration file (/usr/local/etc/aide.conf) should be stored on removable media such as a floppy or a CD-R and then loaded back before checking for modified files. This is to prevent someone from modifying these files to avoid detection. The command "aide --update" can be used to check the database against the filesystem like the --check option does as well as generate a new database, but in either case, the "cp /usr/local/etc/aide.db.new /usr/local/etc/aide.db" must be performed afterwards and then migrate it to media as mentioned earlier.

MMencode (MetaMail)

The Protector package requires an mmencode binary to encode and decode MIME encoded attachments between their true binary form and an specially encoded ASCII format that can be safely transmitted via email and later reconstituted into the original binary form. This is commonly shipped within Linux systems, but is not present within OpenBSD and must thus be manually added. It is found within the MetaMail package which has not been updated in many years. Because this program is part of a larger package which is not of interest to this installation, the procedure only builds the one executable needed.

Build mmencode:

Action	Explanation
tar xvzf mm2.7.tar.Z	Unpack the distribution
cd mm2.7/metamail	Notice that this is skipping further into the distribution rather than just to the top level.
make mmencode	Compile the mmencode binary.
cp mmencode /usr/local/bin	Install the one binary that is relevant onto the system.
cd ../man	Enter the man directory within the distribution.
cp mmencode.1 /usr/local/man/man1/mmencode.1	Install the corresponding man page onto the system.
rehash	Cause the shell to find any new binaries within the directories within the current path.

Locking down the BIOS

Restart the system just built (sync; reboot) and press the proper key code to enter the BIOS (sometimes the Delete key). Set the system to require a password at least each time the system enters setup. Set a corresponding password. Also set the system to only boot from the hard drive. The exact steps vary significantly due to different BIOS manufacturers and versions. See the documentation for your particular BIOS to resolve any issues.

Ongoing Maintenance

A good administrator needs to keep up with any developing issues and subscribing to the announcements mailing lists for the various products being used is a good start in making sure an awareness is kept concerning vulnerabilities and available upgrades.

List/Subject	How to Subscribe
Procmail-Announce	http://www.procmail.org/era/lists.html
H+BEDV Newsletter	http://www.hbedv.com/infos/newsletter.htm
Fetchmail-announce	http://lists.ccil.org/mailman/listinfo/fetchmail-announce
OpenBSD security-announce	http://www.openbsd.org/mail.html
OpenSSH	http://www.openssh.com/list.html

gnupg-announce	http://lists.gnupg.org/

Backups

Tape, CD-R, backup hard drive, and storage on an alternate system are all possibilities for storing copies of the system configuration. A complete backup should be made once the system has been setup properly. Incremental backups should be made when new users are added and periodically to cover newly added messages. Alternate boot disk can be used to maintain a live copy which can be updated each night. CD-R discs can be written with mkisofs which is inherently available on OpenBSD and cdrecord which can be retrieved as indicated in the references section. A command such as "mkisofs -l -L -T -v -V "<*Volume Label goes here*>" -A "<*Further notes about CD contents*>" -o cd.iso -x cd.iso /usr/cd_raw" could be used to create an ISO image at /usr/cd.iso from the files within the directory hierarchy within the /usr/cd_raw directory.

Updates

It is usually a good idea to keep products fairly up-to-date with respect to current versions. The benefits can be the correction of minor issues that may be of relevance to security that did not warrant a special patch or rate the admission of a vunerability, better support for the product from colleagues and developers in the community, and the most obvious is the addition of potentially useful features. The likeliehood of gaining these benefits depend somewhat upon selecting well engineered and maintained products to begin with. Each product upgrade should be investigated and its success within the community monitored. If there are compelling features/fixes or its stability has been demonstrated then it should be installed as an alternate installation from that used for production to work out any new issues and ensure its appropriateness to this particular environment. Ideally any installation for testing will be performed on a dedicated machine where it is less likely to affect any production capabilities. Should budget or available equipment not allow this, often packages can be installed in alternative directories and setup to use other network ports that do not conflict with production software. In either case any potential detrimental effects to security should be considered in an analysis of the impact of adopting an update to include verification of the continuance of appropriate levels of security. The timeliness of backups should be verified prior to transitioning production to the updated version.

Patches

Similar to updates, but often more focused in terms of potentially affected functionality, patches should typically only be applied when they affect security or a critical feature to this installation. Investigation into the issues resolved by the patch and its effectiveness at other installations should be performed. If the installation is currently experiencing a serious problem that is purported to be solved by the patch or it resolves a serious security issue for which there is no other alternative, then it may be the best course of action to immediately install the patch on the production system. Just as with updates, it is still important to verify backups are current and available before applying patches and ensure that security is not compromised with its adoption. In the case of a less immediate need, the application of patches should be handled similarly to updates discussed previously by thoroughly testing and verifying a patched environment prior to migrating to the fixes in production.

Periodic scans

This machine as well as the entire enterprise should be verified to still have the necessary security measures are present and effective. Nessus is an excellent network vulnerability assessment tool that should be used to ensure that the defenses are hardened from the network perspective. A process listing after reaching steady state from a clean boot should be verified against the known processes required for proper operation of the system. A listing of open network ports that are listened on should be checked similarly by using the "netstat -an" as a minimum or even better "lsof".

Logging and monitoring

A package such as LogSentry (formerly logcheck) can effectively provide a mechanism to keep aprised of any developing issues. Log messages of immediate interest can be distributed by email to the relevant administrators. Delivery to an alternative email capability is preferable since the integrity of the logs local to a potentially compromised machine could themselves be called into question. Establishment of a centralized server dedicated to hosting logs of the enterprise is advisable once the scope of the installation permits. The log files within /var/log should be manually checked periodically for evidence of any unexpected and unreported issues the system may be experiencing.

Changes

Web traffic is restricted to only outgoing requests to H+BEDV servers for the purpose of retrieving program and virus signature updates. Because no DNS name could be found to represent the server responding to update requests, the range of network addresses corresponding to H+BEDV currently are hardcoded into the packet filter rules. The range was used to reduce the likeliehood of an impact to our rules should things change at the anti-virus company. However, if the IP addresses assigned and utilized by H+BEDV changes, then the corresponding rule will have to be corrected. Using the pflog custom script redirected into tail ("pflog | tail") after a failed virus update attempt ("/usr/lib/AntiVir/antivir --update") will show any rejected HTTP packets and the corresponding IP address. Another alternative is to simply permit outgoing web traffic to any IP address (with the exception of private addresses) which can be argued to be more than sufficient. Being this paranoid protects against malicious code that is not sophisticated enough to display the packet filter from attacking other servers.

The self-signed IMAP server certificate will need to be periodically reissued since it will expire at some point. By default, the certificates are issued for one year, but the "-days" option to the openssl command within the /usr/local/share/mkimapdcert script can be modified before issuing a certificate.

Viruses and Rejected Attachments

The Protector program will intercept any attachments that are of a questionable type, don't match the stated type in the attachment header, or is considered a virus by the anti-virus software. The administrator can revive a rejected mail attachment using the procedure shown at http://www.lowth.com/protector/bin/view/Protector/DocGuide100.

Additionally, the quarantine directory (/var/protector/rejects) will need to be manually periodically cleaned out.

Check your Configuration

Verify that IMAP is SSL enabled

We should not see a simple response such as " * OK IMAP ready." from the IMAP server which would imply non-SSL IMAP. The advantage of SSL encrypted channels is that handshaking occurs early on to negotiate the encryption and getting it started before the tunneled protocol begins to run and any users begin to enter passwords or other sensitive information. Examining the maillog indicates that the connection was recieved by the IMAP server, but was not the proper protocol. Later when the virus detection is tested, a true IMAP client will be used to retrieve mail and the settings will be made to show the connection is SSL protected to fully prove it is in fact SSL.

Action	Results
telnet 127.0.0.1 11993	Trying 127.0.0.1... Connected to 127.0.0.1. Escape character is '^]'.
CAPABILITY	Connection closed by foreign host.
tail /var/log/maillog	Sep 20 02:01:28 mail imapd-ssl: Connection, ip=[127.0.0.1] Sep 20 02:01:58 mail imapd-ssl: couriertls: accept: error:14760FC:lib(20):SSL23_GET_CLIENT_HELLO:unknown protocol

Verify what ports are open when looking locally

While logged into the IMAP server, execute the "netstat -an" command to display the state of all sockets and display IP addresses instead of hostnames. The first entry shows a process listening on the local loopback address of 127.0.0.1 at TCP port 11993. This is the Courier IMAP server. The remaining entries of interest are all UDP ports. There are three entries relating to NTP which is UDP port 123. The NTP daemon is listening on this port on two different addresses: the loopback as well as the IP address for the physical network interface. There is a wildcard entry as well (*) which corresponds to the socket used when the NTP daemon acts as a client to issue outgoing NTP requests to time servers. Lastly, there is an entry for UDP port 514 which is the syslog protocol. The syslog daemon opens up this socket so that it can use for client access as well. The "-u" option is not present when syslogd is running, so it is not configured to accept syslog messages originating remotely.

```
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
tcp        0      0  127.0.0.1.11993        *.*                    LISTEN
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
udp        0      0  192.168.0.78.123       *.*
udp        0      0  127.0.0.1.123          *.*
udp        0      0  *.123                  *.*
udp        0      0  *.514                  *.*
Active UNIX domain sockets
Address    Type   Recv-Q Send-Q     Inode      Conn     Refs    Nextref Addr
0xe0f93210 dgram       0      0       0x0 0xe0f38980        0x0 0xe0fad100
0xe0f93630 dgram       0      0       0x0 0xe0f38980        0x0 0xe0fadec0
0xe0f936e0 dgram       0      0       0x0 0xe0f38980        0x0 0xe0f38bc0
0xe0f93160 dgram       0      0       0x0 0xe0f38980        0x0        0x0
0xe0f93000 dgram       0      0 0xfd51e1f8        0x0 0xe0fd6380        0x0 /dev/log
```

Performing a "ps -ax" command shows a listing of all processes shows a number of items. The init process is the fundamental process running (notice how the process ID is 1). The syslog daemon is the next entry (syslogd). Followed by the packet filtering daemon (pflogd). The next couple processes relate to the IMAP Courier package. The cron process handles running at scheduled times automatically and is utilized within this installation. There will likely be some processes which show up in the display relating to the current shell session used to perform the ps command such as a C shell (csh) and the actual command (ps). Lastly, OpenBSD supports four terminals accessible from the console and the four getty processes service any of this activity.

```
  PID TT   STAT      TIME COMMAND
    1 ??  Is      0:00.01 /sbin/init
14201 ??  Is      0:00.02 syslogd
27499 ??  Is      0:00.08 pflogd
 9264 ??  I       0:00.00 /usr/local/libexec/couriertcpd -address=127.0.0.1 -st
```

```
14833 ??  I      0:00.00 /usr/local/libexec/courierlogger imapd-ssl
21626 ??  Is     0:00.01 cron
 6202 ??  Is     0:00.17 ntpd
17814 C0  Is     0:00.04 -csh (csh)
 3157 C0  R+     0:00.00 ps -ax
 1313 C1  Is+    0:00.00 /usr/libexec/getty Pc ttyC1
23415 C2  Is+    0:00.00 /usr/libexec/getty Pc ttyC2
10218 C3  Is+    0:00.00 /usr/libexec/getty Pc ttyC3
 9984 C5  Is+    0:00.00 /usr/libexec/getty Pc ttyC5
```

Scan for Vulnera bilities - Nessus

No issues were found with the packet filtering activated. The following issues were pointed out when the test was performed again with the packet filtering disabled.

Nessus run against the IMAP server with the packet filter active results in it not being able to contact it sufficiently to test it.

```
Nessus Scan Report
------------------

SUMMARY

 - Number of hosts which were alive during the test : 0
 - Number of security holes found : 0
 - Number of security warnings found : 0
 - Number of security notes found : 0

TESTED HOSTS

DETAILS

-------------------------------------------------------
This file was generated by the Nessus Security Scanner
```

First of all, Nessus incorrectly identifies this machine as a Mac.

ICMP Timestamp Replies:
These are not permitted through the packet filter rules, but it could be useful to disable this capability in case a problem develops with the packet filtering. No method is currently available with the shipping version of OpenBSD 3.1 to avoid replying to ICMP timestamp requests short of a packet filter. There is a patch available that could be applied and the /etc/sysctl.conf updated to use the new capability to suppress IMCP timestamp replies. Refer to http://monkey.org/openbsd/archive/ bugs/0204/msg00205.html for more information on the patch. It does not appear to be a significant enough issue to warrant bothering with the patch. Hopefully this will be available within OpenBSD 3.2 and can then be easily turned on in /etc/sysctl.conf.

NTP Server:
Nessus points out that ntpd has been vulnerable to buffer overflows in the past and indicates that it is very important to upgrade to the most recent version of the software to avoid known security issues. Since we have just installed this system with the latest daemon, this is not an immediate issue. However, this points out that we should be sensitive to this being a possible weakness of our system which warrants our future diligence. The advantage of running the NTP daemon is that this machine can serve time to other machines within the local network. If this is not a compelling enough reason, then an alternative is to not run the daemon and use the ntpdate command within cron periodically. This will not be a problem immediately, but once enough machines are present within an installation it becomes imperative to supply your own time in some fashion. Public time servers will not be pleased when time is requested by 10 different machines for a single facility.

```
Nessus Scan Report
------------------

SUMMARY

 - Number of hosts which were alive during the test : 1
 - Number of security holes found : 0
 - Number of security warnings found : 2
 - Number of security notes found : 2

TESTED HOSTS

 192.168.0.78 (Security warnings found)

DETAILS
```

+ 192.168.0.78 :
. List of open ports :
  o general/tcp (Security notes found)
  o general/icmp (Security warnings found)
  o general/udp (Security notes found)
  o ntp (123/udp) (Security warnings found)

. Information found on port general/tcp

Nmap found that this host is running MacOS 8.5

. Warning found on port general/icmp

The remote host answers to an ICMP timestamp request. This allows an attacker to know the date which is set on your machine.

This may help him to defeat all your time based authentication protocols.

Solution : filter out the ICMP timestamp requests (13), and the outgoing ICMP timestamp replies (14).

Risk factor : Low
CVE : CAN-1999-0524

. Information found on port general/udp

For your information, here is the traceroute to 192.168.0.78 :
192.168.0.78

. Warning found on port ntp (123/udp)

An NTP server is running on the remote host. Make sure that you are running the latest version of your NTP server, has some versions have been found out to be vulnerable to buffer overflows.

*** Nessus reports this vulnerability using only
*** information that was gathered. Use caution
*** when testing without safe checks enabled.

If you happen to be vulnerable : upgrade
Solution : Upgrade
Risk factor : High
CVE : CVE-2001-0414

This file was generated by the Nessus Security Scanner

NTP Protections

Temporarily disable the packet filters on the IMAP server with the command "pfctl -d" (can be enabled later with "pfctl -e"). From another machine on the local network with the ntpq binary (part of NTP package) available:

Action	Results

ntpq 192.168.0.78	ntpq>
assoc	192.168.0.78: timed out, nothing received ***Request timed out ntpq>
quit	#
ntpdc 192.168.0.78	ntpdc>
sysinfo	192.168.0.78: timed out, nothing received ***Request timed out ntpdc>
addserver 192.168.0.7	MD5 password:
<CR>	Invalid password ntpdc>
quit	#
ntpdate 192.168.0.78	20 Sep 14:49:09 ntpdate[2674[: adjust time server 192.168.0.78 offset -0.369271 sec

The ntpq command permits an NTP daemon to be monitored and an IP or hostname can be supplied to remotely monitor NTP. Remote access to the NTP running on the IMAP server is attempted with the "assoc" command which should show the time servers it is using, but it does not respond. Similarly the ntpdc command allows an NTP daemon to be queried concerning its configuration and the configuration can even be udpated. The modification capabilities are authenticated by the NTP daemon. Once again a command is sent to query for information about the NTP daemon, but this time the command is "sysinfo" which is supposed to show various system variables. Secondly, an attempt is made to ask the NTP daemon to begin retrieving time from an additional time server. This demonstrates that a local machine on the network cannot query or modify the configuration of NTP on the IMAP server.

The ntpdate command is used to adjust the time of the test machine from the IMAP server. This shows that the IMAP server permits other machines on the network to get its time. Thus, the IMAP server doubles as a time server for the local network.

Verify what ports are open with nmap

Nmap is a general portscanner with OS fingerprinting abilities. This was used to verify that only the ports intended to be open are such. Each open port represents a possible avenue of attack similar to doors and windows within a home to be protected against burglary. A myriad of different options are available within nmap and many were used. Both TCP and UCP as well as ICMP echo request were checked. These tests were performed from a Linux machine within the local network. Normally a machine will respond to a SYN packet to a TCP port with a RST packet if the port is closed, and nmap takes note when this is not seen and marks the port as filtered. Similarly a closed UDP port would normally result in an ICMP port unreachable message unless something along the way is preventing this message from being returned. So UDP ports not returning any response are considered filtered as well. So filtered can be considered as closed except that the attacker "knows" you are doing something to shield the machine if they can get some other sort of reaction from the machine revealing its presence.

Scans were run with the host-based firewall up first to establish the protections in place normally. The first scan was a default nmap run against the machine which terminated very quickly because it couldn't ping the machine. Therefore the -P0 option is used for all further scans to avoid attempting the initial ping and forge ahead. The second scan was for the standard TCP ports with a normal TCP connect scan which found our one TCP port of 993 corresponding to SSL encrypted IMAP. The next scan is of UDP ports which did not find any open ports. Since we are permitting local machines to access NTP which is present on UDP port 123, we might expect to see this show up. However, our packet filter only accepts packets with a source port of 123, so that is likely the reason this probe was not successful. The next scan performs an OS fingerprinting check along with a few specific TCP ports with a TCP connect scan. The OS fingerprinting is not able to identify the machine, but notes that conditions are not optimal for such detection.

```
# nmap (V. 2.54BETA22) scan initiated Fri Sep 20 16:45:34 2002 as: nmap -oN tmp 192.168.0.78
# Nmap run completed at Fri Sep 20 16:46:04 2002 -- 1 IP address (0 hosts up) scanned in 30 seconds

# nmap (V. 2.54BETA22) scan initiated Fri Sep 20 16:46:04 2002 as: nmap -P0 -sT -oN tmp 192.168.0.78
Interesting ports on  (192.168.0.78):
(The 1541 ports scanned but not shown below are in state: filtered)
Port       State      Service
993/tcp    open       imaps
# Nmap run completed at Fri Sep 20 16:49:13 2002 -- 1 IP address (1 host up) scanned in 189 seconds

# nmap (V. 2.54BETA22) scan initiated Fri Sep 20 16:49:14 2002 as: nmap -P0 -sU -oN tmp 192.168.0.78
All 1453 scanned ports on  (192.168.0.78) are: filtered
# Nmap run completed at Fri Sep 20 17:18:29 2002 -- 1 IP address (1 host up) scanned in 1755 seconds

# nmap (V. 2.54BETA22) scan initiated Fri Sep 20 17:56:39 2002 as: nmap -P0 -sT -O -p 80,143,993,11993 -oN
tmp 192.168.0.78
Warning:  OS detection will be MUCH less reliable because we did not find at least 1 open and 1 closed TCP port
Insufficient responses for TCP sequencing (0), OS detection may be less accurate
Insufficient responses for TCP sequencing (0), OS detection may be less accurate
```

Interesting ports on (192.168.0.78):
Port State Service
80/tcp filtered http
143/tcp filtered imap2
993/tcp open imaps
11993/tcp filtered unknown
No exact OS matches for host (test conditions non-ideal).
TCP/IP fingerprint:
SInfo(V=2.54BETA22%P=i386-redhat-linux-gnu%D=9/20%Time=3D8BA7BF%O=993%C=-1)
TSeq(Class=TR%IPID=RD)
T1(Resp=N)
T1(Resp=Y%DF=Y%W=403D%ACK=S++%Flags=AS%Ops=MNWNNT)
T2(Resp=N)
T2(Resp=N)
T3(Resp=N)
T3(Resp=N)
T4(Resp=N)
T4(Resp=N)
T5(Resp=N)
T5(Resp=N)
T6(Resp=N)
T6(Resp=N)
T7(Resp=N)
T7(Resp=N)
PU(Resp=N)
PU(Resp=N)
Nmap run completed at Fri Sep 20 17:57:03 2002 -- 1 IP address (1 host up) scanned in 24 seconds

The same nmap scans are run again once the packet filter has been disabled (pfctl -d). The really interesting thing right out of the bat is that the TCP scan does not find any ports while the one with the packet filter enabled found the SSL enabled IMAP port of 993. The reason for this is that the packet filter daemon is the process actually listening normally on this port and redirecting it to the TCP port 11993 on the loopback interface. Now the UDP port scan finds both the NTP daemon listening on port 123 and the syslog daemon listening on port 514. We have attempted to configure the syslog daemon to not listen for remote logs, so this is troublesome with no rational explanation readily at hand. The process list earlier confirms that it does not run with the -u option that is supposed to be needed to make it listen on this port. The packet filter is normally protecting us so we don't have an open door normally here. Once again the OS is identtified as a Mac.

nmap (V. 2.54BETA22) scan initiated Fri Sep 20 18:26:38 2002 as: **nmap -oN tmp 192.168.0.78**
All 1542 scanned ports on (192.168.0.78) are: closed
Nmap run completed at Fri Sep 20 18:26:55 2002 -- 1 IP address (1 host up) scanned in 17 seconds

nmap (V. 2.54BETA22) scan initiated Fri Sep 20 18:26:55 2002 as: **nmap -P0 -sT -oN tmp 192.168.0.78**
All 1542 scanned ports on (192.168.0.78) are: closed
Nmap run completed at Fri Sep 20 18:27:11 2002 -- 1 IP address (1 host up) scanned in 16 seconds

nmap (V. 2.54BETA22) scan initiated Fri Sep 20 18:27:12 2002 as: **nmap -P0 -sU -oN tmp 192.168.0.78**
Interesting ports on (192.168.0.78):
(The 1451 ports scanned but not shown below are in state: closed)
Port State Service
123/udp open ntp
514/udp open syslog
Nmap run completed at Fri Sep 20 18:27:46 2002 -- 1 IP address (1 host up) scanned in 16 seconds

nmap (V. 2.54BETA22) scan initiated Fri Sep 20 18:27:47 2002 as: **nmap -P0 -sT -O -p 80,143,993,11993 -oN tmp** 192.168.0.78
Warning: OS detection will be MUCH less reliable because we did not find at least 1 open and 1 closed TCP port
All 4 scanned ports on (192.168.0.78) are: closed
Remote operating system guess: MacOS 8.5
Nmap run completed at Fri Sep 20 18:27:48 2002 -- 1 IP address (1 host up) scanned in 1 second

Verifying Anti-Virus Protection

From a machine with email out capability, visit http://www.eicar.org/anti_virus_test_file.htm and download the various generic test files to use to safely verify an anti-virus program is working properly. At the bottom of the resultant web page download each of the files within the "Download" table: eicar.com, eicar.com.txt, eicar_com.zip, and eicarcom2.zip. You will likely need to temporarily disable your anti-virus software on the system used and, if not, an investigation should be commenced as to why it did not prevent these files from being downloaded. Send various emails with these as attachments to one of the email address of a POP account that is now setup to be handled by the new IMAP server. Remember to reactivate virus protection on the sending system.

Configure an MUA IMAP client such as Outlook Express to check the new IMAP server. These directions should be roughly applicable to any client. Create a new account by selecting the pulldown item "Tools -> Accounts" and an "Internet Accounts" window should appear. Select "Add -> Mail".
A new window titled "Internet Connection Wizard" should appear. Walk through the items entering information. The absolutely critical items are as follows:
Incoming mail server type: IMAP
Incoming mail server: mail
Account name: *<imap user>*

After creating the account, the server needs to be marked as an SSL enabled. Select the new account, click Properties, select the Advanced tab and check "This server requires a secure connection (SSL)" and the port number will authomatically update to 993

Define the hostname and its corresponding IP address used on the CN line (Common Name) within the enterprise's naming service or locally on each client machine.
Whatever is used within the certificate must be used by the client software to reference the server or else you will likely see errors on the client side.
C:\WINNT\I386\HOSTS:

<192.168.0.78> mail mail.*<sans.org>*

Wait until the POP polling interval has passed (5 minutes suggested) and then access the corresponding IMAP account via an IMAP client within the local SOHO network. Verify that each of the attachments were intercepted and quarantined by the anti-virus software. The attachment should be replaced by the output from the anti-virus scanner.

```
From: Fred Citizen
To: Joe User
Date: Thu, 19 Sep 2002 22:06:46 -0500
MIME-Version: 1.0
Content-type: Multipart/Mixed; boundary=Message-Boundary-11411
Subject: test 1 eicar
Message-ID: <3D8A4A76.2593.12778FD@localhost>
Priority: normal

--Message-Boundary-11411
Content-type: text/plain; charset=US-ASCII
Content-transfer-encoding: 7BIT
Content-description: Mail message body

test 1 eicar

---
Fred Citizen


--Message-Boundary-11411
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 7bit
Content-Description: Warning Message - /tmp/protector.7221/virus_scan_output

--- Warning message from your e-mail virus checker (protector 1.00.6) ---

AntiVir / OpenBSD Version 2.0.4-5
Copyright (C) 1994-2002 by H+BEDV Datentechnik GmbH.
All rights reserved.

Loading /usr/lib/AntiVir/antivir.vdf ...

VDF version: 6.15.0.7 created 10 Sep 2002

AntiVir license: XXXXXXXX for Joe User, Planet Earth

/tmp/protector.7221/decoded
 Date: 20.09.2002  Time: 03:13:15  Size: 69
 ALERT: [Eicar-Test-Signatur virus] /tmp/protector.7221/decoded <<< Contains code of the Eicar-Test-Signatur
virus
 not removable
```

```
----- scan results -----
 directories:       0
       files:       1
    infected:       1
    repaired:       0
     deleted:       0
     renamed:       0
   scan time: 00:00:01
------------------------
Thank you for using AntiVir.

    Content-type: text/plain; charset=US-ASCII
    Content-transfer-encoding: 7BIT
    Content-description: Text from file 'eicar.com'
    X-Copy-Of-Original:

--Message-Boundary-11411--
```

References

Ackley, Jason. "kernel/2576: add sysctl for ICMP timestamp replies." OpenBSD Bug Archive. 27 Apr. 2002. URL: http://monkey.org/openbsd/archive/bugs/0204/msg00205.html (20 Sept. 2002).
Coene, Wouter. "The OpenBSD Packet Filter HOWTO." 20020405.2. 5 Apr. 2002. URL: http://www.openbsdjournal.org/pf-howto/html/ (20 Sept. 2002).
Connors, David. "RE: aide 0.9 and invalid lstat()." The Mail Archive - aide. 17 July 2002. URL: http://www.mail-archive.com/aide@cs.tut.fi/msg00127.html (20 Sept. 2002).
Eicar. "Anti-Virus test file." Eicar Online. 3 Sept. 2002. URL: http://www.eicar.org/anti_virus_test_file.htm (14 Sept. 2002).
Free Software Foundation. "Bison." 27 Aug. 2001. URL: http://www.gnu.org/software/bison/bison.html (17 Sept. 2002).
Free Software Foundation. "Gdbm - Replacement for the 'dbm' and 'ndbm' libraries." URL: http://www.gnu.org/directory/gdbm.html (14 Sept. 2002).
Free Software Foundation. "GNU Make." 20 Apr. 2002. URL: http://www.gnu.org/software/make/make.html (14 Sept. 2002).
Free Software Foundation. "The GNU Privacy Guard." 11 Sept. 2002. URL: http://www.gnupg.org/ (16 Sept. 2002).
Fyodor. "Nmap stealth port scanner." 10 Aug. 2002. URL: http://www.insecure.org/nmap/ (20 Sept. 2002).
"gettytab Man Page." FreeBSD File Formats Manual. URL: http://man.dnswatch.com/cgi-bin/htmlman?gettytab+5 (15 Sept. 2002).
"gettytab Man Page." NetBSD Programmer's Manual. URL: http://www.tac.eu.org/cgi-bin/man-cgi?gettytab+5 (15 Sept. 2002).
Guenther, Philip. "Welcome to procmail.org." URL: http://www.procmail.org/ (14 Sept. 2002).
H+BEDV. "AntiVir for servers" Products. URL: http://www.hbedv.com/produkte/products.htm#Server (14 Sept. 2002).
H+BEDV. "Your Gateway to File Access Control." Dazuko. URL: http://www.dazuko.org/ (20 Sept. 2002).
Lehti, Rami. "AIDE - Advanced Intrusion Detection Environment." URL: http://www.cs.tut.fi/~rammer/aide.html (17 Sept. 2002).
Lehti, Rami. "The Aide Manual." URL: http://www.cs.tut.fi/%7Erammer/aide/manual.html (20 Sept. 2002).
Lowth, Chris. "Chris Lowth's "Protector" - GNU Licenced e-mail virus blockade (1.00.6) ." URL: http://www.lowth.com/protector (15 Sept. 2002).
Mavroyanopoulos, Nikos and Sascha Schumann. "Mhash." URL: http://mhash.sourceforge.net/ (17 Sept. 2002).
Mills, Dave. "Time Synchronization Server." 4 Aug. 2002. URL: http://www.eecis.udel.edu/~ntp (14 Sept. 2002).
Mullet, Dianna and Kevin Mullet. Managing IMAP. Cambridge: O'Reilly & Associates, Inc., 2000.
OpenBSD. "OpenSSH." 1.154. 1 Sept. 2002. URL: http://www.openssh.com/ (14 Sept. 2002).
Provos, Niels. "Documentation and Frequently Asked Questions." OpenBSD. 1.143. 11 Sept. 2002. URL: http://www.openbsd.org/faq/index.html (13 Sept. 2002).
Provos, Niels. "Security." OpenBSD. 1.221. 11 Aug. 2002. URL: http://www.openbsd.org/security.html (13 Sept. 2002).
Psionic Technologies. "Psionic LogSentry." URL: http://www.psionic.com/products/logsentry.html (20 Sept. 2002). Raymond, Eric Steven. "The fetchmail Home Page." 6 Sept. 2002. URL: http://tuxedo.org/~esr/fetchmail/ (14 Sept. 2002).
Schilling, Jörg. "CDRecord." URL: http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html (16 Sept. 2002).
Shaffer, George. "Hardening OpenBSD Internet Servers." GeodSoft Website Consulting. URL: http://geodsoft.com/howto/harden/ (13 Sept. 2002).
Streib, M. Drew. "Public Key Server Commands." URL: http://www.us.pgp.net/pgpnet/pks-commands.html (20 Sept. 2002).
Varshavchik, Sam. "Courier IMAP." URL: http://www.inter7.com/courierimap/ (14 Sept. 2002).