Some SecurityBasics: FreeBSD

ccTLD Workshp

September, 2005
Nairobi, Kenya

Hervey Allen
Network Startup Resource Center

FreeBSD vs. Linux

Security models are almost identical, but actual implementation is a bit different.

We include some FreeBSD-specific steps and tips here...

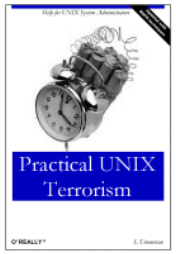
Core security concepts

Set the stage... In the end your goal is?:

- **Availability**
  Are our systems up and running?
- Maintain confidentiality.
- Keep data safe from intruders.
- **Integrity:** protect from loss or change.
- **Authentication**
  Is this person who they claim to be?
  Is this person allowed access?

Steps to take: Realistic

- Run only the services that are necessary
- Stay up-to-date and patch services as needed
  - I.E., sign up for mailing lists!
- Restrict root access to services
- Restrict access to services via tcpwrappers if appropriate
- Restrict access to your box usin firewall services (ipfw, ipf, pf)
  - Buffer overflow attacks. Be aware of them.



Steps to take cont.

- Log events and understand your logs.
- Back up your server's data!
- Consider using network intrusion detection software (snort)
- Think about physical security
- Test your security model

Steps to take: Next level

These items are discussed here:

http://ws.edu.isoc.org/workshops/2005/ccTLD-Nairobi/day3/security/security-full.pdf

- Use secure passwords and force your users to use them
  - Initial level is easy... (use PAM)
  - Advanced level is work (cracking)
- Consider if you need quotas
- Install intrusion detection software on your server (file change alerts)
- Don't forget about your clients
  - Your network is probably infested

Steps to take: Another view

Here's someone who would disagree:

http://www.ranum.com/security/computer_security/editorials/dumb/

His key points:

- *Default Permit* = whitelisting vs. blacklisting
- *Enumerating Badness* = keeping track of all exploits
- *Penetrate and Patch* = patching poorly designed software
- *Hacking is Cool* = learning how to hack is a waste of time
- *Educating Users* = if it works, then it should have worked
- *Action is Better Than Inaction* = action can be painful

A few resources

The FreeBSD Handbook:

- /usr/share/doc/en/books/handbook/index.html
- Chapter 14: Security (this is pretty paranoid)
- *Practical UNIX and Internet Security (a "classic")*
  - http://www.oreilly.com/catalog/puis3/
- *Mastering FreeBSD and OpenBSD Security*
  - http://www.oreilly.com/catalog/mfreeopenbsd/

Security repository with references and examples:

- http://nsrc.org/security/

Reduce number of services cont.

To see what is running you could use:

- `lsof -i (if installed)`
- `netstat -an -f inet`
- `ps -auxw | more`
- `sockstat -4`
- `fstat (with grep, read man page)`

Know what each and every item is.

Simplify – remove any and all services you are not using.

Use cryptographic methods to access services

ccTLD registry split these services out!

- POP/IMAP with SSL only.
- Consider TLS-Enabled SMTP.
- Don't run or allow telnet.
- Remove FTP replace with SCP or SFTP.
- Anonymous FTP is OK, but be careful if you allow user uploads.
- Require HTTPS (HTTP over SSL) for sensitive information. Critical for registries!

How to enforce good passwords

By default FreeBSD allows for completely insecure passwords. Test this using `passwd` as a user.

You can use cracklib with Pluggable Authentication Modules (PAM).

Cracklib keeps a user from creating trivial passwords.

You can find cracklib here:

- /usr/ports/security/cracklib

You should enable it here:

- /etc/pam.d/passwd

Requires installing cracklib and uncommenting one line in /etc/pam.d/passwd.

Cracklib

From "`locate cracklib`" under FreeBSD 5.4 after installation :

```
/usr/local/libdata/cracklib
  /usr/local/libdata/cracklib/pw_dict.hwm
  /usr/local/libdata/cracklib/pw_dict.pwd
  /usr/local/libdata/cracklib/pw_dict.pwi
  /usr/local/man/man3/cracklib.3.gz
  /var/db/pkg/cracklib-2.7_2
  /var/db/pkg/cracklib-2.7_2/+COMMENT
  /var/db/pkg/cracklib-2.7_2/+CONTENTS
  /var/db/pkg/cracklib-2.7_2/+DESC
  /var/db/pkg/cracklib-2.7_2/+MTREE_DIRS
```

As you can see cracklib is installed, a cracklib dictionary, and the PAM cracklib shared library.

You can install via "`pkg_add -r cracklib`" or by compiling in */usr/ports/security/cracklib*

More cracklib

Taken directly from the cracklib README file:

```
4) it's MIND-NUMBINGLY THOROUGH!

(is this beginning to read like a B-movie flyer, or
what?)

CrackLib makes literally hundreds of tests to
determine whether you've chosen a bad password.

It tries to generate words from your username and
gecos entry to tries to match them against what
you've chosen.

It checks for simplistic patterns.

It then tries to reverse-engineer your password into
a dictionary word, and searches for it in your
dictionary. (> million entries!)

- after all that, it's PROBABLY a safe(-ish)
password.  8-)
```

Extra: more ways to control users

Look in to /etc/login.conf if you wish to define login classes for your users to control their access to resources.

FreeBSD Handbook section 13.7

/usr/share/doc/en/books/handbook/users-limiting.html

Consider file system quotas.

FreeBSD Handbook section 16.14

/usr/share/doc/en/books/handbook/quotas.html

Back up your server's data!

Pretty hard to stress this more. If your security is compromised what will you do without a backup? How many here do this?

A few basic items to consider are:

- What needs to be backed up.
- How often do you need to backup?
- Where will your backup media be in case of disaster (fire, flood, earthquake, theft)?
- What happens in case of total loss?
- What tools will you use? Tar, Arkeia, cpio, dump, dd, rsync with ssh?

Tools to use for backups

- **Arkeia**: commercial product:
  - http://www.arkeia.com/
  - http://nsrc/security/#backups
- **dd**: convert and copy a file.
  - `man dd`
  - `dd if=/dev/ad0 of=/dev/fd0/bootsector.bin bs=512 count=1`

Backs up a boot sector to a floppy.

  - `dd if=/dev/fd0/bootsector.bin of=/dev/ad0 bs=512 count=1`

Recovers from floppy to ad0. Be *very* careful doing this!

Tools to use for backups cont.

- **cpio**: copy files to and from archives:
  - cpitool: http://www.nickb.org/utils/
  - `man cpio`
- **dump**: ext2/ext3/ufs filesystem backup.
  - `man dump`
- **rsync:** remote copy.
  - `man rsync` (not installed by default)
- **tar**: read
  - `man tar`

A few practical backup tricks

You can use ssh and tar together to quickly backup parts of your server. For instance, to backup all /home directories to another server as a single image:

```
root@machine1# tar xzvf - /home/ | \
ssh machine2 “cat > machine1-homes.tgz”
```

Or, you can use rsync over ssh if you wish to keep directories synchronized between two locations. FreeBSD uses ssh by default with rsync:

```
rsync -av . remote:/home/docs/
```

rsync with ssh and ssh keys

In /etc/periodic/daily/ you can set up a cron script to do the following:

```
rsync -a /var/www/html/ \
backup.machine:/var/www/html/
```

This recursively copies your root web documents to a backup machine using rsync via ssh. Note no “v” (verbose) option was used.

If you use the “--delete” option in rsync, then files removed on your local machine would be removed on the remote machine as well when you run this.

Log events and understand your logs

This is time consuming – even with the many tools that are available.

You need to go through each service running and decide if you want to log events from this service. This has already been partially done for you in /etc/syslog.conf under FreeBSD.

Ideally logs should be created or saved off your server. A cracker will alter your logs to cover their tracks.

Networking monitoring/logging

A few useful network monitoring tools:

- **Snort:** heavily used network intrusion-detection system with built-in packet rules for common attack. Find all things Snort at http://www.snort.org/
- **Nagios**: monitors services running on hosts on your network as well as resources. Can monitor you of events via email, pager, etc. Find this at http://www.nagios.org/.
- **nmap**: network exploration tool and security scanner can identify machines and services on your network. Find this at http://www.insecure.org/nmap/.
- **ntop**: from http://www.ntop.org/ gives full featured protocol analysis of who's talking to whom on your network. Includes graphical reports and web interface.

Caveat: these tools can get you in trouble. Be sure you have permission to run them.

Automated logging

To configure what is logged read "man syslog.conf" for full details on how this file is formatted.

FreeBSD sends a daily summary of events and system status generated by cron to root by default.

Consider using a central logging server. You can use /etc/syslog.conf to send events to another server via your network.

Yet more logging...

A few useful tools to monitor activity:

- **Swatch**: Simple WATCHer is available from http://swatch.sourceforge.net/ or in the port collection in /usr/ports/security/swatch. Will watch for "trigger" events in your logs and notify you immediately.
- **syslog and periodic:** see "`man syslog`" and "`man periodic`" to understand how daily log and system activity summaries are generated in FreeBSD.
- See http://nsrc.org/security/#logging for some more tools.

Patching your software

- As needed download patches for the services you run. You should be notified of these via the mailing lists mentioned.
- For your OS the vendor will often provide specific patches or update installers.
- For FreeBSD the FreeBSD project will provide port updates or new packages.
- Or, use cvsup and ports. If software is a port and it is patched, then a simple "`make`" in /usr/ports/category/package/ may do the trick.

Where to find some security mailing lists

General security mailing lists:

- BugTraq: http://www.securityfocus.com/
- CERT: http://www.cert.org/
- Rootshell: http://www.rootshell.com/

For Apache, Bind, Exim and SSH:

- http://www.apache.org/
- http://www.isc.org/ *(Bind)*
- http://www.exim.org/
- http://www.openssh.org/

FreeBSD Security Notifications Mailing List:

- http://lists.freebsd.org/mailman/listinfo/freebsd-security-notifications

Think about physical security

All the security in the world does nothing against a disgruntled employee, server sitting out in the open, people who copy keys, and so on.

Backups: where do you physically keep your them? Who has access to them. Are they separate from your server?

Logs: are they on a separate and physically secure log server? Printed to a separate printer?

Bootloader password and encrypted files: what happens if someone walks off with your machine?! Or, how about just the hard drive(s)?

Physical access = total access

Consider if some services should run under the inetd tcpwrapper

- Access control for services is done in /etc/hosts.allow (hosts.deny is deprecated).
- /etc/inetd.conf determines what services will run under the inetd wrapper.
- Enable /etc/inetd in /etc/rc.conf with:
  - inetd_enable="YES"
- What does inetd provide? ==>

What does inetd provide?

- The inetd daemon (service) listens for network packets for each service started in /etc/inetd.conf.
- inetd saves on memory and resources as a service is only started if a packet arrives for it, but it's better not to use inetd for a loaded service like http.
- You can control how packets arrive or don't arrive on a service-by-service basis in a detailed manner using inetd.

inetd vs. ipfw

- Note: FreeBSD doesn't use xinetd.
- ipfw permits full control over packets arriving for a service or server.
- ipfw provides a more complete ruleset that you can apply to a service, including more fine-grained control over icmp and udp packets.
- ipfw is part of the kernel, thus it is more efficient, but you'll need to recompile to get kernel vs. module access to ipfw, ipf, or pf.
- inetd has (imho) an easier syntax to understand.
- inetd can send messages for rejected items.

More inetd information

If you are interested in all the parameters you can specify on a service-by-service basis in both /etc/inetd.conf and /etc/hosts.allow, and when you start the inetd daemon, then see:

- man inetd
- man hosts_access
- man hosts_options

Restrict root access to a minimal set of services

Check for files with setuid/setgid bits running as root. If you don't need these files, or users don't need to run them, then remove this bit (chmod 000)

Consider running a service in a "sandboxed" environment using chroot.

Consider running a service under a different userid if possible.

Practical restriction tips ==>

Practical root restriction tips

To find all files with setuid or setgid bits set on a machine you can do:

```
find / -perm +6000 -type f -exec ls -ld {} \; >
setuid.txt &
```

You'll have a file listing all setuid/setgid files (but not scripts) on your machine.

You can turn off setuid or setgid bits by doing either:

```
chmod 0nnn filename
chmod 0000 filename
```

Be aware of what your changes imply. FreeBSD 5.4 ships preconfigured with many setuid & setgid files and warns if additional files are set.

Practical root restriction tips cont.

Use `chroot` to run services with their own root directory – i.e. in a "sandbox" or "jail".

You can use the FreeBSD `jail` facility.

Several services already run "sandboxed" by default, including ntalk, comsat, and finger

The named service has configuration options in /etc/defaults/rc.conf.

See FreeBSD Handbook 14.3.2 for more details.

How apache runs as user "www"

Taken directly from */usr/local/etc/apache/httpd.conf*:

```
# If you wish httpd to run as a different user or group, you must run
# httpd as root initially and it will switch.
#
# User/Group: The name (or #number) of the user/group to run httpd as.
#  . On SCO (ODT 3) use "User nouser" and "Group nogroup".
#  . On HPUX you may not be able to use shared memory as nobody, and the
#    suggested workaround is to create a user www and use that user.
#  NOTE that some kernels refuse to setgid(Group) or semctl(IPC_SET)
#  when the value of (unsigned)Group is above 60000;
#  don't use Group "#-1" on these systems!
#
User www
Group www
```

Buffer overflow attacks

A Cracker pushes more data on to a services buffer than space provides. They can "break out" of the program space and execute arbitrary commands on your system with the privileges of the compromised service.

Many security patches deal with newly discovered buffer overflow holes.

The Linux world has several solutions for this, but also has more problems with this issue.

Configure and use an IDS

- Intrusion Detection System = IDS
- Network Intrusion Detection System = NIDS
- And, System Integrity Checking is a generic term for this.

An IDS monitors network traffic and warns if suspicious behavior is detected.

A System Integrity Checker looks for changes to files that are not expected and warns you of these.

For a list of many tools see http://nsrc.org/#integrity

Snort intrusion detection system

Snort from http://www.snort.org/ is a very popular tool to detect unexpected network events using a known set of rules and patterns. This is a signature-based IDS. We will be using Snort tomorrow.

Additional Snort add-ons include (much of this is now built-in to the product as well):

- **ACID**: Anaylsis Console for Intrusion Databases. Web front-end to IDS alert database(s). Good for large site. From http://acidlab.sourceforge.net/.
- **Sguil**: Snort GUID for Lamerz. Complex system to analyze possible IDS events with tools such as ethereal and TcpFlow as well as Snort. From http://sguil.sourceforge.net/.
- **Snort_inline**: from http://snort-inline.sf.net/. Detect intrusions and react to them.
- **SnortSam**: from http://www.snortsam.net/ to update firewalls on the fly to deal with attacks.

Restrict access to your box using IP firewall services (ipfw)

FreeBSD 5.4 ships with no less than three "ready-to-go" firewall solutions. These are:

1.) IPFIREWALL: or *ipfw*. Version 2, or ipfw2, comes with FreeBSD 5.4. Sample (outdated) ruleset in /etc/rc.firewall if installed.

2.) IPFILTER: or *ipf* (the "Handbook's pick").

3.) Packet Filter Firewall: or *pf* from the OpenBSD project.

Detailed discussion can be found in:

- /usr/share/doc/en/books/handbook/firewalls.html

Firewalling cont.

From the Handbook:

The configuration of the IPFW software is done through the ipfw(8) utility. The syntax for this command looks quite complicated, but it is relatively simple once you understand its structure.

There are currently four different command categories used by the utility: **addition/deletion**, **listing**, **flushing**, and **clearing**. Addition/deletion is used to build the rules that control how packets are accepted, rejected, and logged. Listing is used to examine the contents of your rule set (otherwise known as the chain) and packet counters (accounting). Flushing is used to remove all entries from the chain. Clearing is used to zero out one or more accounting entries.

Firewalling cont.

To use ipfw you should place ipfw rulesets in /etc/rc.conf.

Logging is recommended when you first build your ipfw ruleset to help debug what you are doing.

A couple of example ipfw rules:

```
ipfw add deny tcp from evil.doers.org to nice.people.org 22
ipfw add deny log tcp from evil.crackers.org/24 to nice.people.org
```

We explain these on the next page ==>

Firewalling cont.

This command will deny all packets from the host evil.doers.org to the ssh port of the host nice.people.org:

```
ipfw add deny tcp from evil.doers.org to nice.people.org 22
```

The next example denies and logs any TCP traffic from the entire crackers.org network (a class C) to the nice.people.org machine (any port).

```
ipfw add deny log tcp from evil.crackers.org/24 to nice.people.org
```

Firewalling cont.

Before starting:

- Read FreeBSD Handbook on ipfw and/or firewalls
- Read "`man ifpw`" - "`man ipf`" - "`man pf`"
- See pf's comprehensive user guide here:
  - http://www.openbsd.org/faq/pf/

- Setting up a useful and functioning ruleset can be quite complex.
- The FreeBSD Handbook's firewall discussion is *excellent* and you should use this.

Test your security model

Connect to your machine(s) externally and see if your model works!

Run some security scanning software against your machine.

A common tool is `nmap`.

Another tool is Nessus, which we'll make available on your workshop CD.

See:

- http://www.insecure.org/nmap/
- http://www.nessus.org/.

Test your security model: nmap

Warning! Don't run nmap against machines or networks without giving prior notice!

Now try scanning your neighbor's box:

- `nmap 196.216.0.NN`
- `nmap -O 196.216.0.NN`
- `nmap -sS -O -p 1-1024 -v \ subnet.`*`nnn`*

Read the `nmap` man pages to figure out what's going on, and a decent `nmap` discussion:

`http://linuxgazette.tolix.org/issue56/flechtner.html`

Don't forget about your clients

Make sure that your users must connect to your servers in such ways as to help ensure the integrity of their data and their user accounts.

Insist on software clients that use encryption like SSH vs. Telnet, SCP/SFTP vs. FTP, POP/IMAP over SSL.

Human clients running their OS'es... Dealing with Windows security issues such as viruses, Windows Updates, worms, spyware, etc...

Virus scanning software to defang email on your server?

Scripts as well – can rename files like .exe, .pif, .com, .scr, .vbs, .bat to fn.ft.*txt*.

Social issues. Security is inconvenient. For instance, Windows *still* does not ship with SSH – This is painful.

Later we'll look at Windows XP and 2000 practical security tips.

Some resources

CERT (Coordinated Emergency Response Team)

- http://www.cert.org/ and http://www.us-cert.gov/cas/index.html

Nice List of Security Resources for Linux/UNIX

- http://www.yolinux.com/TUTORIALS/LinuxSecurityTools.html

nmap: Network exploration tool and security scanner

- http://www.insecure.org/nmap/

O'Reilly Books

- http://www.oreilly.com/

SANS Computer Security and Mailing Lists

- http://www.sans.org/ and http://www.sans.org/newsletters/risk/

Security Documents from nsrc.org

- http://nsrc.org/security/ and http://nsrc.org/freebsd-tips.html

And, don't forget your own local help at http://www.afnog.org/!

More resources

The FreeBSD Handbook Security Section

- http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/security.html

FreeBSD Website "intrusion detection" Software

- http://www.freebsd.org/cgi/ports.cgi?query=intrusion+detection&stype=all

FreeBSD Security Notifications Mailing List

- http://lists.freebsd.org/mailman/listinfo/freebsd-security-notifications

Nessus Security Auditing Package

- http://nessus.org/

Conclusion

More security means less convenience, but a security breach can be the least convenient moment of all.

There is always a tradeoff between how much security you put in place and what services you are providing.

Your users may grumble, but they'll really grumble if their data is compromised – Remind them of this :-)