Scope and Scale

- Focus: FreeBSD - enterprise hardware support and most 'mainstream' of the open source BSD trees.
- Security refresher and some new and interesting BSD security information.
- Emphasis on host-based security, one of the first layers of the security 'onion' complimented with network-level security [defense in-depth].
BSD – making inroads in the Enterprise market

- BSD and systems w/ BSD frameworks being deployed in the enterprise and with the end user.
- Nokia firewalls - run FireWall-1 on IPSO [based on FreeBSD 3.2]
- Juniper's Internet backbone router products, designed for high-growth, high-capacity networks, use code from FreeBSD.
- Other commercial BSD implementors include Yahoo! and LinkExchange
The Basics

• If modifying an existing system, especially in a production environment, make backups!

• Unnecessary services - go through /etc/inetd.conf and rc.conf; disable what you don't need [inetd.conf now shipped with everything off by default] [rc.conf - disable sendmail, SMTP and submission ports 25/587]

• Work with the latest version of the OS - tracking STABLE is the best idea
Encrypted Communications

- Disable telnet (default in recent FreeBSD releases) and enable SSH. OpenSSH is included in the FreeBSD base system.
- Upgrade all your systems to OpenSSH 3.4p1 and use SSH version 2 with privilege separation.
- Enable the sftp subsystem built into the SSHv2 protocol rather than a standard ftpd implementation if possible.
- Set up public key authentication with SSH [DSA keys!] to prevent password transmission, encrypted or otherwise!
File System Lockdown

- Partition out as much as possible; /, /usr, /var, /tmp at a minimum. /home and /usr/local should be considered as well.
- Mount non /usr or / [for /sbin] filesystems with the 'nosuid' argument, especially /tmp.
- Search for and remove suid bits off of non-used binaries [especially uucp - setgid]
- Use the chflags to set variables such as sappnd on log files, schg on system binaries, etc.
- [Explain different securelevel aware file variables here - sappnd, schg]
Kernel Securelevels

- Kernel securelevels allow variable security level increases on the fly.
- Levels range from -1 -> 3, -1 and 0 are referred to as 'insecure mode'.
- Securelevels can only be raised, not lowered, once the system is in multi-user mode.
Kernel Securelevels [cont.]

- **Securelevel 1** - sappnd and schg flags cannot be disabled - LKMs may not be loaded or unloaded.
- **Securelevel 2** - Securelevel 1 + no writing to disks except for mount(2). Time changes clamped to +/- 1 second.
- **Securelevel 3** - Securelevel 2 + IPFW rules cannot be modified.
- **Schg flag on files in /, /bin, /usr/bin, /sbin, /usr/sbin/** for maximum effectiveness.
Sysctl and rc.conf variables

- [sysctl] net.inet.tcp.blackhole=2 and net.inet.udp.blackhole=1 - don't generate RSTs on connection attempts to ports with no socket listening [TCP] and doesn't generate an ICMP port unreachable message on a port with no socket listening [UDP]. This breaks traceroute.
- [rc.conf] kern_securelevel_enable=“YES”, kern_securelevel=“X” - enable kernel securelevel
- [rc.conf] icmp_drop_redirect=“YES” - drop ICMP redirect packets. you don't want these.
- [rc.conf] tcp_drop_synfin=“YES” - drop packets with SYN+FIN bits set. breaks RFC, do it anyway! SYN+FIN scans are frequent.
Secure your services

- Start potentially dangerous programs such as bind in a chroot'd environment. Many popular services now support chroot() jail functionality. [named, sshd, httpd]
- `log_in_vain="YES"` in rc.conf - show connections to non-listening tcp/udp ports - goes well with robust packet filtering ruleset.
- Use packet filtering software such as IPFW or ipfilter to restrict access to services, even if the machine sits behind a corporate firewall [defense in depth!]
Serving files with ftpd

- FreeBSD powers large FTP software sites like ftp.cdrom.com - securely!
- Put individual users in the /etc/ftpchroot file to restrict them to their $HOME.
- Start ftpd with `-l -l` to enable extended logging.
- If running an anonymous archive, use ftpd `-A` [only allow anonymous connections] and `-r` [read-only mode for the server]
• Start syslogd with the '-ss' flags to prevent the daemon from opening 514/udp.
• Centralize syslog to a central server in addition to local logging: *.* @remotehost.org
• Add /var/log/ftpd for ftp.*
• Add /var/log/security for security.* [IPFW logs on security facility; allows for parsing of ipfw logs via 'ipfw add deny log..' command.]
Nifty kernel tricks

- www.trojanproof.org trojan detection kernel patch [OpenBSD/FreeBSD] - alerts based on md5 variations on files executed on your system; works well with Tripwire/AIDE.
- cerber.sf.net - real time interception and logging of potentially dangerous system calls; execve(), ptrace(), setuid(), etc. all configurable via sysctl commands. excellent logging. [think entercept functionality for BSD]
- Disable BPF in your kernel - uncomment 'pseudo-device bpf [n]' in your kernel. This prevents an attacker from sniffing traffic coming off your connection.
Keeping people out

- Use TCP wrappers [/etc/hosts.allow] to allow / deny access to certain TCP services. FTP / SSH / other potentially non 'public' services [not as useful = HTTP and SMTP].
- Use AllowUsers / AllowGroups SSH configuration options to restrict SSH usage to certain users and groups. This works well along with TCP wrapper usage and privilege separation.
- Give users who only require ftp access the /sbin/nologin shell to prevent access to a 'real' shell.
Checking your system

- /usr/ports/security/nmap - port scan yourself to check for strange services.
- /usr/ports/security/whisker - audit your web server for potential vulnerabilities
- /usr/ports/security/tripwire-1.31 - academic source release of tripwire, file integrity assurance.
Other tips and tricks

• Use ntpdate to synch your clock with a time server [e.g. ntp.nasa.gov]. Crontab it routinely to keep it reliable.

• In /etc/ttys change the 'secure' flag to 'insecure' on each local TTY to prevent direct root login; login should always be done through a user account and then 'su' to root.

• Enable sudo for restricting the root password on your system; grant certain users root privileges for certain commands.

• Enable 'pseudo-device snp 4' and use the 'watch' command to non-interactively attach yourself to a user's tty. Nifty :)
Links to related material

- This presentation: http://www.subterrain.net/presentations/
- FreeBSD security advisories and info: http://www.freebsd.org/security/
- Free FreeBSD stuff courtesy of: FREEBSDMALL.COM. Thanks Murray!