Installing a Secure Linux Server

Linux Server Hardening Techniques

Shachar Shemesh

http://www.shemesh.biz

Why Harden a Server

- Need to remotely manage a server.
  - SSH is the best solution
- Need to keep the server's clock synchronized
  - NTP is the best solution
- Both SSH and NTP has had serious vulnerabilities in the past!
- Conclusion – we must trust the services we run as little as possible.

The Hardening Process

- Create layers of isolated services
- Minimize points of exposure
- Limit daemons' permissions to the bare minimum
- Set up mechanisms to alert about violations
- Make each breach harder

Minimize Points of Exposure

- Disable all unnecessary services
- Firewall rules to block access
  - Use TCP wrappers, such as *tcpd*
- Uninstall unnecessary software
  - Better yet – start with nothing installed, only install neccessary stuff
- Separate each daemon from the others
  - Permissions
  - File access

Privilege Separation

- In the past, all daemons ran as user "nobody".
  - "nobody" became the second most powerful user in the system.
- Today, nobody uses "nobody". Instead, each service has its own, unique, user.

Privileged Ports Problem

- Only "root" can bind to low (<1024) ports.
- Bind needs to run as user "named", and listen on port 53.
  - Inherent dropping privileges
    - Daemon's code binds to port 53 as "root", and then drops the priveleges.
    - Can do so permenantly or temporarily
  - External port binding
    - A wrapper (inetd) binds to the low port, and drops priveleges before running the daemon.

Change Root Jails

- Once a non-priveleged service has been compromised, attacker will seek escalation.
  - Local vulnerabilities
- Often relies on uncareful local configs.
- Vulnerable SUID executables
  - Another reason to have as little as possible installed

Change Root Jail (cont.)

- A subdir becomes the root (jail)
- Process cannot access files outside jail.
  - All files required for actual running must be inside the jail already.
- Some daemons have builtin support
  - Very easy to set up
  - Usually doesn't require imports
- Others don't
  - chroot before running
  - daemon's executable must be in jail

chroot Helper Tools

- ldd – analyzes a program's dynamic dependencies
- ln – create a link between files
  - Symbolic – resolved on use – cannot link to files outside the jail.
  - Hard – Cannot link files on different filesystems.
  - Will usually use symlinks **into** the jail.
- Debian only – makejail

chroot Challenges

- Binary packages install outside the jail.
  - Debian's "makejail" goes a long way torwards a solution.
- Different daemons may require the same data
  - mount -bind *olddir newdir* (≥2.4.x)
  - mount -rbind *olddir newdir* (≥2.4.x)
  - mount -move *olddir newdir* (≥2.5.1)
    - 2.6 is unreleased yet.

The Ultimate Jail – Virtual Machine

IsraeLinux User Group

- "User Mode Linux", or "UML"
  - The daemon process runs on an entirely different kernel.
  - Not emulation – no significant performance penalty

Security Patched Kernels

- OWL – openwall.com
  - Nonexecutable stack, early patching
- GRSecurity – grsecurity.net
  - OWL's nonexec stack, proc fs restrictions, some ACL support, many other enhancements.
- LSM – lsm.immunix.org
  - Allows per-syscall permissions hook
  - Infrastructure – plugins "sold" separately.