Introducing the Bastille Hardening Assessment Tool

Def Con 13 - Las Vegas 2005

Jay Beale
Security Consultant
Intelguardians Network Intelligence, LLC

Lead Developer
Bastille Linux Project

www.bastille-linux.org/jay/
www.intelguardians.com

Updated Slides

These slides are version 1.1.

The most recent version of these slides is at:

www.bastille-linux.org/jay/Talks/DC13-bastille.pdf

Background: Bastille Linux

- Bastille Linux is a hardening program for Linux, HP-UX, Mac OS X, and soon Solaris.
- Bastille educates the system administrator about hardening and automates the process.

Bastille Now Does Assessment

Bastille now has an “audit” function, so it can assess how well-hardened the system is.

“Well-hardened” can be defined by your company, by the Bastille team, or by your friendly neighborhood standards body.

Demo

We demonstrate Bastille in action here, assessing a system to generate both a list of items that are or aren't hardened as well as a score. We then harden the system more fully and generate a new assessment.

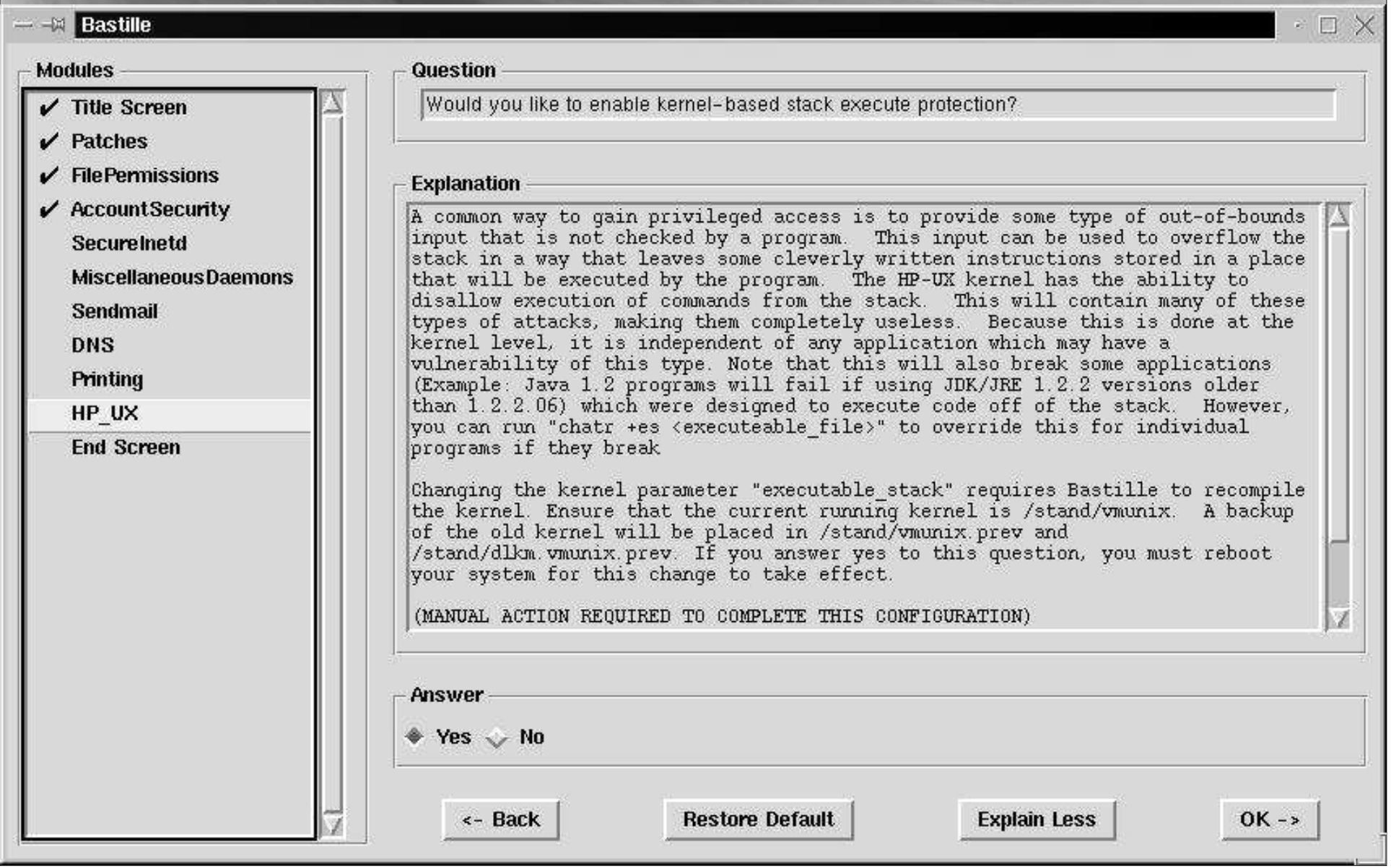
Why Do Assessment?

- Education

  Teaching the system administrator about hardening possibilities.

  Bastille has always had education as a second purpose.

Education


Bastille
Modules
✔ Title Screen
✔ Patches
✔ FilePermissions
✔ AccountSecurity
SecureInetd
MiscellaneousDaemons
Sendmail
DNS
Printing
HP_UX
End Screen
Question
Would you like to enable kernel-based stack execute protection?
Explanation
A common way to gain privileged access is to provide some type of out-of-bounds input that is not checked by a program. This input can be used to overflow the stack in a way that leaves some cleverly written instructions stored in a place that will be executed by the program. The HP-UX kernel has the ability to disallow execution of commands from the stack. This will contain many of these types of attacks, making them completely useless. Because this is done at the kernel level, it is independent of any application which may have a vulnerability of this type. Note that this will also break some applications (Example: Java 1.2 programs will fail if using JDK/JRE 1.2.2 versions older than 1.2.2.06) which were designed to execute code off of the stack. However, you can run "chatr +es <executeable_file>" to override this for individual programs if they break
Changing the kernel parameter "executable_stack" requires Bastille to recompile the kernel. Ensure that the current running kernel is /stand/vmunix. A backup of the old kernel will be placed in /stand/vmunix.prev and /stand/dlkm.vmunix.prev. If you answer yes to this question, you must reboot your system for this change to take effect.
(MANUAL ACTION REQUIRED TO COMPLETE THIS CONFIGURATION)
Answer
Yes
No
<- Back
Restore Default
Explain Less
OK ->

Why Do Assessment?

- Triage

  Bastille can rank a system against others in your organization, allowing you to decide where to spend your efforts.

Why Do Assessment?

- Auditing

  Bastille can audit your organization against guidelines from your organization, CIS, ISACA, or the organization of your choice.

Why Do Assessment?

- Compliance

  Bastille might aid in confirming compliance with HIPAA, Sarbanes Oxley, and GLBA.

  But Bastille's assessment functionality generates a score, which lets you do comparative analysis of due diligence.

Why Do Assessment?

- Network Protection

  Bastille can assess systems before they're placed on a network, allowing for policy enforcement and protection against the introduction of weak transient hosts.

Why Scoring?

- Scoring has a strong psychological power.
- People who would “get around to that later” soon find themselves working for that higher score. This happens!

Psychological Power

A security instructor beta-tested a scoring tool that I once wrote. His system scored 6/10. He got upset enough that he started running through easy hardening steps until it came up to an 8.

He didn't see himself as a 6/10 kind of guy!

Psychological Power

One major bank deployed a scoring tool on a completely voluntary basis and found their sysadmins competing for a higher score!

The base deployed Linux system might score around a 5.00/10.00.

Hardening / Assessment

There's a power to combining assessment with hardening.

Bastille creates hardening policy files when you run through it interactively, so....

Policy File for Both!

You can assess systems against the policy used to harden them.

You can create policy templates for different types of servers, speeding build time and giving you a “known standard hardening state” that each type should be in.

Skew Detection

Assessing a system often serves as a kind of “skew detection,” demonstrating where your patches or general system rot have weakened the state of the system.

Politics

This gets you skew detection, but also lets you make the case to others that systems should be hardened.

Politics is possibly the toughest challenge in hardening.

“Those systems score far worse than the ones we’ve hardened.”

Politics

- Bastille's Assessment mode can run read-only, so that it can measure a system that you're not allowed to modify in any way.
- Read-only mode can put people's minds at ease when they don't trust tools that modify a system.

Manual Hardening

Assessment really works for shops that want to do all of their configuration by hand.

Read-only means you don’t have to change your mindset.

Bastille-ix?

We're working toward creating a bootable CD that can run Bastille in read-only mode.

This is good for incident response, but also for politics:

"We won't modify the system **at all**, since we'll mount the hard drive read-only."

Why do I need to harden?

Hardening is the process of setting system configuration settings for greater resilience to attack.

But why should **you** harden a system?

Why Should You Defend?

The first objection that we normally face to hardening is that people think they'll never be attacked.

They think they're not interesting enough or high-value enough to be targeted.

Does anyone here actually think this? What about after being on DC wireless?

Low Value Targets Aren't?

First, not all targets are as low-value as they think…

Your box is useful as:

- …the next hop on the way to the target.
- …a good peer-to-peer host.
- …warez distribution site?
- …IRC server or bot?

Targets of Opportunity

Often, you're not even "targeted."

You're a "target of opportunity."

Your attacker has an exploit for one particular version of PHP and scans large swaths of the Internet looking for Web servers with that version.

Patching

- We apply a huge number of patches each year to operating systems, applications and networking hardware.
- Even if we're diligent about patching, it's not good enough – we still had windows of vulnerability.

Windows of Vulnerability

Window of Vulnerability (n):

The time period in which someone has a working exploit and our systems are still vulnerable.

Components of the Window

Windows of vulnerability are made up of three periods:

- Exploit exists, but vendor isn't aware of the issue. (0-day)
- Exploit exists, but vendor isn't done creating/testing the patch.
- Patch exists, but you haven't applied it yet.

Reactive Security?

A reactive security practice leaves you constantly playing the odds, hoping that your next window of vulnerability won't be the one where you get attacked.

- Patching
- Reactive firewall configuration
- Incident Response

Proactive Security

You can massively lessen your odds of being successfully compromised by:

- Configuring your hosts and your network to decrease their odds of being successfully hacked. (Hardening)
- Intelligently choosing policies ahead of time.

What is Hardening?

Hardening is the process of configuring a system for increased security.

It **does** involve deactivating unnecessary programs and auditing the configurations of those that remain.

It **does not** involve kernel-level modification of the system, along the lines of SELinux, Pitbull, or Trusted BSD/Solaris.

What is Hardening? 2/2

It **does** involve auditing the permissions and/or file access control lists and considering whether permissions are appropriate or too lax.

It **does** involve tweaking core operating system parameters to give users only what access they need, to the extent that the operating system allows this.

Let's be more specific, though.

Principles of System Hardening

Basically, system hardening comes down to tuning an operating system and its applications for Least Privilege and Minimalism.

Least Privilege

Each application or O/S component grants only what privilege each user type needs.

Minimalism

We configure the software for as few features as possible, to better our chances of not having vulnerable functionality active.

System Hardening: Practice

- Deactivate all system programs that aren't being used. (minimalism)
- Configure all remaining system programs for least privilege and functionality minimalism.
- Audit permissions/ACL's for least privilege.

Kernel level vs Hardening

Kernel-level technologies like SeLinux, TrustedOS, and other Mandatory Access Control (MAC) measures are complementary to hardening.

(Defense in Depth!)

Kernel level vs Hardening

- Often this technology **contains** the attacker after the exploit, where hardening would have **broken** the exploit.
- Example: If the exploit is in functionality that we've deactivated, say by not loading an unnecessary Apache module, the vulnerable code isn't even available on your machine!

Kernel level vs Hardening

- Kernel-level technologies require profiles for each major network daemon, at the least.
- Profiles can be created by your vendor, but the vendor can only make very general profiles if they want to avoid breaking things.
- You can learn to make profiles, but that requires medium to extensive amounts of training.

Kernel level vs Hardening

- You should definitely seek out kernel-level technologies, but understand their pros and cons.
- Continue to harden systems.

Hardening isn't Difficult

Hardening isn't as difficult as it might sound.

- The Center for Internet Security (CIS)'s Best Practices benchmarks are written for simplicity and ease of use by very novice system administrators.
- Black Hat offers a two-day hardening class that covers Solaris and Red Hat-like Linux distributions comprehensively.

Is Hardening Effective?

Hardening can be extremely effective at avoiding vulnerabilities.

Examples:

- NSA's Test of CIS's Guides
- Bastille Linux's test on Red Hat 6.0

Center for Internet Security

The Center for Internet Security produces simple industry best-practices system hardening guides.

The NSA's Information Assurance Directorate evaluated a system locked-down following CIS's Windows 2000 guide. **90 percent** of all the vulnerabilities in this platform were mitigated by the guide.

Hardening on Linux

Doing the same with the Linux guide got about 90-95% of all exploits mitigated.

Bastille Linux (Bastille Unix?)

Bastille is a programmatic solution that we're showing you today.

Bastille was written right after the release of Red Hat 6.0, before any vulnerabilities were discovered and published.

It could stop or contain almost every publicly released exploit.

Bastille Effectiveness

Red Hat 6.0 Vulnerabilities Stopped or Contained:

- BIND – remote root hole
- WU-FTPd – remote root hole
- lpd+sendmail – remote root hole
- dump/restore – local root privilege escalation
- gpm – console root-level privilege escalation

We didn't stop vulnerabilities in the man or nmh commands.

What is Bastille?

- Bastille can lock down these operating systems:
  - Red Hat: Classic, Enterprise, Fedora Core (*)
  - HP-UX
  - Mandrake/Mandriva Linux
  - Debian Linux
  - SuSE Linux (*)
  - Gentoo Linux
  - Mac OS X
  - Solaris?
- It can assess those that are marked with an *.

Why educate the sysadmin?

One of Bastille's real differentiators was that you could run it in interactive mode.

Bastille educates the sysadmin. Why?

Example: telnet

Example: we want to deactivate telnet.

If we do so without asking, we might “break” their remote administration interface.

We need to explain why telnet is bad: password stealing and session takeover.

We need to tell them that SSH is an alternative.

Bastille for Sysadmin Education

- “I've learned more just by going through the script than by hacking through a stack of O'Reilly books.
  (Maximum Linux Magazine)

- It explains itself extremely well during the course of a Bastille session. If you take the time to read this script's explanations of its own questions, you'll learn a lot about system-hardening.
  (Linux Journal)

Why Else Should I Use Bastille?

- Bastille is an automate-able solution, allowing you to create one policy file that you can apply to a huge number of similar systems.
- To apply a policy file to another system:

```
# scp /etc/Bastille/config root@another_host:/etc/Bastille/
# ssh root@another_host “bastille -b”
```

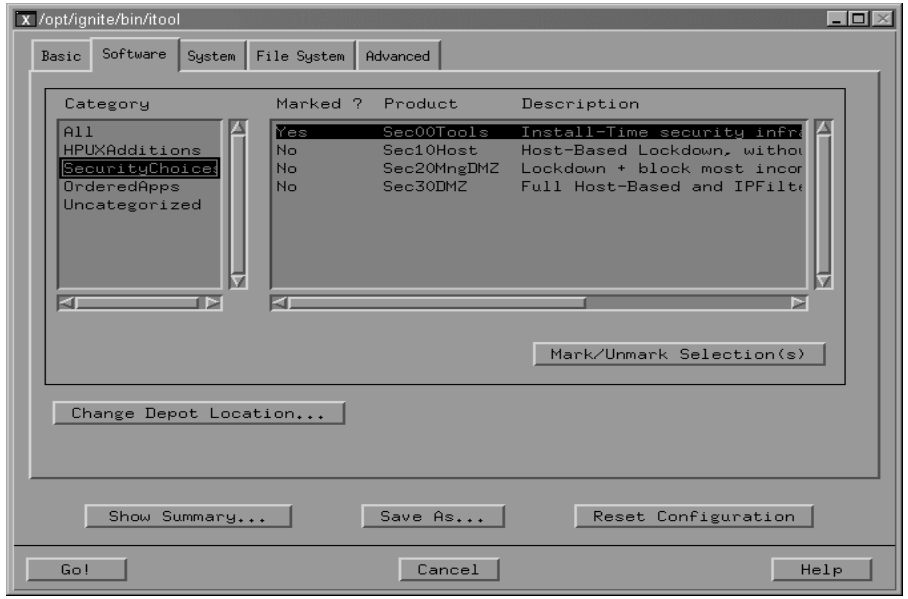
What if I don't have many hosts?

- Even with one or two systems, automation gets you **consistency**!
- Just as you create a standard build configuration file for your O/S installer, you can create a standard hardening profile for Bastille.
- Using saved policy files, you only have to choose your hardening steps once per release of the operating system.

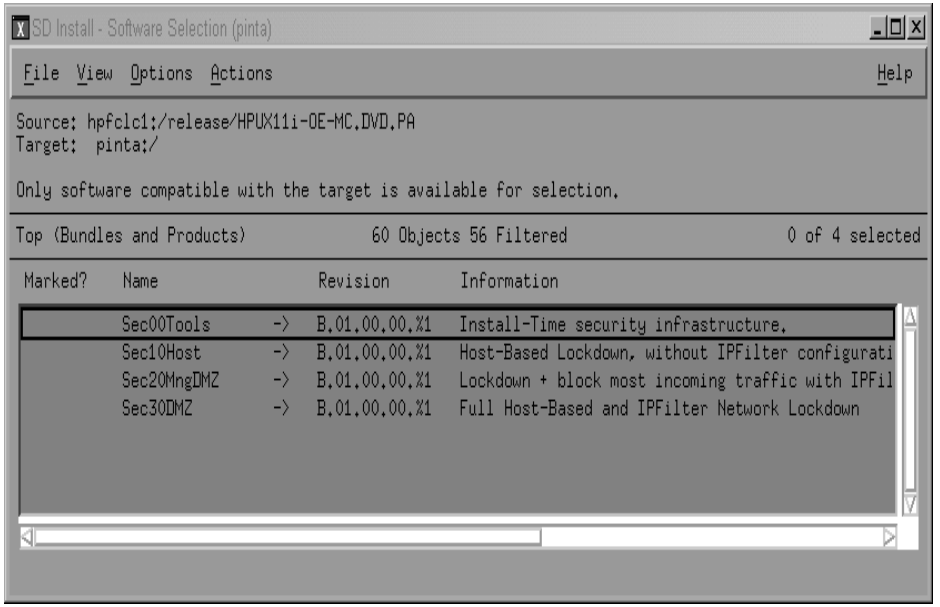
HP-UX Integration: Install-Time Security

- Deploy HP-UX into high threat environments quickly
  - make security or compatibility decisions suited to your needs
  - security tradeoffs no longer configured for the “generic user”
- Customers can be “secure-by-default,” at installation,
  - Can later revise settings with Bastille

1) Ignite/UX



2) Software Distributor



2) Manual

```
# swinstall –s <depot> -x autoreboot=true <level>
```

4) Update/UX

```
# update-ux –s <depot> <OE> <level>
```

Bastille Linux's Modules

- Looking at Linux functionality
  - Linux modules list
    - File Permissions
    - Account Security
    - Boot Security
    - Secure Inetd
    - Disable User Tools
    - Configuring PAM
    - Miscellaneous Daemons
    - Sendmail

Bastille Linux's Modules

- More Linux functionality
  - DNS
  - Apache
  - Printing
  - FTP
  - TMP Directory
  - Firewall
  - PSAD

What Does Bastille Do?

We can look a bit more at what Bastille actually does to a system, what particular things it can either harden or assess.

Bastille Linux's File Permissions Module

- Administration Utilities
  - Privilege reduction for management applications via the removal of world executable permissions
- Disabling SUID root permissions
  - "mount" and "umount"
    - File system activation and deactivation tools
  - "ping"
    - Network connectivity testing utility
  - "dump" and "restore"
    - file system backup and restoration utilities
  - "cardctl"
    - PCMCIA device control utility

Bastille Linux's File Permissions Module

- Disabling SUID root permissions (continued)
  - "at"
    - Individual task scheduling
  - "DOSEMU"
    - DOS emulation software
  - "inndstart" and "startinnfeed"
    - INN news server tools
  - "rsh", "rcp", and "rlogin"
    - Remote connection client utilities
  - "usernetctl"
    - Network interface control utility

Bastille Linux's File Permissions Module

- Disabling SUID root permissions (continued)
  - "traceroute"
    - General network configuration test utility
  - "Xwrapper"
    - X windowing system wrapper script for X binaries
  - "XFree86"
    - X server binary

Bastille Linux's Account Security Module

- Disable clear-text r-protocols (rlogin, rsh)
  - Removes execution permission from server binaries
  - Removes r-protocol service entries in inetd/xinetd configurations
  - Modifies system PAM configuration
- Enforce password aging
  - 180 day default cycle in "/etc/login.defs"
  - Handles removal of inactive accounts

Bastille Linux's Account Security Module

- Restrict "cron" to administrative accounts
  - "cron" scheduling can be useful, for legitimate and illegitimate ends
- Default system "umask"
  - Default configurations for bash, csh, ksh, and zsh.
- Restricting root login on tty's
  - Restricts root login on console

Bastille Linux's Boot Security Module

- GRUB configuration
  - Boot prompt password protection
- LILO configuration
  - Boot prompt password protection
  - Removes boot all boot prompt delay
- Securing the "inittab"
  - Disables "ctrl-alt-del" rebooting
  - Password protects single-user mode

Bastille Linux's Secure Inetd Module

- TCP Wrapper configuration
  - Default deny settings for inetd, xinetd and TCP Wrapper aware services
- Services using clear text protocol
  - Disables telnetd
  - Disables ftpd
- "Authorized Use Only" messages
  - Login time banner serves as an unwelcome mat
    - Possibly helpful in the prosecution of system crackers.

Bastille Linux's Disable User Tools and PAM Modules

- Disable User Tools
  - "gcc" complire
    - Removes user execution privileges from the "gcc" binary
- Configure PAM
  - The number of allowed core files is reduced to zero
  - Individual users are limited to 150 processes each
  - Individual files are limited to 100MB each
  - Console access is limited to a small group of users, including but not limited to the "root" user

Bastille Linux's Logging Module

- Logs status messages to the 7th and 8th virtual terminals
- Adds two addition log files to the basic setup
  - "/var/log/kernel" logs kernel messages
  - "/var/log/loginlog" logs all user login attempts
- Adds sensitivity to current system logs
  - "/var/log/syslog" will contain messages of severity "warning" as well as severity "error"
- Sets up remote logging
- Enables process accounting

Bastille Linux's Miscellaneous Daemons Module

- Disables apmd
  - Battery power monitor used almost exclusively by laptops
- Disables NFS
  - Network file system transfers data in clear-text and uses IP based authentication
- Disables Samba
  - CIFS server which transfers data in clear-text
- Disables PCMCIA services
  - Allows the use of easily removable credit-card-sized devices used almost exclusively by laptops

Bastille Linux's Miscellaneous Daemons Module

- Disables DHCP daemon
  - Used to distribute temporary IP (Internet) addresses to other machines.
- Disables GPM daemon
  - Used in console (text) mode to add mouse support
- Disables the news server daemon
  - Provides news services to outside machines
- Disables routed daemon
  - "routed" is a legacy daemon which provides network routing
    - Commonly replaced by "gated"

Bastille Linux's Miscellaneous Daemons Module

- Disables "gated" daemon
  - A daemon used to route network traffic
- Disable NIS server daemons
  - An NIS (Network Information System) server distributes network naming and admin info to other machines on a network.
- Disable NIS client daemons
  - Used to receive NIS information from a server.
- Disables SNMPD
  - Used to aid in management of machines over the network

Bastille Linux's Miscellaneous Daemons Module

- Disables Zeroconf mDNSResponder daemon
  - mDNSResponder broadcasts information on the network to find serves as well as config info.
  - Zeroconf is also called Rendezvous or Bonjour.

Bastille Linux's Sendmail Module

- Disables the sendmail daemon
- Configures a periodic run of sendmail to process the mail queue
- Disables SMTP (Simple Mail Transport Protocol) VRFY and EXPN commands for systems running a sendmail daemon.
  - The VRFY command allows connecting systems to "verify" the existence of a system user
  - EXPN allows connecting systems to "expand" user name aliases
- Run Sendmail semi-chrooted

Bastille Linux's DNS Module

- Configures BIND (Berkeley Internet Name Domain) services to be more secure
  - Configures a "chroot" jail for the BIND daemon
  - Configures the BIND daemon to run as a non-root user
  - Restricts zone transfers to avoid disclosure
  - Restricts recursion to break attack vectors
- Disables unneeded BIND services running on a system

Bastille Linux's Apache Module

- Disables Apache daemon if it is unneeded
- Hardens Apache configuration
  - Binds the Apache daemon to specified network interfaces
  - Prohibits Apache from following symbolic links
  - Disables Apache server-side includes
  - Prohibits Apache from executing CGI scripts
  - Disables Apache indexes, the auto generation of index files when an index file is not present
  - Disables Apache modules that aren't in use.

Bastille Linux's Printing, FTP, and TMP directory Modules

- Printing configuration
  - Disables the printing daemon lpd
  - Removes suid and gid bits from the lp and lprm commands
- FTP daemon configuration
  - Disables user privileges on the FTP daemon
  - Disables anonymous download
- TMP directory configuration
  - Configures TMPDIR and TMP environment variables for systems users

Bastille Linux's Firewall Module

- Configures an iptables, ipchains, ipf Firewalls
  - Zones network interfaces into three separate trust domains
    - Public Interfaces are completely untrusted
    - Internal Interfaces are untrusted but can have a configuration all together separate from the Public interfaces
    - Trusted Interfaces are completely trusted, e.g. the loopback interface
  - By protocol, service auditing
  - IP address source verification
  - IP Masquerading / NAT (Network Address Translation)

Bastille Linux's PSAD Module

- PSAD (Port Scan Attack Detection) Integration
  - Tunable port scan detection interval
  - Tunable port range scan threshold
  - Scanning tool signature reporting
  - Danger levels reports based on tunable packet thresholds
  - Configurable e-mail notification system
  - Automatic blocking of scanning IP addresses via host based firewall configuration
  - Boot time start up scripts

Programmer's Reference

- We've attached a programmer's reference to this talk.
- Help us extend Bastille further, with new content or new operating system support.
- You can definitely help if you don't code -- check out:

  http://www.bastille-linux.org/how-to-help.html

Adding to Bastille's Content

Bastille is in Perl, which makes it very easy to add onto.

It is designed around an easy-to-use API, so you can add items without knowing much Perl at all.

Adding an Item

To add an item, we must do two things.

First, we'll need to add the question text to a particular Modules's Questions/<module>.txt file.

(If we added a module, we'd have to add that module named to Modules.txt.)

LABEL: suiddump

SHORT_EXP: "Dump and restore are used for… extremely unlikely that… any problems with disabling SUID for dump and restore."

QUESTION: "Would you like to disable SUID status for dump and restore?"

REQUIRE_DISTRO: LINUX DB SE TB OSX

DEFAULT_ANSWER: Y

YN_TOGGLE: 1

REG_EXP: "^Y$|^N$"

YES_CHILD: suidcard

NO_CHILD: suidcard

PROPER_PARENT: suidping

Coding a New Item

Coding a new item is fairly simple:

```
if
   (&getGlobalConfig("FilePermissions","suiddump"
   ) eq "Y") {
      &B_remove_suid(&getGlobal('BIN',"dump"));
      &B_remove_suid(&getGlobal('BIN',"restore"));

  }
```

&getGlobalConfig(*module*,*question*) gives user's answer

Bastille API 1/4

- &B_chmod (*mode*,*file*)
  change the permission bits of *file* to *mode*
- &B_chmod_if_exists (*mode*,*file*)
  change the permission bits of *file* to *mode* if *file* exists
- &B_chown (*uid*,*file*)
  change the owner of *file* to *uid*
- &B_chgrp(*gid*,*file*)
  change the group owner of *file* to *gid*
- &B_remove_suid(*file*)
  remove suid bit of *file*
- &B_symlink(*file*,*new_link*)
  create a symbolic *new_link* to *file*

Bastille API 2/4

- &B_chkconfig_off(*script_name*)
  remove all S links to *script_name* from rcX.d dirs
- &B_chkconfig_on(*script_name)*
  create S links to *script_name* from rcX.d dirs
- &B_createdir(*directory*)
  make directory *directory*
- &B_cp(*file,destination*)
  copy *file* to *destination*
- &B_mknod (*prefix,file,suffix*)
  make a device node named *file* using the mknod
  command as in mknod (*prefix*) *file* (*suffix*)

Bastille API 3/4

- &B_blank_file (*file*, *pattern*)
  blanks *file* if no lines match *pattern*
- &B_append_line (*file*,*pattern*,*line*)
  appends *line* to *file* unless it matches *pattern*
- &B_insert_line (*file*,*pattern*,*line*,*line_to_follow*)
  inserts *line* into *file* after *line_to_follow*
  unless a line exists matching *pattern*
- &B_prepend_line(*file*,*pattern*,*line*)
  prepends *line* to *file* unless it matches *pattern*
- &B_replace_line(*file*,*pattern*,*line*)
  replaces lines in *file* matching *pattern* with *line*

Bastille API 4/4

- &B_hash_comment_line(*file,pattern)*
  hash-comments lines in *file* matching *pattern*
- &B_hash_uncomment_line(*file,pattern*)
  hash-uncomments lines in *file* matching *pattern*
- &B_delete_line(*file,pattern*)
  deletes lines matching *pattern* in *file*

Configuring Audit

Audit items are also not hard to add:

```
$GLOBAL_TEST{'FilePermissions'}{'suidping'}=
    sub {
   if (&B_is_suid($ping) or &B_is_suid($ping6)) {
      return $ASKQ;
   }
 return $SKIPQ;
 };
```

Test (Audit) API (1/3)

- B_is_service_off ($service_name)
  - Does rc-script/inetd-launched service run on boot?
- B_match_line ($file,$pattern)
  - Does $file contain a line matching $pattern?
- B_return_matched_line ($file,$pattern)
  - Returns lines matching $pattern in $file
- B_match_chunk ($file,$pattern)
  - Multi-line version of B_match_line
- B_is_package_installed ($package)
  - Is $package installed?

Test (Audit) API (2/3)

- IsProcessRunning ($pattern)
  - Are any processes running matching $pattern?
- B_is_executable ($file)
  - Is $file executable?
- B_is_suid ($file)
  - Is $file Set-UID?
- B_is_sgid ($file)
  - Is $file Set-GID?
- B_check_permissions($file,$perm)
  - Are $file's permissions at least as strong as $perm

Test (Audit) API (3/3)

- B_get_user_list ()
  - Output a list of users on the system
- B_get_group_list ()
  - Output a list of users on the system
- B_parse_fstab ()
  - Parses /etc/fstab into a special data structure
- B_parse_mtab
  - Returns a hash of currently mounted filesystems
- B_is_rpm_up_to_date($rpm,$ver,$rel,$epoch)
  - Checks whether $rpm is older than $ver,$rel,$epoch

Looking Further

Let’s look further at Bastille.

Bio and Contact Information

Jay Beale is a consultant with Intelguardians Network Intelligence, LLC, specializing in security architecture review and ethical hacking.

www.intelguardians.com

Also:

- Author of Center for Internet Security's Unix Scoring Tool
- Member of the Honeynet Project
- Author of Information Security Magazine, SecurityFocus and SecurityPortal articles
- Co-author of books on Snort, Nessus, as well as two hacker fiction books "Stealing the Network"
- Security articles at: **www.bastille-linux.org/jay**