Bastille Linux: Security Through Transparency

Jon Lasser
University of Maryland, Baltimore County (UMBC)
jon@umbc.edu

2000 March 24

Overview

- A Brief History of Bastille Linux
- Philosophy
- Step-By-Step Overview
- Lessons Learned

The "Linux Problem", Ca. Q3 ’98

Linux spreading rapidly in universities, but no centralized control

- Inexperienced administrators
- Wide-open systems
- Infrequent security updates

. . . and therefore frequent break-ins

The "Linux Problem"

It’s hard to do the right thing:

Linux is easy to install

... but hard to administer

(se

What UMBC Linux Got Wrong

Our most serious failure was social, not technical:

Despite the very large number of Linux users, especially at universities, we did not seek to share with others performing similar work.

Had we shared our plans and results with other schools, we would have duplicated less work and been able to accomplish more.

Sans ’99 Conference, project proposed

- improve university security
- harden defaults
- discourge dangerous and obsolete tools
- simplify customization of install
- to be based on Red Hat 6.0.

Why Red Hat 6?

- Market share
- Easy for beginners to install
- Fairly open by default
- Some Distributions Have Hardening Software
  - SuSe
  - Mandrake 7

...and Away From Bastille Linux?

- little happened immediately
- Admins are busy folk
- project had a distant payoff
- Red Hat 6.0 is more secure out-of-box, so pressure off.
- Difficult to make Red Hat's devel model work over the 'net
  - Monolithic, too Slow at Internet Speeds
  - Network and server reliability is still an issue
  - It's too hard to keep up with Red Hat!

A Change of Direction...

Instead of a Distribution, a Hardening script:

- Full Compatibility
- Community Support
- Less work to update for new releases
- Adaptable to other distributions
- Avoid export concerns by simply installing crypto via ftp from Europe

Releases To Date (1)

1.0

- The Conference Release

1.0.1

- Brown Paper Bag Release

1.0.2

- Additional Bugfixes

Releases To Date (2)

1.0.3

- Red Hat 6.1 support
- Included automation examples

1.0.4

- TUI
- Defaults for all choices
- No more single-user mode
- Mandrake 6.x support
- Bugfixes

The Future

Incremental Development

- patch security
- further audits and granularity
- better logging
- additional security
- stateful firewall

"Version 2.0" (New architecture)

- Multiple Back-ends possible
- Multiple Front-ends possible
- Canonicalize data in the middle
- More intelligent standard back-end

Philosophy of Bastille Linux (1)

A Living, Executable "Best Practices" Document

Based on Community Resources:

- SANS Securing Linux Step-By-Step
- Kurt Seifried's Linux Administrator's Security Guide

Leveraged Existing Code

- Jay Beale's Solaris Hardening Scripts

Philosophy of Bastille Linux (2)

Grounded in Open-Source Methodology:

- Many eyes (audit)
- Many minds (experience)
- Many arguments (community)

How Can We Stop Crackers?

What Bastille Linux Does

- Apply vendor-produced patches
- Disable unnecessary services
- Secure default configurations
- Set up a firewall

What Bastille Linux Doesn't Do (Yet)

- Automated techniques to scan for crackers
- Automated protection from certain attacks (StackGuard)

Basic Features Walkthrough (1)

- Completely Modular

- Install RPM Updates
  - Dynamic list from our server
  - As secure as commercial vendors

- SUID and Permissions Audits
  - Permissions audit based on SANS document, with changes
  - SUID audit granular, Permissions audit not (yet!)

Basic Features Walkthrough (2)

- Account Security
  - Use md5 password hashing
  - Use shadow passwords
- Create Second Admin Account
  - Like root, with different name
  - Root logins become a sign of intrusion
  - Controversial, but useful for some
  - Remember, all Bastille Linux functionality is optional!
- Install SSH
- Disable Dangerous R* Utilities
  - rlogin, rsh, rcp, etc.
  - Create empty .rhosts file for each user, root owned, mode 0400, empty
  - Mode 0400 empty /etc/hosts.equiv
  - Use ssh, scp, etc. instead

Basic Features Walkthrough (3)

- Protect Bootloader
  - Require password for single-user mode
  - Reduce prompt delay
  - Alter permissions to prevent users reading (necessary for passworded mode)
- Restrict Console Reboots (Control-Alt-Delete)
- Remote Access Restrictions
  - Extra logging for portscans
  - Security risks of telnet and FTP discussed
  - TCP Wrapper host-specific configuration
  - "Authorized Users Only" banners
- For Servers, Disable Compilers
  - A pinch more security, though admittedly not much
- Prevent remote root logins

Basic Features Walkthrough (4)

- Limit console logins to administrators
- Denial-of-Service Protections
  - Per-user limits
  - No core files
  - Limit users' file size
- Additional Logging
  - Direct to virtual consoles 7 and 8
  - Remote loghost
  - Separate local kernel and system logs
  - Separate local user login log
  - Process Accounting

Basic Features Walkthrough (5)

- Sendmail Configuration
  - Fix most known holes
  - Turn off daemon mode
  - Turn off VRFY/EXPN (anti-spam/recon)
  - Red Hat 6 already restricts relaying
- Restrict Cron Access
- Disable Unnecessary Daemons

Basic Features Walkthrough (6)

- Chrooted DNS server
  - Set this up, even if BIND not running, in case it's enabled later
  - Deactivate DNS server by default
- Apache Configuration
  - Deactivate, or bind to localhost only
  - Don't follow symbolic links
  - Disable server-side includes
  - Disable CGI scripts
- FTP Configuration
  - Disable user privileges
  - Disable anonymous access

Features Walkthrough: Firewall

- Aimed at experts (Created interactively, good defaults)
- Trusted, Public, and Internal interfaces
- TCP, UDP, ICMP audit logging
- Different services on different interface classes
- TCP, UDP, ICMP blocking
- Block source spoofed packets
- IP Masquerading

Features: Automation

Set up one machine, use the same configuration on hundreds or thousands of boxes

Several default setups:

- Firewall
- Mail server
- Web server
- Workstation

Lessons Learned

ESR's Cathedral and the Bazaar a Standard Model.

- 1. Every good work of software starts by scratching a developer's personal itch.
  - Our Verdict:True
- 2. Good programmers know what to write. Great ones know what to rewrite (and reuse).
  - Our Verdict:Unclear --- tending towards true
- 3. ''Plan to throw one away; you will, anyhow.'' (Fred Brooks, ''The Mythical Man-Month'', Chapter 11)
  - Our Verdict:True!

ESR's Aphorisms (2)

- 4. If you have the right attitude, interesting problems will find you.
  - Our Verdict:Unclear
- 5. When you lose interest in a program, your last duty to it is to hand it off to a competent successor.
  - Our Verdict: N/A (Hasn't Happened Yet)
- 6. Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging.
  - Our Verdict:True
- 7. Release early. Release often. And listen to your customers.
  - Our Verdict:True

ESR's Aphorisms (3)

- 8. Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone.
  - Our Verdict:True (we think)
- 9. Smart data structures and dumb code works a lot better than the other way around.
  - Our Verdict:True
- 10. If you treat your beta-testers as if they're your most valuable resource, they will respond by becoming your most valuable resource.
  - Our Verdict:True

ESR's Aphorisms (4)

- 11. The next best thing to having good ideas is recognizing good ideas from your users. Sometimes the latter is better.
  - Our Verdict:True
- 12. Often, the most striking and innovative solutions come from realizing that your concept of the problem was wrong.
  - Our Verdict:True
- 13. ''Perfection (in design) is achieved not when there is nothing more to add, but rather when there is nothing more to take away.''
  - Our Verdict:Unclear

ESR's Aphorisms (5)

- 14. Any tool should be useful in the expected way, but a truly great tool lends itself to uses you never expected.
  - Our Verdict:Unclear
- 15. When writing gateway software of any kind, take pains to disturb the data stream as little as possible -- and *never* throw away information unless the recipient forces you to!
  - Our Verdict: N/A (Not gateway software)
- 16. When your language is nowhere near Turing-complete, syntactic sugar can be your friend.
  - Our Verdict:True

ESR's Aphorisms (6)

- 17. A security system is only as secure as its secret. Beware of pseudo-secrets.
  - Our Verdict:True -- In fact, this has been a guiding principle all along!
- 18. To solve an interesting problem, start by finding a problem that is interesting to you.
  - Our Verdict:True
- 19: Provided the development coordinator has a medium at least as good as the Internet, and knows how to lead without coercion, many heads are inevitably better than one.
  - Our Verdict:True

The End

Please mail questions or comments to jon@umbc.edu
The Bastille Linux homepage is http://bastille-linux.sourceforge.net/

To subscribe to the announcement mailing list, send mail to bastille-linux-announce-request@lists.bastille-linux.org

To subscribe to the discussion list, please send mail to bastille-linux-discuss-request@lists.bastille-linux.org