Linux Security

Trent Jaeger
February 16, 2004

Systems Security: Web Page Before

Web Page After

Systems Security Problems

- Buffer overflows
- Authentication: Identify perpetrator
- Access Control: Integrity of web page
- Access Control: Containment of compromised app
- Patches: Update vulnerable applications
- Hardening: Run applications in hardening configs
- Audit: Capture behavior
- Intrusion Detection: Misuse or anomaly

Vulnerability Rate Is Rising!

Bugs Are Not a New Problem

January 1, 1973
- “…From a practical standpoint the security problem will remain as long as manufacturers remain committed to current system architectures, produced without a firm requirement for security. As long as there is support for ad hoc fixes and [ad hoc] security packages for these inadequate designs and as long as the illusory results of penetration teams are accepted as demonstrations of a computer system security, proper security will not be a reality.”
Will Firewalls Protect Us?

Oh Hey! I just love these things!… Crunchy on the outside, and a chewy center!

Systems Security View

Linux Access Control

- Discretionary Access Control
  - Mode bits (ugo)
  - Owner-administered
  - Setuid
- Mandatory Access Control – Linux Security Modules
  - Reference Monitor Interface (LSM interface)
  - Enforcement Mechanism (Module)
  - Mandatory Policy (Module-defined)

Reference Monitor Model

LSM Design Principles

- Interface w/i kernel
  - Race removal, Kernel translation, Kernel context
- Hook choices
  - Based on needs of modules
  - Restrictive of DAC checks
  - Permissive for capabilities checks
- Supports limited
  - Audit, Stacking

LSM Implementation

- Opaque security fields
  - Store security labels
  - Rejected by kernel maintainers in favor of xattrs
- Security hooks
  - Define the LSM interface
  - Accepted with reduced levels of indirection
    - security_ops->inode_ops->mkdir; security_inode_mkdir
- Generic security system call
  - Enable input from security-aware applications
  - Rejected by kernel maintainers; pseudo filesystem is used (selinuxfs)
    - A variety of file_operations are possible given the semantics of the system call
LSM Implementation (con’t)

- Register/Unregister
  - Hooks for security modules to register/unregister their use of the LSM interface
  - LSM recognizes only the primary module
  - `mod_reg_security` enables a second module to stack

- Capabilities Module
  - `capable` is a wrapper for an LSM hook
  - Copies capability bit vector duplicates task structure
  - `Cgapt/capset` hooks are added which duplicates syscall

Performance Impact

- LMBench microbenchmark: capabilities LSM
  - Replaces capability tests
  - Dummy static functions “return 0;”
  - Performance impact is noise except
    - Open/close/delete due to multiple checks in path resolution
    - Select impacted by the LSM Netfilter hooks (since removed)

LSM Status

- LSM hooks accepted in Linux 2.6
  - Excepting networking hooks
- Around 200 hooks
  - About 150 are for mediation
  - Others for allocation/free, labeling, ad hoc management
- SELinux module included in Linux 2.6
  - Example MAC policy under development by SELinux community
- Networking hooks were not all accepted
  - Present: NetFilter
  - Accepted: socket, socket_sock_rcv_skb
  - Rejected: Fragment, Encapsulate, Decode options

SELinux module

- Formerly
  - DTOS – integrated with Mach
  - Flask – integrated with Fluke
- Goals
  - Generic reference monitor checking for MAC
  - Flexible support for policies
- Issues
  - Simple mechanism is necessary
  - More invasive interaction of access control and system
- Paper
  - Microkernel implementation of SELinux module
  - Complicated by microkernel issues

Policy Flexibility

- **provide total security policy flexibility**
  - if the security policy can interpose atomically on any operation performed by the system.
- A more realistic approach is to
  - Identify that portion of the system state that is potentially security relevant and to control operations that affect or are affected by that portion of the state.
- Recall the definition of a reference monitor

Policy Flexibility (con’t)

- Flexibility Limits
  - Some (e.g., finer-grained) operations proceed outside of the control of the security policy
  - Restricts operations that may be performed by the reference monitor, and
  - Permits some system state to exist beyond the scope of a single access check
- Flexibility Issues
  - The need to revoke previously granted accesses,
  - The type of input required to make access decisions,
  - The sensitivity of policy decisions to external factors like history or environment, and
  - The transitivity of access decisions
- Revocation
  - Retracting capabilities (e.g., fd) based on previous policy
Simple Mandatory Mechanism

- Capability systems
  - ‘Pure’ capabilities
  - complex to control
  - Validated capabilities
  - validation unclear
- Request interposition
  - Def: a layer intercepts requests and authorizes
  - Example: Wrapper that checks access to an object server
  - Problem: Object server changes are invisible to wrapper
- Solutions require:
  - Ability to validate permissions on delegation and use
  - More invasive checks

Flask (SELinux) Design

- Object Manager
  - Integrate access checks within the object mgr
- Security Manager
  - Makes security decisions based on policy
- Object Manager operations
  - Object labeling
    - Security context: opaque value represents security-relevant info
    - SID: Security server fn \( x = \text{SID(security context)} \)
    - Object mgr must ask the security server to construct context and return SID
  - Client-server labeling
    - ‘Subject labeling’: Address space or self-restricted

Flask (SELinux) Design (con’t)

- Security decisions
  - Object manager requests
  - Check local cache
  - Check security server
  - Cache decisions
- Polyninstantiation
  - For partitioning shared spaces (ports, directories)
  - Request member sid for partition
  - Not implemented in SELinux

Flask (SELinux) Design (con’t)

- Kernel Mechanisms
  - Kernel is an object manager for memory
  - Subjects have SIDs and so do pages
  - Authorized on page faults
  - Also for IPCs, presumably for tasks
  - Some authorizations depend on relationships!
- Revocation
  - Keep object manager cache consistent with security server
  - Not so difficult for LSM implementation, but becomes an issue for external servers (XWindows)
  - Revocation trxn:
    - 3 steps: (1) notify relevant object mgrs; (2) make change; (3) notify security server
  - Problem: arbitrary delay by untrusted object managers

SELinux Installation/Administration

- Kernel configuration
  - CONFIG_SECURITY
  - CONFIG_SECURITY_NETWORK
  - Build kernel and SELinux module
  - Add each user and associated roles to the policy/users file
  - Configuration should not run X – work underway
  - Customize and build policy configuration
  - Build libsecure (SELinux aware library)
  - Build SELinux modified applications (e.g., login)
  - Edit files for default login contexts for users, cron, initrc scripts
  - Label all files using setfiles based on custom file->label map
  - Configure bootmanager to load new kernel and boot
  - Run setfiles again to label files created by old kernel during shutdown

Pluggable Authentication Modules (PAM)

- Unlike access control, Linux authentication is done outside the kernel
- PAM: shared libraries for constructing application authentication mechanisms
- Applications call pam_authenticate which does the rest
- Choose PAM configuration (/etc/pam.conf or in /etc/pam.d/)
- Load libraries automatically (/lib/security)
- NOTE: Need source code to pamify
**PAM Example: openSSH**

- Password verification is done by Linux-PAM
  - `pam_start`, `pam_authenticate`, `pam_end`
- `Pam.conf` specifications
  - `<service-name> <module-type> <ctl-flag> <module-path> <args>`
  - `sshd auth required pam_unix.so shadow nodelay`
  - `sshd password required pam_cracklib.so`
- PAM module types
  - authentication management: authenticated access
  - account management: non-authentication access (time of day)
  - session management: before/after actions
  - password management: update authentication token
- Modules are stacked based on order in configuration file

**Bastille Linux (system hardening)**

- A security hardening script
- Currently available for various Linux distros
  - SuSE 7.2, 7.3 and 8.0
  - TurboLinux 7.0
  - Mandrake (several versions)
  - RedHat (several versions)
  - Debian 2.2 and 3.0
- Also available on MacOS X and HP/UX

**Bastille Linux**

- Hardening Tasks
  - Set up firewall
  - Set-UID and Permissions Audit
  - Deactivate unnecessary daemons/apps
  - Tighten configuration of applications
  - SET is up PSAD (port scan attack detector)
- How is it used?
  - Utilizes Perl, Tk, or Curses for a user-interface
  - User answers questions pertaining to various aspects of system security
    - Upon completion of the questions, user ok's the changes and updates are made to the system via Perl scripts
    - A config file is created during session which can be reused for later use

**Other Linux Security Areas**

- IDS: Snort
- Audit: SNARE, LAuS (SuSE)
- Port scans detectors
- CryptoAPI: symmetric key, hashes, MACs
- Firewall: IPChains, NetFilter
- Patches?
- Policy development; Tresys, SLAT, Hitachi, Gokyo