Integrating Flexible Support for Security Policies into the Linux Operating System

http://www.nsa.gov/selinux

Stephen D. Smalley
sds@tycho.nsa.gov

Information Assurance Research Group
National Security Agency

(Slight modifications by David Martin for 91.562 in red)
Outline

• Motivation and Background
• What SELinux Provides
• SELinux Status and Adoption
• Ongoing and Future Development
Why Secure the Operating System?

• Information attacks don’t require a corrupt user.
• Applications can be circumvented.
• Must process in the clear.
• Network is too far.
• Hardware is too close.
• End system security requires a secure OS.
• Secure end-to-end transactions requires secure end systems.
Mandatory Access Control

- **MAC**: mandatory access control
  - Beware confusion with Media Access Control, Message Authentication Code

- A “missing link” of security in current operating systems.

- Defined by three major properties:
  - Administratively-defined security policy.
  - Control over all subjects (processes) and objects.
  - Decisions based on all security-relevant information.
Discretionary Access Control

- Existing access control mechanism of current OSes.
- Limited to user identity / ownership.
- Vulnerable to malicious or flawed software.
- Subject to every user's discretion (or whim).
- Only distinguishes admin vs. non-admin for users.
- Only supports coarse-grained privileges for programs.
- Unbounded privilege escalation.
What can MAC offer?

- Strong separation of security domains
- System, application, and data integrity
- Ability to limit program privileges
- Processing pipeline guarantees
- Authorization limits for legitimate users
MAC Implementation Issues

- Must overcome limitations of traditional MAC
  - More than just Multi-Level Security / BLP (Bell-LaPadula)
    - MLS/BLP: Strict hierarchy of security labels
      - Top Secret > Secret > Confidential > Unclassified
    - No reading from higher level, no writing to lower level
- Policy flexibility required
  - One size does not fit all!
- Maximize security transparency
  - Compatibility for applications and existing usage.
Prior Research Prototypes

• Distributed Trusted Mach (DTMach)
  – Outgrowth of TMach and LOCK OSes
  – Integrated flexible MAC framework into Mach OS

• Distributed Trusted Operating System (DTOS)
  – Improved design and implementation in Mach
  – Studies of policies, composability, security, assurability

• Flux Advanced Security Kernel (Flask)
  – Integrated DTOS security architecture into Flux OS
  – Added support for dynamic policies and revocation
  – Basis for SELinux
Decision to move to Linux

- Recognized need to move to a mainstream platform
- Past strategies not producing desired results
- National Security Council interest in Open Source
- Technology transfer opportunities
- Linux chosen as best alternative
SELinux provides Flexible MAC

- Flexible MAC integrated into Linux kernel
- Application of the Flask security architecture
- Integrated into major kernel subsystems
- Provides object class and permission abstractions
- Labels kernel objects with security contexts
  - Both in memory and on file system: processes, IPC mechanisms, and anything accessed through a file descriptor: files, devices, sockets, etc.
- Enforces access decisions on kernel operations
  - Basically, every interesting system call is subject to MAC checks
SELinux Policy Engine

- Referred to as the “security server” due to origins.
- Implements a combination of:
  - Role-Based Access Control
  - Type Enforcement
  - Multi-Level Security (optional)
- Security Policy specified through a set of configuration files.
Type Enforcement

- Domains for processes, types for objects
- Control access to objects (domain-to-type)
- Control process interactions (domain-to-domain)
- Control entry into domains
- Bind domains to code (through types)
Type Enforcement: Rules

• Let sshd bind a TCP socket to the SSH port.
  – allow sshd_t ssh_port_t:tcp_socket name_bind;
  
<table>
<thead>
<tr>
<th>domain</th>
<th>object</th>
<th>class</th>
<th>permission to grant</th>
</tr>
</thead>
</table>

• Let sshd read the host private key file.
  – allow sshd_t sshd_key_t:file read;

• Let sshd create its PID file.
  – allow sshd_t var_run_t:dir { search add_name };
  – allow sshd_t sshd_var_run_t:file { create write };
  – type_transition sshd_t var_run_t:file sshd_var_run_t;
    | domain   | obj-to-create | new-obj-type |
  |----------|--------------|-------------|
Role-Based Access Control

- Roles for processes
- Specifies domains that can be entered by each role
- Specifies roles that are authorized for each user
- Initial domain associated with each user role
- Ease of management of RBAC with fine granularity of TE
SELinux Status

• Initial public release in Dec 2000, regular updates
• Active public mailing list, >900 members
• Motivated development of Linux Security Module (LSM) framework (2001)
  – LSM adopted into Linux 2.5 development series (2002)
  – Provides infrastructure for supporting SELinux
• SELinux adopted into Linux 2.6 stable series (2003)
SELinux Adoption

• Integrated into Red Hat distributions
  – Fedora Core 3 or later
  – Red Hat Enterprise Linux 4 (supported product)
• Integrated into Hardened Gentoo for servers
• Partial support in Debian and SuSE
  – requires additional packages available separately
• Foundation for NetTop
• Basis for Trusted Computer Solution's Trusted Linux
• Port exists for FreeBSD 5 (SEBSD)
Ongoing Development

- Enhanced MLS support (TCS, IBM)
- Security-Enhanced X (originally NSA, now TCS)
- Enhanced Audit subsystem (IBM, Red Hat)
- IPSEC integration (IBM)
- Enhanced application integration (Red Hat)
- Policy tools / infrastructure (Tresys, MITRE, IBM)
- Scalability and performance (NEC, Red Hat, IBM)
Future Work

- Integrate SELinux into other userspace object managers.
- Modify other applications to better leverage SELinux.
- Enhance policy tools and infrastructure.
- Integrate with non-MAC policies (e.g. Crypto)
- Enhance revocation support.
- Develop flexible trusted path mechanism.
- Develop NFSv4 support and upstream it.
Questions?

• Download code and documents from http://www.nsa.gov/selinux
• Mailing list: Send 'subscribe selinux' to majordomo@tycho.nsa.gov
• Contact our team at: selinux-team@tycho.nsa.gov
• Contact me at: sds@tycho.nsa.gov
• SELinux for Distributions: http://selinux.sourceforge.net