



# ***Information System Security Operation*** **Security-Enhanced Linux**

## **Flexible Architecture, Configurable Security Policy, and Strong Access Controls**

### **Overview**

In order to provide a complete security solution, information must be protected with an appropriate security policy. Traditional approaches to secure operating systems have been limited by a narrow, hardcoded definition of the security policy. The Security-Enhanced Linux (SELinux) project is developing security mechanisms for the Linux kernel that are both strong and flexible.

We have worked on SELinux to provide strong access controls that can confine processes to least privilege, protect the confidentiality and integrity of processes and data, and support application security needs. SELinux has a flexible architecture and configurable security policy that can meet a wide range of security requirements.

### **Objective**

Systems must be able to enforce the separation of processes and data based on confidentiality and integrity requirements to provide system security. Operating system security mechanisms are the foundation for ensuring such separation. Existing mainstream operating systems lack the critical security feature required for enforcing separation: mandatory access control (MAC). Traditional MAC mechanisms have not been widely adopted because they are limited to a particular model of security that is poorly suited for many security requirements. Consequently, application security mechanisms are vulnerable to tampering and bypass, and malicious or flawed applications can easily cause failures in system security.

SELinux can be used to confine processes, including superuser processes, to least privilege, to protect the integrity and confidentiality of processes and data, and to support protected subsystems or assured pipelines. System

services such as Apache, BIND, and Sendmail can be limited to the minimal set of objects and permissions needed to perform their task, so that an exploit of a flaw such as a buffer overflow in a system service can be confined.

Likewise, client programs such as web browsers can be placed into separate security domains with limited access so that malicious mobile code can be confined. System software, configuration files, and log files can be protected against unauthorized modification.

Sensitive information can be protected against unauthorized disclosure or modification. Software of uncertain pedigree can be run in restricted security domains so that no serious harm can be done if the software is malicious. These are merely a few examples of the flexibility and security provided by SELinux.

### **Approach**

To address this objective, the NSA Information Assurance Research Group developed a flexible MAC architecture that can support a wide variety of security policies. After experimenting with several research prototypes of this architecture in the Mach and Fluke operating systems, the NSA implemented this architecture in the Linux 2.2 kernel, creating SELinux. We implemented additional controls for the kernel, developed an example security policy configuration to demonstrate how the system can be used to meet specific security objectives, and ported the code to the Linux 2.4 kernel.

The resulting system was presented to the Linux kernel developers. In response, the kernel developers described a general security framework they would be willing to consider for inclusion in the mainstream Linux kernel. The Linux Security Modules (LSM) project was started to develop such a framework. SPARTA, then

---

This work sponsored by the National Security Agency, Contract Number MDA904-01-C-0926.



## Security-Enhanced Linux

Flexible Architecture, Configurable Security Policy, and Strong Access Controls

McAfee® Research, contributed to the development of the LSM kernel patch, ensuring that it is adequate to meet the needs of SELinux, and reworked the SELinux implementation to use LSM rather than its own kernel patch. Several public releases of the LSM-based SELinux system have been made on the NSA SELinux web site, (<http://www.nsa.gov/selinux>). SPARTA and other LSM contributors have submitted LSM to the Linux kernel developers, and they have already merged significant portions of it into the mainstream Linux kernel.

SELinux assigns a collection of security attributes, called a security context, to each process and object in the system, and controls every system operation based on the relevant security contexts in accordance with the system security policy. The security policy logic, including the interpretation of security contexts, is encapsulated in a separate component. SELinux code is independent of the specific security policy and merely enforces the security decisions provided by the security server. A general security interface is defined between the security server and the rest of the SELinux code.

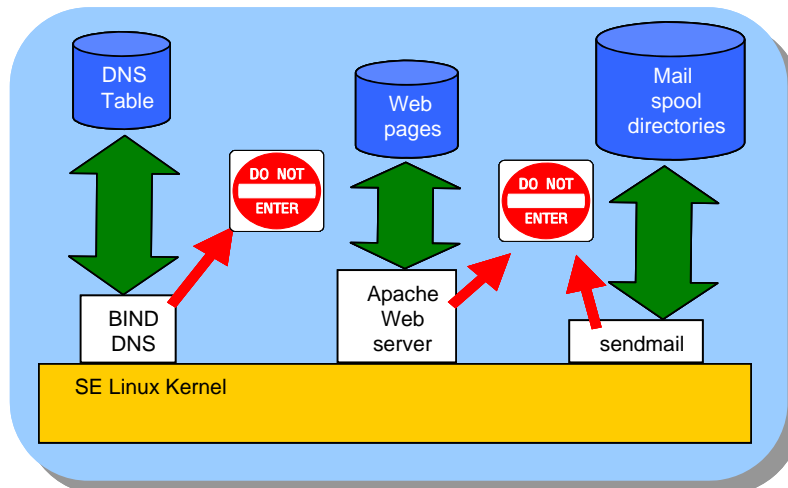
SELinux comes with an example security server that implements a combination of Role-Based Access Control (RBAC), \ Type Enforcement (TE), and, optionally, a Multi-Level Security (MLS) policy.

The example security server implements security contexts that consist of user identity, role, domain or type, and, optionally, MLS range or level. The RBAC and TE policies can be configured to meet

many security requirements. An example configuration is provided that shows how to configure these policies to meet particular security objectives and can be used as a starting point for creating custom policy configurations. Most of the security logic is expressed through the TE configuration, which assigns domains to processes and types to objects, and defines permissions between pairs of domains and between domains and types. Domain transitions are defined based on the calling domain and the type of program, so fine-grained distinctions can be provided among processes on the system. Roles are used to conveniently group sets of domains for assignment to users.

The rest of the SELinux code consists of:

- An access vector cache (AVC) provides caching of access decision computations obtained from the security server to minimize the performance overhead of the SELinux security mechanisms.
- Persistent label mapping provides a mechanism for maintaining security contexts with persistent objects such as files and file systems.
- New system calls allow for the development of security-aware applications.
- SELinux implementations of LSM hook functions manage the security information associated with processes and objects and perform the SELinux access controls for each kernel operation. These hook functions call the security server and AVC to obtain and apply the security decisions.



For more information call us at 443-430-8000, send an e-mail to [ISSO-research@sparta.com](mailto:ISSO-research@sparta.com), or visit us on the Web at <http://www.iss0.sparta.com/research>.