2008 International Conference on Security Technology

Analysis and Study of Security Mechanisms inside Linux Kernel

Gaoshou Zhai, Yaodong Li
School of Computer and Information Technology, Beijing Jiaotong University
gszhai@bjtu.edu.cn

Abstract

It's very important to analyze and study security mechanisms provided by an operating system. Currently, Linux is becoming one of the most popular operating systems because of its excellent performance and open source philosophy. Since lots of individuals and enterprises are switching to Linux, access control mechanism of Linux has been improved from time to time for new security requirements. For instance, SELinux sub-system can enforce a policy based Mandatory Access Control (MAC) and provide flexible security policy configuration. However, there are still some defects in current Linux access control mechanism. In this paper, available Linux security mechanisms are analyzed at first, while permission division principle is summarized. Then a new MAC mechanism is devised based on some popular information security models such as RBAC, DTE and etc and it is characteristic of cross-layered permission assignment. Finally, a corresponding prototype system is implemented and further research directions are summarized.

1. Introduction

Human society is going through a digitized era because of the explosion of information technology. A great lot of information has been digitized and maintained centrally by information systems. More and more valuable and critical information are being managed by information systems. Hence, the security capability of an information system is now becoming bottleneck of an information system. On the other hand, security of an operating system is the precondition to guarantee security of an information system because operating system is the base and the kernel of an information system. Therefore, it is very important and necessary to study security mechanisms provided by an operating system [1]. The foremost target of information security is to assure data security which can be protected by enforcing access control. Currently, Linux is becoming one of the most popular operating systems because of its excellent performance and open source philosophy. Since lots of individuals and enterprises are switching to Linux, access control mechanism of Linux has been improved from time to time for new security requirements (refer to figure 1).

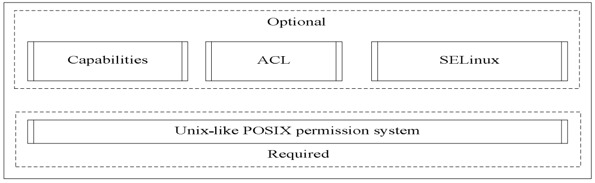

Figure 1. **Security Mechanisms of Linux**

For instance, SELinux sub-system can enforce a policy based Mandatory Access Control (MAC) and provide flexible security policy configuration. However, there are still some defects in current Linux access control mechanism. The main target of this paper is to analyze and improve access control mechanism of current Linux operating system. To achieve this goal, available Linux security mechanisms ought to be analyzed at first.

2. Discretionary access control mechanism of Linux

The foremost security mechanism inside Linux kernel is Unix-like POSIX permission system, and then capabilities and access control lists (ACL) permission are later introduced as optional components.

2.1. Unix-like POSIX permissions system

Unlike a Microsoft ACE/DACL control system, Linux implements a Unix POSIX permissions system [2]. In this system, every file and directory can have a combination of read, write and execute privileges and these permissions are represented numerically with different binary bits. At the same time, these permissions are organized and be set according to ownership, group membership and others.

978-0-7695-3486-2/08 $25.00 © 2008 IEEE
DOI 10.1109/SecTech.2008.17

Although it is simple and convenient to configure the system permissions and the corresponding cost is saving. But its granule for permission control is too big to satisfy permission configuration requirement with fine granules. For example, it can't be used to define per-user or per-group permissions. Meanwhile, controlled objects are limited to files and the other potential objects such as processes are ignored. Furthermore, security identification information of a file can be modified dynamically during the system running period. In addition, the existence of super-user or administrator is the hidden trouble of system security.

2.2. Capabilities

Capabilities [3-5] are introduced in order to solve super-user problems. The essential principle of capabilities is to add a 32-bit permission map into process control block so as to control the corresponding process under given permissions. Because each bit is corresponding to a kind of permission, capabilities can describe at most 32 different permissions.

Capabilities have refined system permissions and can effectively control super-user under the precondition of proper configuration. And they can implement process-level permission control with compact but effective descriptions. However, they focus on subjects and objects are ignored and impaired. Moreover, they can't improve confidentiality of system data effectively and there is still a long way to reach the target of flexible security policy configuration.

2.3. ACL

To configure the system permissions with well-fined granules, ACL [6] can be employed. Furthermore, each file is equipped with an access control list as an extended attribute to describe all permissions for different subjects as to it. Thus any given permissions can be assigned for any special user as to any file.

ACL improves flexibility of permission configuration with refined granules (subjects). But operation granules for permission are still rough and it requires excessive storage space to save ACL. Nevertheless, it can't control process-level permissions effectively.

3. Mandatory access control mechanism of Linux (SELinux)

The up-to-date Linux kernel has been integrated with SELinux[7] which enforces a MAC policy and its security performance is improved to a great extent.

3.1. Architecture and principle of SELinux

SELinux employs role-based access control (RBAC), type enforcement (TE) and a optional multi-level security (MLS) synthetically (refer to figure 2).

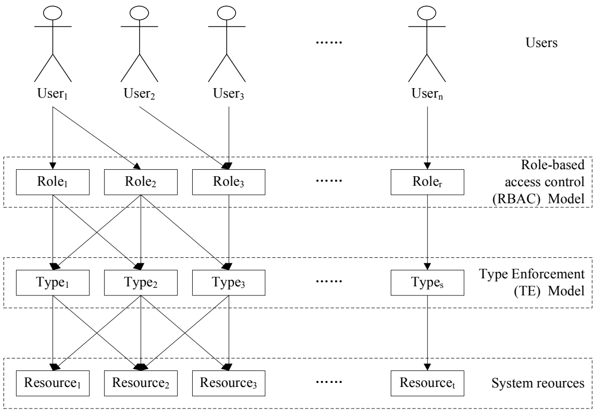

Figure 2. **Access control model of SELinux**

The architecture of SELinux can be referred to Figure 3.

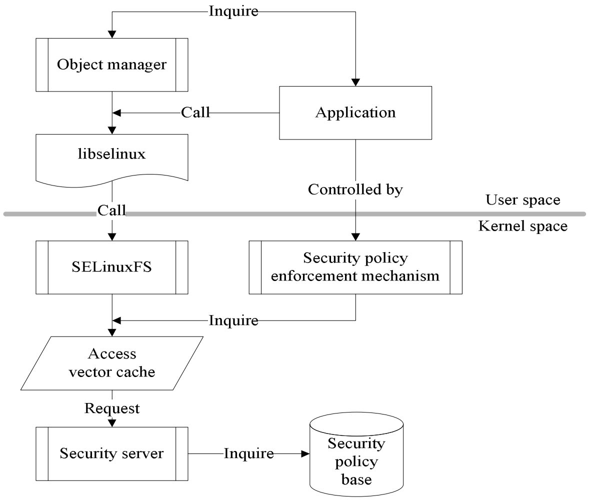

Figure 3. **Architecture of SELinux**

3.2. Policy configuration of SELinux

Security policy configuration for SELinux is made up of definitions for TE access vector rules, RBAC rules and constraints rules. All of these are described in security policy description language provided by SELinux.

3.3. Evaluation of SELinux

SELinux supports multiple security models and MAC policy. It is extensible and flexible with refined permission configuration. But it is the bottleneck of SELinux to configure the security policy of the system correctly. Moreover, flexibility of policy configuration is always weakened because there are close relationships among multiple models of SELinux. In addition, policy description and policy configuration of SELinux is too complicated to be mastered.

4. A new mandatory access control mechanism

It can be concluded that permission/privilege division is the essential way to solve systematic security problem. In another word, security performance of a system is dependent on the way of permission/privilege division.

4.1. LYSLinux access control model

A hybrid access control model named LYSLinux is devised by integrating RBAC and Domain Type Enhancement (DTE) together (refer to figure 4).

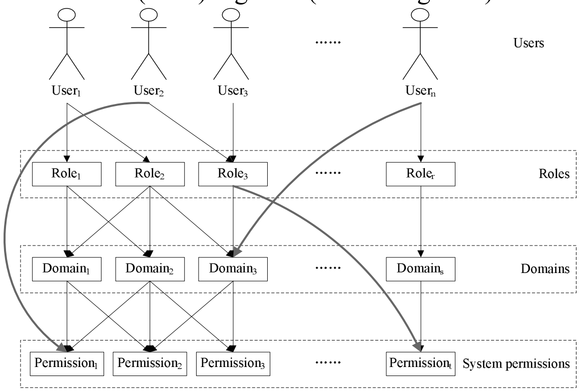

Figure 4. **Access control model of LYSLinux**

Note that although the model is a layered permission assignment model, permissions can be assigned crossing layers, as is more consistent with that in real world. Naturally, it can simplify the security policy configuration. In addition, FLASK architecture ought to be adopted so as to ensure configurability of systematic permission and separation between access control and security policy.

4.2. Description of security policy

The whole security model can be described by type table, permission table, domain table, role table, user table and relationship table (refer to figure 5). And the relationships among different tables are more like tree structure, which can be described in extensible mark language (XML).

4.3. Storage and management of security policy

All factors such as interaction modes between user space and kernel space, communication efficiency between MAC module and security server, and etc. are considered. Therefore, security server is running in kernel space and a special data structure is build up between security server and security policy base (in XML files) to store and describe security policy. When security module is start-up, security policy is read from file system and the data structure is initialized. And in order to be convenient for security server to build up security policy data structure, corresponding binary security policy description file format ought to be designed and the transformation from XML descriptive file to the binary file can be executed in user space.

5. Prototype Implementation

The architecture of the prototype can be referred to figure 6. It is made up of security enforcer, object manager, AVC, security server and a kernel virtual char device. All components of access control modules are running in kernel space and can be implemented in a loadable kernel module.

In addition, the compiler of security policy is programmed in Java.

6. Summary

The prototype is tested and related results show that the MAC mechanism in this paper is functional and feasible. Nevertheless, some aspects ought to be improved in future. For example, security model can be improved to support role ranking, access control for resources other than files (such as special files, network operation) and even other file systems (not ext3) can be supported and security policy configuration tools can be develop to simplify the configuration work.

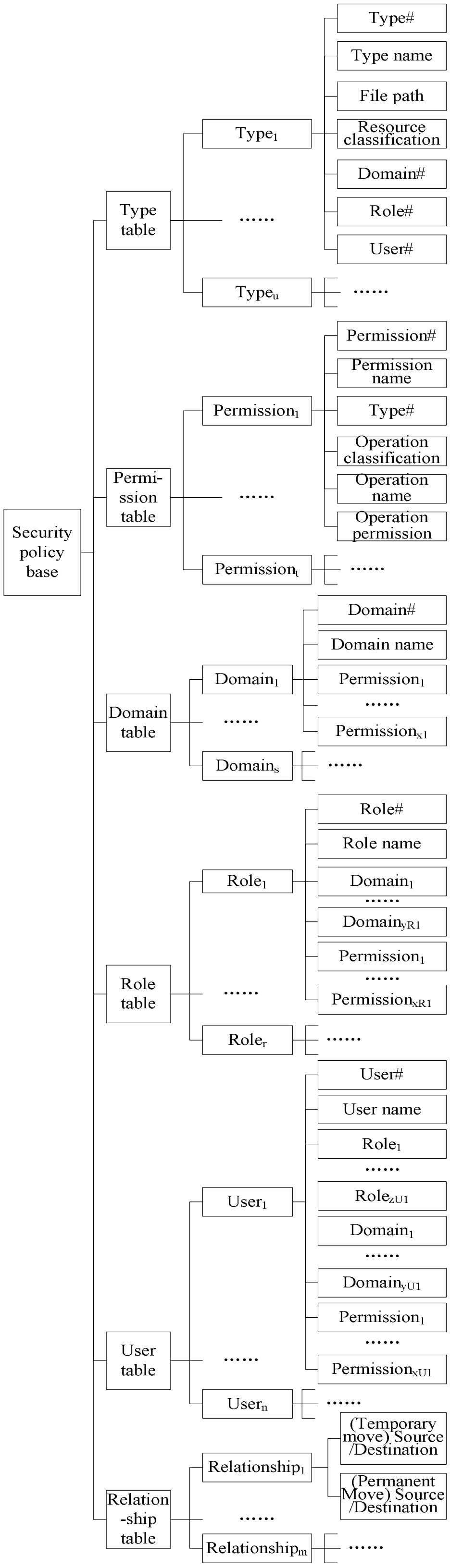

Figure 5. **Security policy of LYSLinux**

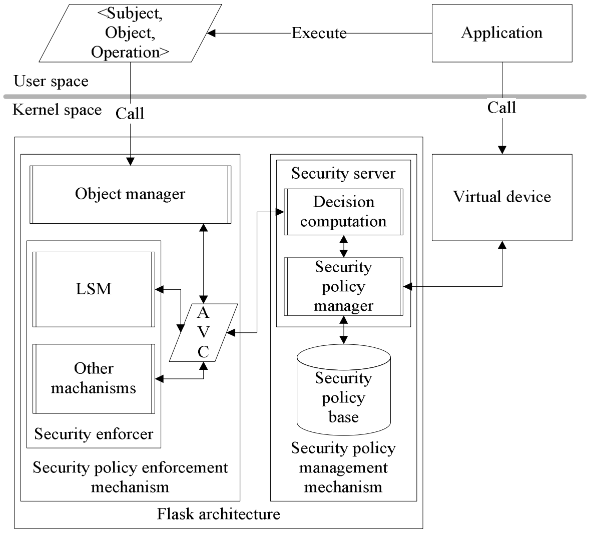

Figure 6. **Architecture of the prototype**

7. Acknowledgements

The research presented in this paper was performed with the support of Beijing Jiaotong University Grants for 2005SM016.

8. References

[1] Gaoshou Zhai, Jie Zeng, Miaoxia Ma, Liang Zhang, "Implementation and Automatic Testing for Security Enhancement of Linux Based on Least Privilege", *IEEE CS Proceedings of The 2nd International Conference on Information Security and Assurance (ISA 2008)*, Busan, Korea, April 24 - 26, 2008, pp. 181-186.

[2]http://www.networkingprogramming.com/1024x768/linux1.html

[3] Wenjie Tu, Haibing Guan, Yingcai Bai, "Enhancing Access Control Function in Linux File System", Computer Applications and Software, vol.23, no. 2, 2006, pp. 117-118, 119. (in Chinese)

[4] Michael Bacarella, "Taking advantage of Linux capabilities", *Linux Journal*, Specialized Systems Consultants, Inc. , vol.2002, no.97, 2002.

[5] Charistias P.J. Unix Online Man Pages: Capability(4). Http://www.mcsr.olemiss.edu/cgi-bin/man-cgi?Capabilities, 1994

[6] http://www.suse.de/~agruen/acl/chapter/fs_acl-en. pdf

[7] Stephen Smalley, Chris Vance, Wayne Salamon, "Implementing SELinux as a Linux Security Module", *NAI Labs Report #01-043*. Feb 2006.

[8] Zanin, Giorgio, "Towards a formal model for security policies specification and validation in the SELinux system", *Proceedings on the Ninth ACM Symposium on Access Control Models and Technologies*, SACMAT 2004, 2004, pp. 136-145.