Rule Set Based Access Control (RSBAC)

Linux Kernel Security Extension

Short Overview for OpenWeekend 2002 in Prague

Amon Ott <ao@rsbac.org>

Contents:

1 Introduction
1.1 History
1.2 Motivation
1.3 Design Goals
1.4 Overview of RSBAC

2 Architecture and Implementation of the Framework
2.1 Subjects, Objects and Requests
2.2 List of Requests with Targets
2.3 Architectural Diagram
2.4 Module Registration (REG)

Contents II:

3 Implemented Models
3.1 MAC, FC and SIM
3.2 PM, MS and FF
3.3 AUTH
3.4 RC
3.5 ACL
3.6 CAP

4 Practical Experience
4.1 Running Systems
4.2 Stability
4.3 Performance

Contents IV:

5 Online Ressources

6 New in 1.2.0

***

1 Introduction

1.1 History

1.2 Motivation

1.3 Design Goals

1.4 Overview of RSBAC

1.1 Introduction: History

- RSBAC Project started as Master Thesis in November 1996
- First public RSBAC version 0.9 for Linux kernel 2.0.30 on January, 9, 1998
- Current stable release 1.2.0 for kernels 2.2.20 and 2.4.18
- 1.2.0 with many changes against 1.1.2

1.2+3 Introduction: Motivation and Goals

- Classic Linux/Unix Access Control is insecure
  - Small Granularity
  - Discrete Control
    - Trusted user?
    - Malware: Invitation to Trojans and Viruses
  - Superuser root
    - Full Access
    - Too often needed
    - Too many exploits (root kits, kernel module attacks etc.)
- Better models for other protection goals
- Flexible Model selection and combination
- Good portability

1.4 Introduction: Overview

- Based on GFAC by Abrams and LaPadula
- Open Source with GPL
- Flexible structure
  - Separation between enforcement (AEF), decision (ADF) and access control information (ACI)
  - Only AEF and part of ACI system dependent
  - Almost any type of model supportable
  - Model independent -> meta policy
  - Runtime Module Registration (REG)

1.4 Introduction: Overview II

- Powerful logging system
  - Request and decision based
  - User based
  - Program based
  - Object based
- Stable production use since March 2000
- Support for current Linux kernels, ports to other systems likely
- Two Linux distributions with RSBAC: ALTLinux Castle and Kaladix

2 Architecture and Implementation of the Framework

2.1 Subjects, Objects and Requests

2.2 List of Requests with Targets

2.3 Architectural Diagram

2.4 Module Registration (REG)

2.1 Architecture: Subjects, Objects and Requests

- Subjects:
  - Processes acting on behalf of users
- Object types (target types):
  - FILE
  - DIR
  - FIFO
  - SYMLINK
  - DEV (devices by block/char and major:minor)
  - IPC (Inter Process Communication)
  - SCD (System Control Data)
  - USER
  - PROCESS
  - NETDEV (new in 1.2.0: Network Devices)
  - NETTEMP (new in 1.2.0: Network Object Templates)
  - NETOBJ (new in 1.2.0: Network Objects (Sockets etc.))

2.1 Architecture: Subjects, Objects and Requests II

- Requests:
  - Abstraction of what a subject wants to do with an object

2.2 Architecture: List of Requests with Targets

R_ADD_TO_KERNEL: NONE
R_ALTER: IPC
R_APPEND_OPEN: FILE, FIFO, DEV, IPC
R_CHANGE_GROUP: FILE, DIR, FIFO, IPC, USER, PROCESS, NONE
R_CHANGE_OWNER: FILE, DIR, FIFO, IPC, PROCESS, NONE
R_CHDIR: DIR
R_CLONE: PROCESS
R_CLOSE: FILE, DIR, FIFO, DEV, IPC, NETOBJ
R_CREATE: DIR (where), IPC, NETTEMP, NETOBJ
R_DELETE: FILE, DIR, FIFO, IPC, NETTEMP
R_EXECUTE: FILE
R_GET_PERMISSIONS_DATA: FILE, DIR, FIFO, IPC, SCD
R_GET_STATUS_DATA: FILE, DIR, FIFO, SYMLINK, IPC, SCD, NETDEV
R_LINK_HARD: FILE, FIFO
R_MODIFY_ACCESS_DATA: FILE, DIR, FIFO
R_MODIFY_ATTRIBUTE: All target types
R_MODIFY_PERMISSIONS_DATA: FILE, DIR, FIFO, IPC, SCD, NONE
R_MODIFY_SYSTEM_DATA: SCD, NETDEV

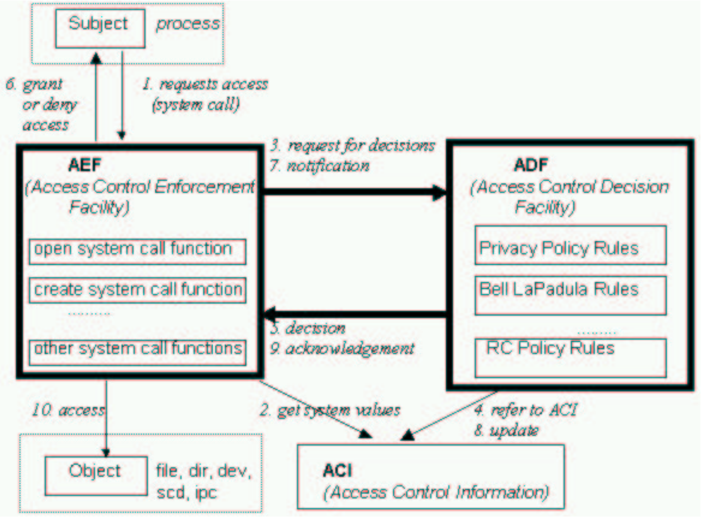
2.2 Architecture: List of Requests with Targets II

R_MOUNT: DIR, DEV
R_READ: DIR, SYMLINK, IPC, NETTEMP (optional: FILE, FIFO, DEV, NETOBJ)
R_READ_ATTRIBUTE: All target types
R_READ_OPEN: FILE, FIFO, DEV, IPC
R_READ_WRITE_OPEN: FILE, FIFO, DEV, IPC
R_REMOVE_FROM_KERNEL: NONE
R_RENAME: FILE, DIR, FIFO
R_SEARCH: DIR, FIFO
R_SEND_SIGNAL: PROCESS
R_SHUTDOWN: NETOBJ, NONE
R_SWITCH_LOG: NONE
R_SWITCH_MODULE: NONE
R_TERMINATE: PROCESS (notify only)
R_TRACE: PROCESS
R_TRUNCATE: FILE
R_UMOUNT: DIR, DEV, NONE
R_WRITE: DIR, SCD, NETTEMP (optional: FILE, FIFO, DEV, NETOBJ)
R_WRITE_OPEN: FILE, FIFO, DEV, IPC

2.2 Architecture: List of Requests with Targets III

(New in 1.2.0)
R_MAP_EXEC: FILE, NONE
R_BIND: NETOBJ
R_CONNECT: NETOBJ
R_LISTEN: NETOBJ
R_ACCEPT: NETOBJ
R_SEND: NETOBJ
R_RECEIVE: NETOBJ

2.3 Architectural Diagram



2.4 Module Registration (REG)

- Runtime registration of decision functions (Rule Sets) and system calls
- Model implementation e.g. as kernel module
- Add or remove models, syscalls or generic (persistent) lists in a running system
- Easy control of module removal by the module itself
- Sample modules provided

3 Implemented Models

3.1 MAC, FC and SIM

3.2 PM, MS and FF

3.3 AUTH

3.4 RC

3.5 ACL

3.6 CAP

3.1 Models: MAC, FC and SIM

- Mandatory Access Control (MAC):
  - Bell-LaPadula
  - 253 security levels
  - 64 categories
  - Automatic adjustment of current_sec_level and current_categories via mac_auto with boundaries
- Functional Control (FC):
  - Simple role model
  - User, Security Officer, System Administrator
  - Object Categories: General, Security, System
- Security Information Modification (SIM)
  - Even simpler role model
  - User and Security Officer
  - Object Types: None, Security Information

3.2 Models: PM, MS and FF

- Privacy Model by Simone Fischer-Hübner (PM):
  - Complex model conforming to EU privacy laws
  - Object Classes, Purposes, Tasks, Necessary Accesses, ...
- Malware Scan (MS):
  - On-Access Malware Scanner
  - File and socket accesses
  - Scan status: unscanned, rejected, accepted-with-level
  - Prototype - only few viruses detected
  - Plug-In interface for better scanning engines
- File Flags (FF):
  - Inheritable FILE, DIR, FIFO and SYMLINK attributes
  - e.g. read-only, no-execute, secure-delete

3.3 Models: AUTH

- Authentication (AUTH):
  - Restriction of CHANGE_OWNER with target PROCESS (setuid)
  - CHANGE_OWNER capabilities (inherited from file to process)
  - auth_may_setuid and auth_may_set_cap
  - Daemon based authentication enforcable

3.4 Models: RC

- Role Compatibility (RC):
  - Unlimited roles and types, types grouped per target type (file, dir, fifo, symlink together)
  - Compatibility of roles
    - with object types
    - with other roles (change role)
    - in request granularity
  - Forced and Initial Roles based on program files
  - Separation of Administration Duties
    - Separate sets of roles
    - Admin Roles
    - Assign Roles
    - Additional access rights: Admin, Assign, Access Control, Supervisor

3.5 Models: ACL

- Access Control Lists (ACL)
  - What subject may access which object with which requests
  - Subjects:
    - RC roles (!)
    - Users
    - ACL Groups
  - ACL Groups:
    - All users can have individual groups
    - Private and global groups
  - Inheritance with masks (similar to Netware 3.xx)
  - Default ACLs on top of hierarchy
  - Special Rights:
    - Access Control
    - Forward
    - Supervisor

3.6 Models: CAP

- Linux Capabilities:
  - Minimum and maximum capability sets for users and programs
  - Applied at CHANGE_OWNER on processes (setuid) and EXECUTE
  - Precedence of Minimum over Maximum Sets
  - Precedence of Program over User Sets
  - Limit rights of root programs or extend rights of normal user programs
  - E.g. run sendmail from normal user account with DAC_OVERRIDE and NET_BIND_SERVICE

4 Practical Experience

4.1 Running Systems

4.2 Stability

4.3 Performance

4.1 Practical Experience: Running Systems

- Compuniverse Firewall Servers
  - Since 2000 with RSBAC (optional in the beginning)
  - Strict encapsulation with full usability is possible
  - Use of AUTH, FF and RC models
  - Software selection for better RSBAC control, e.g. POP3 with separate authentication program
- Many systems by other admins (see RSBAC mailing list)
- Linux distributions ALTLinux Castle and Kaladix

4.2 Practical Experience: Stability

- UP: Very high stability
  - no crash yet on my and customer production systems
  - no crashes for 1.1.2 reported
  - 1.2.0 just released
- SMP: High stability
  - only few problems reported
  - no outstanding problems for 1.2.0 from pre series

4.3 Practical Experience: Performance

- Performance influences
  - Number and dynamic change of attribute objects
  - Number and type of decision modules
  - Logging
- Benchmarks
  - Celeron 333 system, 2.4.18 kernel, RSBAC 1.2.0-pre6
  - Three compile runs of same Linux kernel source each
  - Runtime with framework only (Maint Mode): +0.51% (kernel +7.70%)
  - Runtime with RC, AUTH, network control: +1.77% (kernel +25.22%)
  - Runtime with REG, FF, RC, AUTH, ACL, CAP, network control (def. config): +4.52% (kernel +88.37%)

5 Online Ressources

□ RSBAC Homepage: http://www.rsbac.org

□ Mailing List

  ○ Requests: rsbac-request@rsbac.org
  ○ Mails: rsbac@rsbac.org
  ○ Archive available (see contact page)

6 New in 1.2.0

□ User ID and RC role based symlink redirection support

□ Network Device (NETDEV) targets (for configuration and raw access)

□ Real template based network access control

  ○ Network Object (Socket) templates (NETTEMP) and targets (NETOBJ)
  ○ New requests BIND, CONNECT, etc.

□ CAP module with min and max Linux Capabilities for users and programs

6 New in 1.2.0 II

□ Network and firewall config protection as new SCD targets

□ Unlimited roles and types in RC model

□ Separate request type MAP_EXEC for library mapping (used to be EXECUTE, too)

□ Lifetime limites for many RC and ACL settings, i.e. access rights

***

Rule Set Based Access Control (RSBAC)

Linux Kernel Security Extension

Amon Ott <ao@rsbac.org>

Thank you!