Rule Set Based Access Control (RSBAC)

Amon Ott <ao@rsbac.org>

Contents:

Contents II:

Contents III:

Contents IV:

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 1.1.2 for kernels 2.2.19 and 2.4.8-9
- 1.2.0-pre1 released
- 1.2.0 with many changes (see Outlook)

1.2+3 Introduction: Motivation and Goals

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

1.4 Introduction: Overview

- Based on GFAC by Abrams and LaPadula
- Several publications (see Homepage)
- 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 others systems likely
- Downloads and feedback constantly increasing
- 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
- 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

R_CREATE: DIR (where), IPC

R_DELETE: FILE, DIR, FIFO, IPC

R_EXECUTE: FILE

R_GET_PERMISSIONS_DATA: FILE, DIR, FIFO, IPC, SCD

R_GET_STATUS_DATA: FILE, DIR, FIFO, SYMLINK, IPC, SCD

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

2.3 Architecture: List of Requests with Targets II

R_MOUNT: DIR, DEV

R_READ: DIR, SYMLINK, IPC (optional: FILE, FIFO, DEV, IPC-sock)

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: NONE

R_SWITCH_LOG: NONE

R_SWITCH_MODULE: NONE

R_TERMINATE: PROCESS

R_TRACE: PROCESS

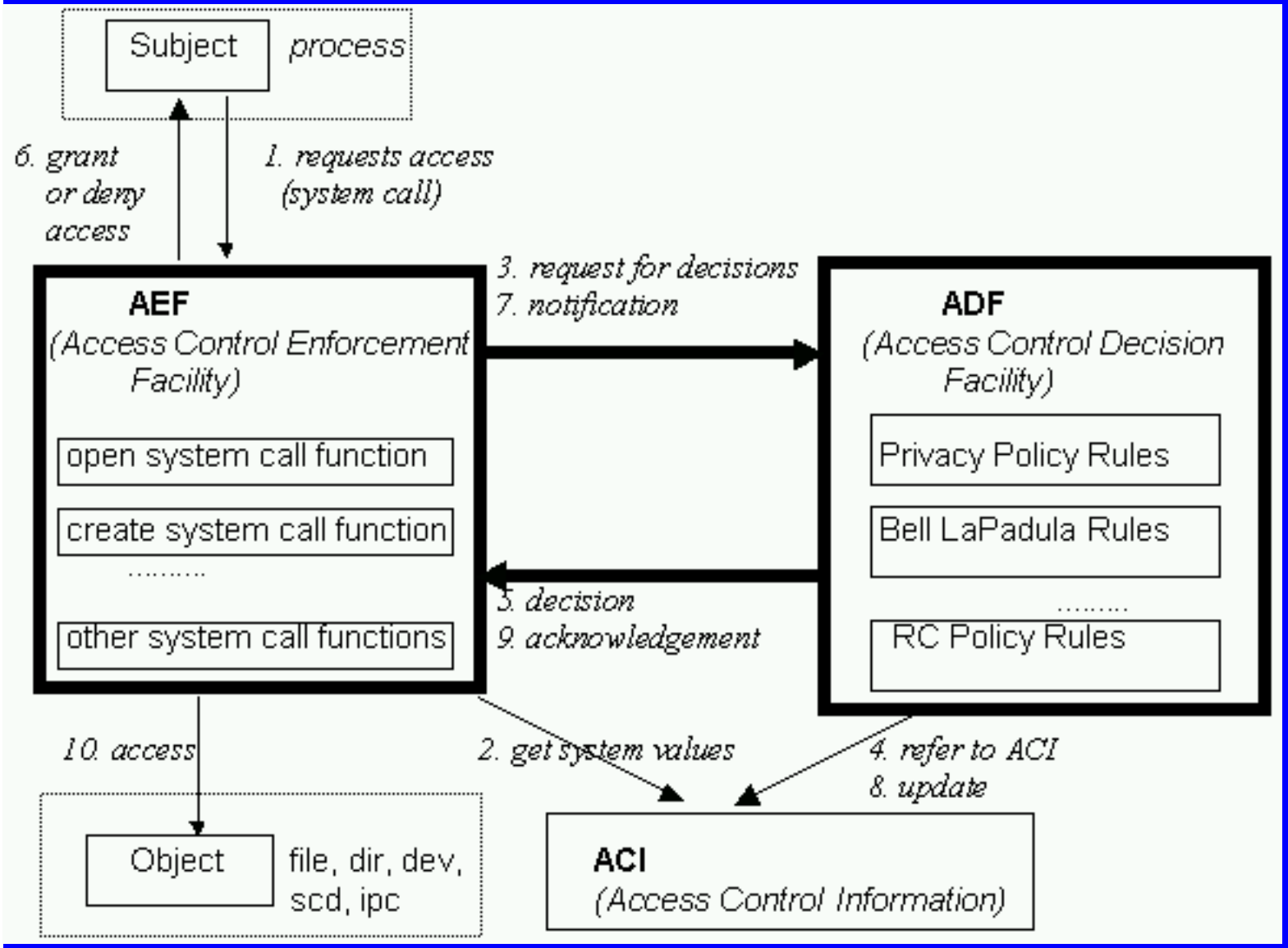
R_TRUNCATE: FILE

R_UMOUNT: DIR, DEV, NONE

R_WRITE: DIR, SCD (optional: FILE, FIFO, DEV, IPC-sock)

R_WRITE_OPEN: FILE, FIFO, DEV, IPC

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

3.1 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
- 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):
  - 64 roles and 64 types per target type (file, dir, fifo, symlink grouped)
  - Compatibility of roles
    - with object types (64 per target type!)
    - 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

3.4 Models IV: 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

4 Installation under Linux

4.1 Linux Kernel
4.2 Administration tools
4.3 First Boot

4 Installation under Linux

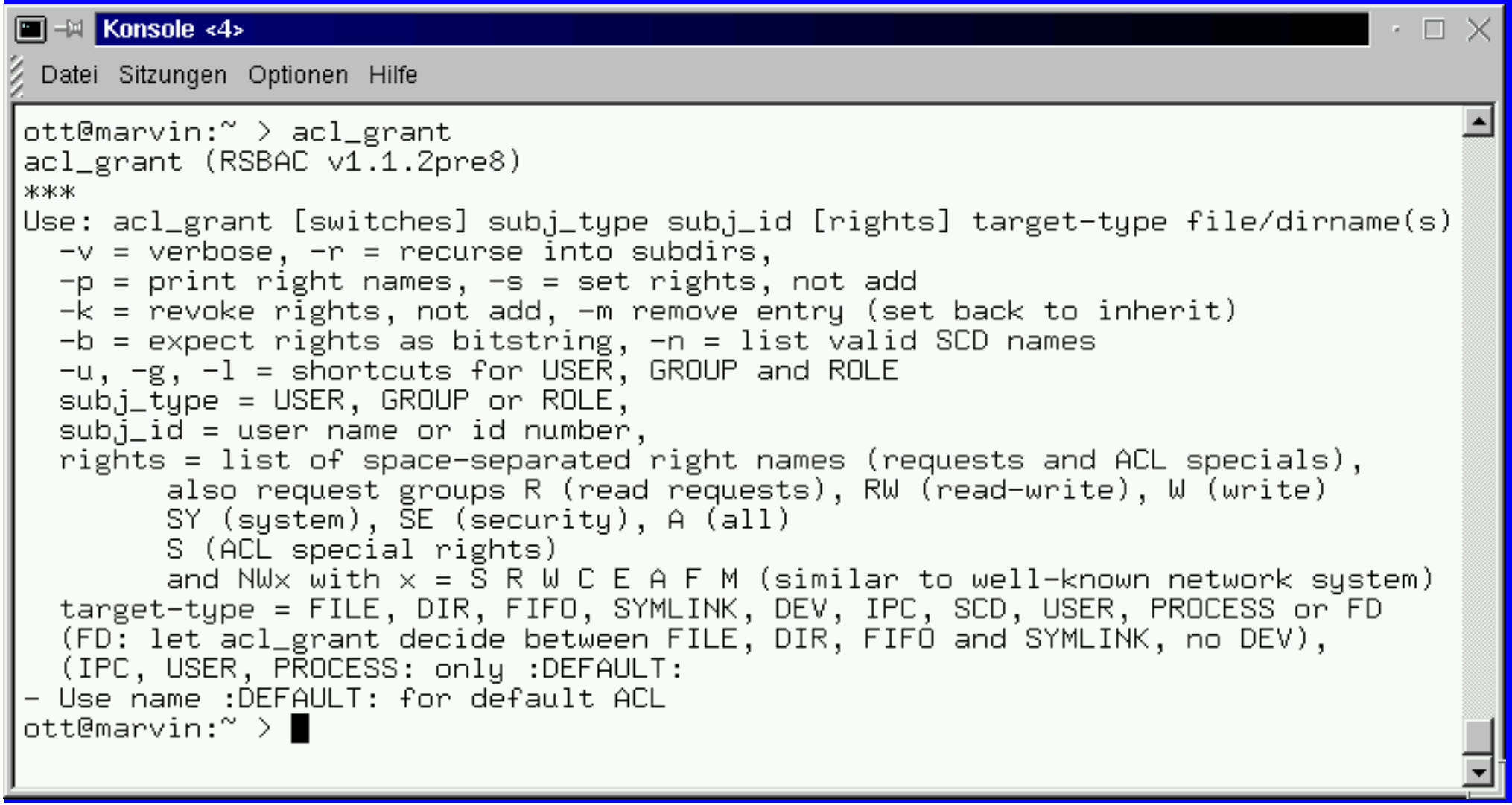
- Linux Kernel
  - Extract tar archive in kernel dir
  - Patch kernel (with patch-x.y.z.gz)
  - Configure, touch Makefile, compile and install
  - RSBAC normal and maint kernels / Soft Mode
- Administration tools
  - Extract tar archive
  - ./configure && make && make install
- First Boot
  - Kernel parameter rsbac_auth_enable_login
  - Add user 400 (Security Officer etc.)
  - Adjust AUTH capabilities for failed services

5 Administration

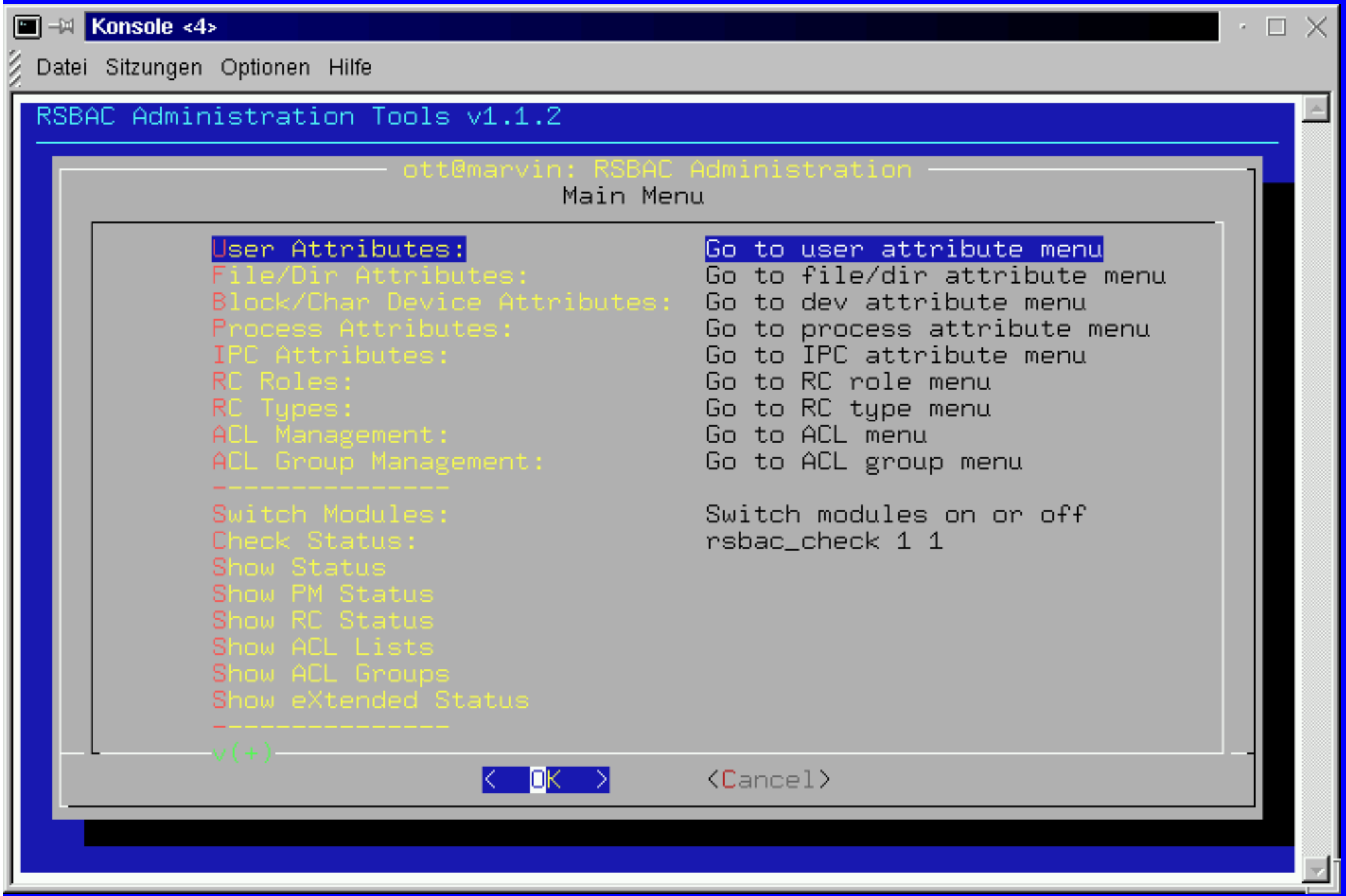
5.1 Attributes
5.2 Command Line Tools
5.3 Menues

5.1+2 Administration: Attributes and Command Line Tools

- General and Model specific (PM, RC, AUTH, ACL)



5.3 Administration: Menues



6 Areas of use

6.1 Workstations
6.2 Server systems

6.1 Areas of use: Workstations

- Protection against unwanted configuration changes
- Malware protection
- Reduced administration work

6.2 Areas of use: Server Systems

- Encapsulation of services
- Need-to-Know principle
- Malware protection

- Firewalls
  - DNS, Proxies, etc.
  - Advanced Protection of base system

- (Virtual) Webservers
  - Apache, Zope etc.
  - Separation of domains
  - Protection of critical data
  - Encapsulation of CGIs

6.2 Areas of use: Server Systems II

- (Virtual) mail servers
  - sendmail, qmail, POP3, IMAP, Mailing Lists etc.
  - Separation of mail areas
- File servers
  - Samba, Coda, etc.
  - Separation of organizational areas
- Application servers
  - Separation between user accounts
  - Protection against user attacks
  - e.g. "Safer Surfing" Server
- Other servers

7 Practical Experience

7.1 Running Systems

7.2 Stability

7.3 Performance

7.1 Practical Experience: Running Systems

- Compuniverse Firewalls
  - More than one year 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 tests systems by other admins (see RSBAC mailing list)
- Linux distributions ALTLinux Castle and Kaladix

7.2 Practical Experience: Stability

- Over one year of very high stability
- SMP systems more than 6 months of high stability
- Single mount time lockups supposed to be solved in 1.1.2

7.3 Practical Experience: Performance

- Performance influences
  - Number and dynamic change of attribute objects
  - Number and type of decision modules
  - Logging

- Benchmark
  - Pentium system, 2.2.18 kernel, RSBAC 1.1.0
  - Three Linux kernel compile runs each
  - Runtime with framework only: +1.1%
  - Runtime with FF, RC, AUTH, ACL: +5.5% (kernel +142%)

  - Celeron 333 system, 2.4.6 kernel, RSBAC 1.1.2-pre8
  - Three Linux kernel compile runs each
  - Runtime with framework only: +1.05%
  - Runtime with REG, FF, RC, AUTH, ACL (def. config): +2.47% (kernel +51.48%)

8 Online Ressources

- RSBAC Homepage: http://www.rsbac.org
- Mailing List
  - majordomo@rsbac.org
  - http://www.compuniverse.de/lwgate/rsbac
- Linux-Kernels
  - ftp://ftp.<country code>.kernel.org

9 Demonstration

10 Outlook

- New Linux Capabilities (CAP) module
  - Minimum and maximum Linux cap sets for users and processes
- Real network access control
  - Socket templates and targets
  - New requests BIND, CONNECT, etc.
- Better user authentication
  - Kernel space user management?
  - RSBAC standard AUTH daemon?
  - Biometric authentication?
- PM overhaul with menues
- (?) Filesystem redirection support

Rule Set Based Access Control (RSBAC)

Amon Ott <ao@rsbac.org>

Thank you!