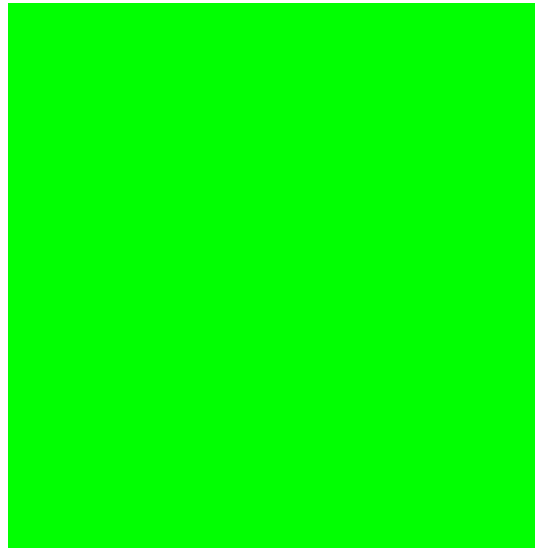


# Rule Set Based Access Control (RSBAC)

Freie Sicherheitserweiterung für den Linux-Kern



Amon Ott <[ao@rsbac.org](mailto:ao@rsbac.org)>

# Inhalt:

---

## 1 Einführung

### 1.1 Motivation

### 1.2 Überblick RSBAC

## 2 Aufbau des Rahmenwerks

### 2.1 Subjekte, Objekte und Entscheidungsanfragen

### 2.2 Architektur-Diagramm

# Inhalt II:

---

## 3 Implementierte Sicherheitsmodelle

3.1 AUTH

3.2 RC

3.3 ACL

3.4 FF

3.5 CAP

3.6 JAIL

3.7 RES

## 4 Installation unter Linux

4.1 Linux-Kern

4.2 Administrations-Programme

4.3 Der erste Start

# Inhalt III:

---

## 5 Administration

### 5.1 Attribute

### 5.2 Kommandozeilen-Programme

### 5.3 Menüs

## 6 Typische Serveranwendungen

## 7 Praktische Erfahrungen

### 7.1 Laufende Systeme

### 7.2 Stabilität

### 7.3 Performanz

# Inhalt III:

---

8 Weitere Informationen

9 Ausblick

# 1 Einführung

---

1.1 Motivation

1.2 Überblick RSBAC

# 1.1 Einführung: Motivation

---

- Klassische Zugriffskontrolle unter Linux/Unix ist unsicher
  - Geringe Granularität
  
  - Diskrete Kontrolle
    - Vertrauenswürdiger Benutzer?
    - Malware: Einladung für Trojaner und Viren
  
  - Superuser root
    - Voller Zugriff
    - Zu oft benötigt
    - Zu viele erfolgreiche Angriffe (root kits, kernel module attacks etc.)
  
- Bessere Modelle für andere Administrationsziele
- Flexible Modellauswahl und -kombination
  
- Gute Portierbarkeit

# 1.2 Einführung: Überblick

---

- Open Source mit GPL
- Flexible Struktur
  - Trennung zwischen Durchsetzung (AEF), Entscheidung (ADF) und Datenhaltung (ACI)
  - Nur AEF und Teil der Datenhaltung systemabhängig
  - Praktisch jede Art von Sicherheitsmodell implementierbar
  - Modellunabhängig durch eine Meta Policy
  - Runtime Module Registration (REG)
- Leistungsfähiges Logging-System
  - Default-Matrix: Anfragetyp, Entscheidung und Zieltyp
  - Individuell: Benutzer, Programm und Ziel-Objekt



# 1.4 Einführung: Überblick II

---

- Stabiler Produktionsbetrieb seit März 2000
- Unterstützt aktuelle Linuxkerne
- Downloads und Feedback wachsen stetig
- Zwei ältere Linux-Distributionen mit RSBAC: ALTLinux Castle und Kaladix
- Neue Trusted Debian-Linuxdistribution mit RSBAC

# 2 Aufbau des Rahmenwerks

---

2.1 Subjekte, Objekte und Entscheidungsanfragen

2.2 Architektur-Diagramm

# 2.1 Rahmenwerk: Subjekte, Objekte und Entscheidungsanfragen

---

## ■ Subjekte:

- Prozesse, die im Namen von Benutzern agieren

## ■ Objekttypen (Zieltypen, target types):

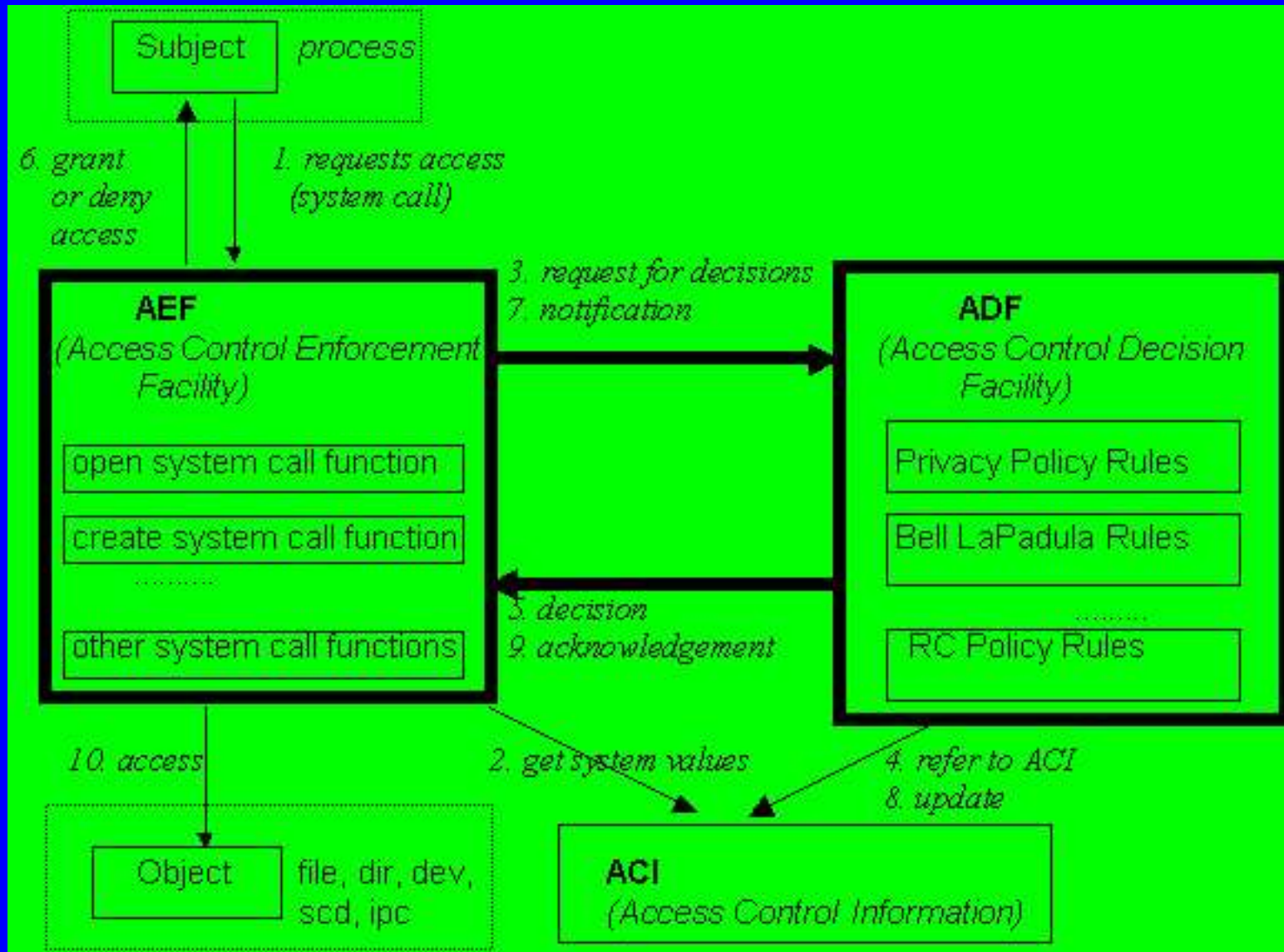
- FILE
- DIR
- FIFO
- SYMLINK
- DEV (Devices nach block/char und major:minor)
- IPC (Inter Process Communication = Prozeßkommunikation)
- SCD (System Control Data = systemweite Konfigurationsdaten)
- USER
- PROCESS
- NETDEV (Network Devices)
- NETTEMP (Network Object Templates)
- NETOBJ (Network Objects (Sockets etc.))

# 2.1 Rahmenwerk: Subjekte, Objekte und Entscheidungsanfragen

---

- Anfragetyp (request type):
  - Abstraktion dessen, wie ein Subjekt auf ein Objekt zugreifen moechte
- Entscheidungsanfrage:
  - Konkrete Anfrage an die Entscheidungskomponente

# 2.2 Architektur-Diagramm



# 3 Implementierte Sicherheitsmodelle

---

3.1 AUTH

3.2 RC

3.3 ACL

3.4 FF

3.5 CAP

3.6 JAIL

3.7 RES

# 3.1 Models: AUTH

## ■ Authentication (AUTH):

- Beschränkt CHANGE\_OWNER mit Zieltyp PROCESS (setuid)
- Optional: Beschränkung von CHANGE\_DAC\_{EFF|FS}\_OWNER (seteuid/setfsuid)
- Setuid capabilities (von der Programmdatei zum Prozeß vererbt):  
Mengen erreichbarer Benutzer-IDs
- auth\_may\_setuid und auth\_may\_set\_cap
- Kann Daemon-basierte Authentisierung erzwingen:
  - ▶ Prozeß authentisiert gegen Daemon
  - ▶ Daemons setzt capability für authentisierten Benutzer am Prozeß
  - ▶ Prozeß setzt Benutzer-ID
- Beschränkte Lebenszeit für alle Capability-Einstellungen

## 3.2 Models: RC

### ■ Role Compatibility (RC):

- Benutzer-Standard- und aktuelle Prozeß-Rollen
- Objekttypen (getrennt nach Zieltyp)
  
- Kompatibilität von Rollen mit Objekttypen nach Anfragetyp (Objektzugriffe)
- Kompatibilität von Rollen mit anderen Rollen (aktuelle Rolle wechseln)
  
- Erzwungene und Initial-Rollen für Programmdateien
  
- Trennung der Administrationsaufgaben
  - ▶ Admin Roles
  - ▶ Assign Roles
  - ▶ Zusätzliche Zugriffsrechte auf Typen: Admin, Assign, Access Control, Supervisor
  
- Beschränkte Lebenszeit für alle Kompatibilitätseinstellungen



# 3.3 Models: ACL

## ■ Access Control Lists (ACL)

- Welches Subjekt darf auf welches Objekt wie zugreifen
- Subjekte:
  - ▶ RC-Rollen (!)
  - ▶ Benutzer
  - ▶ ACL-Gruppen
- ACL-Gruppen:
  - ▶ Jeder Benutzer kann individuelle Gruppen verwalten
  - ▶ Private und globale Gruppen
- Vererbung der Rechte am übergeordneten Objekt, beschränkt durch Maske am Objekt
- Default-ACLs als oberster Vererbungsanker
- Administrationsrechte:
  - ▶ Access Control
  - ▶ Forward
  - ▶ Supervisor
- Beschränkte Lebenszeit für Gruppenmitgliedschaften und ACL-Einträge

# 3.5 Models: FF

---

## ■ File Flags (FF):

- Vererbare Attribute für Dateisystemobjekte (FILE, DIR, FIFO und SYMLINK)
- Z.B. read-only, no-execute, secure-delete, no-mount

# 3.5 Models: CAP

## ■ Linux Capabilities:

- Minimale und maximale Linux Capability Sets für Benutzer und Programme
- Anwendung beim CHANGE\_OWNER auf Prozesse (setuid) und EXECUTE
- Vorrang von Minimum vor Maximum
- Vorrang der Programmattribute vor den Benutzerattributen
- Rechte von root-Programmen beschränken oder normale Benutzer mächtiger machen
- Nur Verwaltung vorhandener Linux-Rechte

## 3.6 Models: JAIL

---

### ■ Process Jails:

- Prozesse in verstärkten chroot-Käfigen einsperren
- Vorkonfektionierte Kapselung von Serverprozessen
- Viele weitere Beschränkungen, einige optional
- Besonders Administrationsaufgaben und Netzwerknutzung stark eingeschränkt

# 3.7 Models: RES

## ■ Linux Resources:

- Minimale und maximale Ressourcen-Schranken für Benutzer und Programme
- Anwendung bei CHANGE\_OWNER auf Prozesse (setuid) und EXECUTE
  
- Vorrang von Minimum vor Maximum
- Vorrang der Programmattribute vor den Benutzerattributen
  
- Nur Verwaltung vorhandener Linux-Prozeß-Attribute:
- Maximale Dateigröße, Anzahl Prozesse, Hauptspeicher je Prozeß, ...

# 4 Installation unter Linux

---

4.1 Linux-Kern

4.2 Administrations-Programme

4.3 Der erste Start

# 4 Installation unter Linux

## ■ Linux-Kern

- Tar-Archiv im Kernquellenverzeichnis auspacken
- Kern patchen (mit patch-x.y.z.gz)
- Alternative: Download vorgepatchter Kernquellen
- Configure, touch Makefile, kompilieren und installieren
- Normaler oder Maintenance-RSBAC-Kern / Softmode

## ■ Administrationprogramme

- Tar-Archiv auspacken
- ./configure && make && make install

## ■ Der erste Start

- Kern-Parameter rsbac\_auth\_enable\_login
- Benutzer 400 anlegen (Security Officer etc.)
- AUTH capabilities für Daemons setzen

# 5 Administration

---

5.1 Attribute

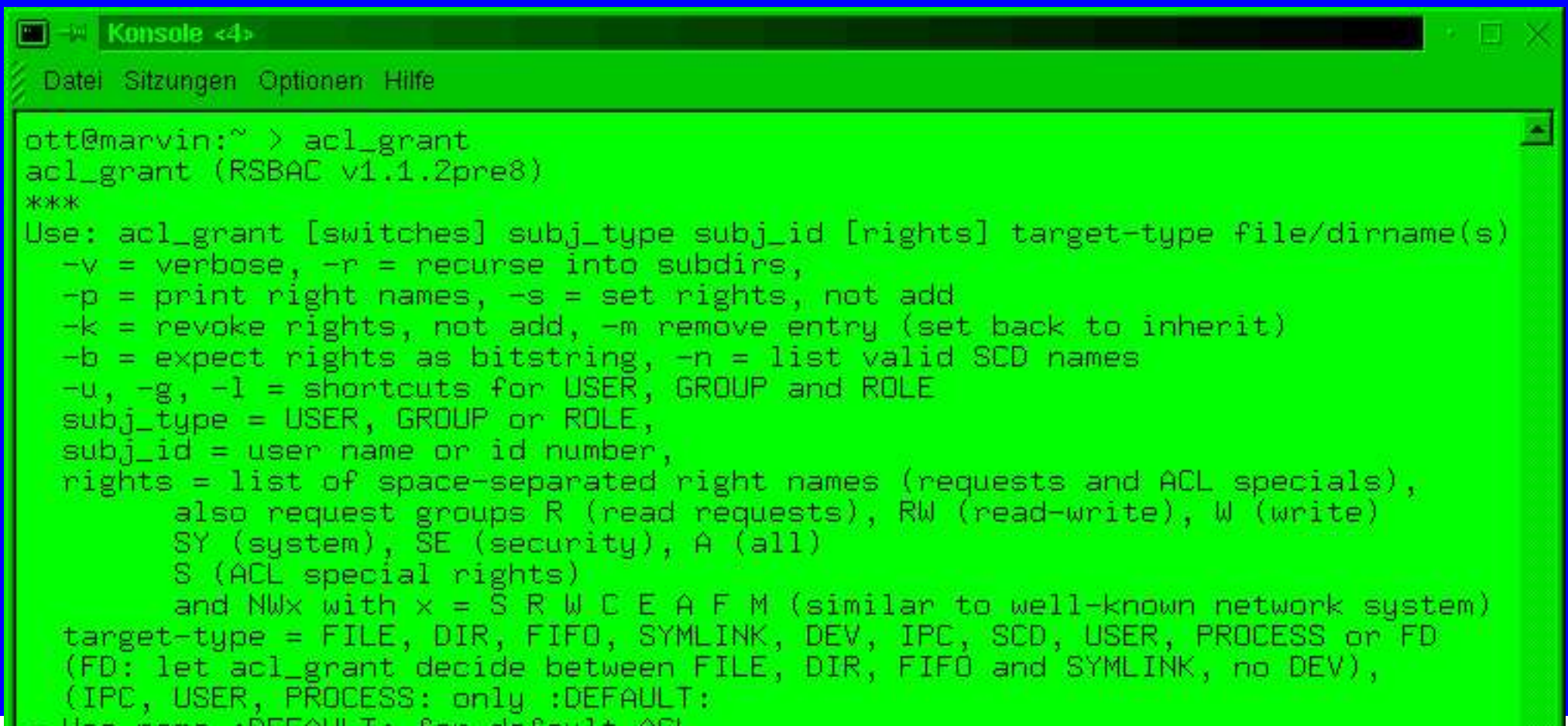
5.2 Kommandozeilen-Programme

5.3 Menüs



# 5.1+2 Administration: Attribute und Kommandozeilenprogramme

- Generelle und modell-spezifische Attribute (PM, RC, AUTH, ACL)



```
Konsole <4>
Datei Sitzungen Optionen Hilfe
ott@marvin:~ > acl_grant
acl_grant (RSBAC v1.1.2pre8)
***
Use: acl_grant [switches] subj_type subj_id [rights] target-type file/dirname(s)
-v = verbose, -r = recurse into subdirs,
-p = print right names, -s = set rights, not add
-k = revoke rights, not add, -m remove entry (set back to inherit)
-b = expect rights as bitstring, -n = list valid SCD names
-u, -g, -l = shortcuts for USER, GROUP and ROLE
subj_type = USER, GROUP or ROLE,
subj_id = user name or id number,
rights = list of space-separated right names (requests and ACL specials),
        also request groups R (read requests), RW (read-write), W (write)
        SY (system), SE (security), A (all)
        S (ACL special rights)
        and NWx with x = S R W C E A F M (similar to well-known network system)
target-type = FILE, DIR, FIFO, SYMLINK, DEV, IPC, SCD, USER, PROCESS or FD
(FD: let acl_grant decide between FILE, DIR, FIFO and SYMLINK, no DEV),
(IPC, USER, PROCESS: only :DEFAULT:
Use :zoo: :DEFAULT: for default :SEL:
```

# 5.3 Administration: Menüs

RSBAC Administration Tools v1.2.0

secoff@rsbac: RSBAC File/Dir/Fifo/Symlink Administration

Main FD Menu

FD List:	Choose from listing of last dir
FD Name:	/secoff/testdir / DIR
Attribute Get Mode:	real
-----	
MAC Security Level:	254 / inherit
MAC Categories:	(too long)
MAC Trusted for User:	4294967293 / NONE
FC Object Category:	3 / inherit
SIM Data Type:	2 / inherit
PM Object Class:	0
PM TP:	0
PM Object Type:	0 / None
MS Scanned:	0 / Unscanned
MS Trusted:	0 / Not trusted
MS Sock Trusted TCP:	0 / Not Trusted

↓(+)

< OK >

<Cancel>

< Help >

# 6 Typische Serveranwendungen

---

- Grundschatz des Basissystems
- Kapselung von Diensten
- Firewalls
  - DNS, Proxies, etc.
  - Besonderer Grundschatz wegen hoher Angriffswahrscheinlichkeit
- (Virtual) Webserver
  - Apache, Zope etc.
  - Trennung der virtuellen Domänen
  - Schutz kritischer Daten
  - Kapselung der CGI's

# 6 Typische Serveranwendungen II

---

## ■ (Virtuelle) Mailserver

- sendmail, qmail, POP3, IMAP, Mailing Lists etc.
- Trennung der Mailbereiche

## ■ Fileserver

- Samba, Coda, etc.
- Trennung der organisatorischen Einheiten

## ■ Applikationsserver

- Trennung der Benutzerbereiche
- Schutz gegen lokale Angriffe
- Schutz vor Netzwerkangriffen durch lokale Benutzer

## ■ Andere Server

# 7 Praktische Erfahrungen

---

7.1 Laufende Systeme

7.2 Stabilität

7.3 Performanz

# 7.1 Praktische Erfahrungen: Laufende Systeme

---

## ■ Compuniverse Firewalls

- Mehr als zwei Jahre mit RSBAC
- Strenge Kapselung mit voller Funktionalität ist möglich
- Benutzt AUTH, FF, RC und CAP-Modelle

## ■ Viele Test- und einige Produktionssysteme anderer Administratoren

## ■ Linux-Distributionen mit RSBAC:

- ALTLinux Castle
- Kaladix
- Neu: Trusted Debian

## 7.2 Praktische Erfahrungen: Stabilität

---

- Drei Jahre sehr hoher Stabilität
- SMP-Systeme mehr als zwei Jahre mit hoher Stabilität

# 7.3 Praktische Erfahrung: Performanz

- Einflußfaktoren für die Performanz
  - Anzahl und dynamisches Verhalten der Attributobjekte
  - Art und Anzahl der Entscheidungsmodule
  - Logging
  
- Benchmarks
  - Celeron 333 system, 2.4.19 kernel, RSBAC 1.2.1
  - Mittelwerte dreier Linux-Kern-Kompilierungsläufe
  - Laufzeit mit leerem Rahmenwerk: +0.68% (Kern +11.33%)
  - Laufzeit mit RC, AUTH, Netzwerk, alle Logging-Optionen: +2.30% (Kern +43.02%)
  - Laufzeit mit REG, FF, RC, AUTH, ACL, CAP, JAIL, Netzwerk, alle Logging-Optionen (def. config): +4.21% (Kern +82.47%)



# 8 Weitere Informationen

---

- RSBAC Homepage: <http://www.rsbac.org>
  
- Mailing List
  - Requests: [rsbac-request@rsbac.org](mailto:rsbac-request@rsbac.org)
  - Mails: [rsbac@rsbac.org](mailto:rsbac@rsbac.org)
  - Archiv verfügbar (siehe Contact)
  
- RSBAC-Artikel: iX 8/2002, Linux-Magazin Nr. 1 und 4 2003
  
- Trusted Debian: [www.trusteddebian.org](http://www.trusteddebian.org)

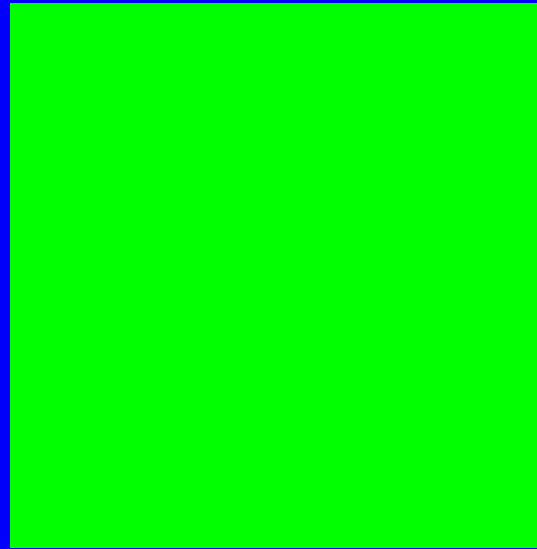
# 12 Ausblick

---

- Listenreplikation auf andere RSBAC-Systeme
- Später: Verteiltes RSBAC-System / RSBAC Cluster
- ???

# Rule Set Based Access Control (RSBAC)

Freie Sicherheitserweiterung für den Linux-Kern



Amon Ott <[ao@rsbac.org](mailto:ao@rsbac.org)>

Danke für Ihre Aufmerksamkeit!