

Mandatory Access Control Systems

Marcus Gelderie Martin Lang

Lehrstuhl für Informatik 7
RWTH Aachen University

16. Januar 2014

DAC

Discretionary Access Control

MAC

Mandatory Access Control

- ▶ “normale” Rechteverwaltung
- ▶ Zusätzlich zu DAC

DAC

Discretionary Access Control

- ▶ “normale” Rechteverwaltung
- ▶ Benutzer entscheiden über Rechte **ihrer** Objekte

MAC

Mandatory Access Control

- ▶ Zusätzlich zu DAC
- ▶ Administrator gibt Berechtigungen vor

DAC

Discretionary Access Control

- ▶ “normale” Rechteverwaltung
- ▶ Benutzer entscheiden über Rechte **ihrer** Objekte
- ▶ Alle Prozesse eines Benutzers gleichberechtigt

MAC

Mandatory Access Control

- ▶ Zusätzlich zu DAC
- ▶ Administrator gibt Berechtigungen vor
- ▶ Individuelle Regeln für unterschiedliche Prozesse
- ▶ feingranulare Rechte

DAC

Discretionary Access Control

- ▶ “normale” Rechteverwaltung
- ▶ Benutzer entscheiden über Rechte **ihrer** Objekte
- ▶ Alle Prozesse eines Benutzers gleichberechtigt
- ▶ Administrative Aufgaben laufen “als root” und dürfen alles

MAC

Mandatory Access Control

- ▶ Zusätzlich zu DAC
- ▶ Administrator gibt Berechtigungen vor
- ▶ Individuelle Regeln für unterschiedliche Prozesse
- ▶ feingranulare Rechte
- ▶ Administrative Prozesse können eingeschränkt werden (“root darf nicht mehr alles”)

1. Software darf meist mehr als sie braucht

- ▶ Missbrauchpotential durch schlecht programmierte/konfigurierte Software:
 - ▶ Webserver liefert private Daten aus, weil Pfade falsch konfiguriert sind.
 - ▶ PDF Viewer wird mit präpariertem Dokument “gehackt” und verschickt SPAM
Eigentlich kein Netzwerkzugriff notwendig

1. Software darf meist mehr als sie braucht

- ▶ Missbrauchpotential durch schlecht programmierte/konfigurierte Software:
 - ▶ Webserver liefert private Daten aus, weil Pfade falsch konfiguriert sind.
 - ▶ PDF Viewer wird mit präpariertem Dokument "gehackt" und verschickt SPAM
Eigentlich kein Netzwerkzugriff notwendig
- ▶ Nichtvertrauenswürdige/closed-source Software
 - ▶ z.B. Skype
(durchsucht jede Menge privater Daten)

1. Software darf meist mehr als sie braucht

- ▶ Missbrauchpotential durch schlecht programmierte/konfigurierte Software:
 - ▶ Webserver liefert private Daten aus, weil Pfade falsch konfiguriert sind.
 - ▶ PDF Viewer wird mit präpariertem Dokument “gehackt” und verschickt SPAM
Eigentlich kein Netzwerkzugriff notwendig
- ▶ Nichtvertrauenswürdige/closed-source Software
 - ▶ z.B. Skype
(durchsucht jede Menge privater Daten)

2. Benutzer treffen “schlechte” Entscheidungen

- ▶ Rechtemanagement kann zentral erfolgen
- ▶ Benutzer können nicht beliebig entscheiden
(weniger Fehler machen)

SELinux

geschrieben '98; im Kernel seit '03

AppArmor

geschrieben '98; in Ubuntu seit '07

SELinux

geschrieben '98; im Kernel seit '03

Drei "Modi":

targeted policy

strict policy

Multi Level Security
(MLS)



AppArmor

geschrieben '98; in Ubuntu seit '07

SELinux

geschrieben '98; im Kernel seit '03

Drei "Modi":



SEContext + Policy

user_u:role_r:type_t
(SEContext)

allow passwd_t shadow_t:file
rw_file_perms;
(Policy)

AppArmor

geschrieben '98; in Ubuntu seit '07

SELinux

geschrieben '98; im Kernel seit '03

Drei “Modi”:

targeted policy

strict policy

Multi Level Security
(MLS)



SEContext + Policy

user_u:role_r:type_t
(SEContext)

allow passwd_t shadow_t:file
rw_file_perms;
(Policy)

AppArmor

geschrieben '98; in Ubuntu seit '07

- Prozesse sind **confined** oder **unconfined**

SELinux

geschrieben '98; im Kernel seit '03

Drei “Modi”:

targeted policy

strict policy

Multi Level Security
(MLS)



AppArmor

geschrieben '98; in Ubuntu seit '07

- ▶ Prozesse sind **confined** oder **unconfined**
- ▶ Policy wird pro Programm geschrieben: Policy für apache, ssh, ...

SEContext + Policy

user_u:role_r:type_t
(SEContext)

allow passwd_t shadow_t:file
rw_file_perms;
(Policy)

SELinux

geschrieben '98; im Kernel seit '03

Drei “Modi”:

targeted policy

strict policy

Multi Level Security
(MLS)



SEContext + Policy

user_u:role_r:type_t
(SEContext)

allow passwd_t shadow_t:file
rw_file_perms;
(Policy)

AppArmor

geschrieben '98; in Ubuntu seit '07

- ▶ Prozesse sind **confined** oder **unconfined**
- ▶ Policy wird pro Programm geschrieben: Policy für apache, ssh, ...
- ▶ Kein “AppArmor Context”

SELinux

geschrieben '98; im Kernel seit '03

Drei “Modi”:



SEContext + Policy

`user_u:role_r:type_t`
(SEContext)

`allow passwd_t shadow_t:file`
`rw_file_perms;`
(Policy)

AppArmor

geschrieben '98; in Ubuntu seit '07

- ▶ Prozesse sind **confined** oder **unconfined**
- ▶ Policy wird pro Programm geschrieben: Policy für apache, ssh, ...
- ▶ Kein “AppArmor Context”
- ▶ Policy arbeitet auf Dateinamenbasis:
`allow /etc/shadow rw, ...`

- ▶ Windows:
 - ▶ Integrity Levels (IL)
 - ▶ Programm läuft unter bestimmtem Level n ; nur Zugriff auf Daten mit Level $\leq n$

- ▶ Windows:
 - ▶ Integrity Levels (IL)
 - ▶ Programm läuft unter bestimmtem Level n ; nur Zugriff auf Daten mit Level $\leq n$
- ▶ Android:
 - ▶ SEAndroid
 - ▶ Portierung von SELinux auf Android
 - ▶ derzeit: nur bestimmte Prozesse überwacht

- ▶ Windows:
 - ▶ Integrity Levels (IL)
 - ▶ Programm läuft unter bestimmtem Level n ; nur Zugriff auf Daten mit Level $\leq n$
- ▶ Android:
 - ▶ SEAndroid
 - ▶ Portierung von SELinux auf Android
 - ▶ derzeit: nur bestimmte Prozesse überwacht
- ▶ MacOS & iOS:
 - ▶ Sandboxing ähnlich zu SEAndroid
 - ▶ basiert auf TrustedBSD (MAC für BSD Systeme)

SELinux

- + Ausdrucksstark
- + Flexibel
- + Ausgereift

AppArmor

SELinux

- + Ausdrucksstark
- + Flexibel
- + Ausgereift
- (beliebig) kompliziert
- nicht in jeder Distribution nutzbar
- nicht robust bzgl. Änderungen der Systemkonfig

AppArmor

SELinux

- + Ausdrucksstark
- + Flexibel
- + Ausgereift
- (beliebig) kompliziert
- nicht in jeder Distribution nutzbar
- nicht robust bzgl. Änderungen der Systemkonfig

AppArmor

- + “einfach”
- + per Design auf bestimmte Prozesse begrenzt
- + läuft unter Ubuntu
- + “keine” bösen Überraschungen

SELinux

- + Ausdrucksstark
- + Flexibel
- + Ausgereift
- (beliebig) kompliziert
- nicht in jeder Distribution nutzbar
- nicht robust bzgl. Änderungen der Systemkonfig

AppArmor

- + “einfach”
- + per Design auf bestimmte Prozesse begrenzt
- + läuft unter Ubuntu
- + “keine” bösen Überraschungen
- Dokumentation
- Probleme können symptomlos bleiben (“fail silently”)
- ungeeignet für komplexe Systeme
- nur beschränkt erweiterbar

SELinux

- + Ausdrucksstark
- + Flexibel
- + Ausgereift
- (beliebig) kompliziert
- nicht in jeder Distribution nutzbar
- nicht robust bzgl. Änderungen der Systemkonfig

gr. Server mit vielen Diensten

AppArmor

- + “einfach”
- + per Design auf bestimmte Prozesse begrenzt
- + läuft unter Ubuntu
- + “keine” bösen Überraschungen
- Dokumentation
- Probleme können symptomlos bleiben (“fail silently”)
- ungeeignet für komplexe Systeme
- nur beschränkt erweiterbar

Workstations & kl. Server