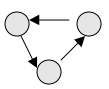
TECHNISCHE UNIVERSITÄT MÜNCHEN

Formale Modellierung von Sicherheitsanforderungen

Guido Wimmel

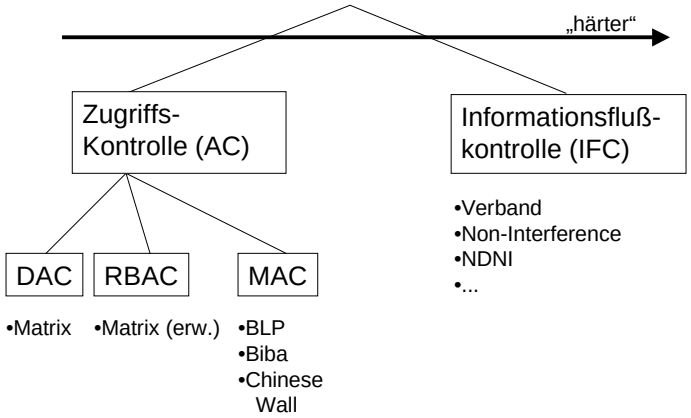
Motivation

- Informationssicherheit
  - Verhinderung unauthorisierter Informationsgewinnung und -veränderung
  - I.w.S. auch Verfügbarkeit, Verbindlichkeit, Anonymität



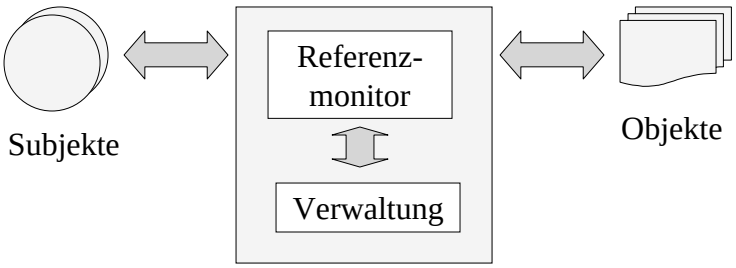
- Formales Sicherheitsmodelle
  - Spezifizieren Sicherheitsstrategie auf hohem Abstraktionsniveau
  - Klare Semantik
  - Formale Beweise von Sicherheitseigenschaften möglich
  - Prinzip: Subjekte, Objekte, Aktionen, Regeln
  - Generische Sicherheitsmodelle ermöglichen Wiederverwendung

Generische Sicherheitsmodelle



- Schwerpunkt: Vertraulichkeit / Integrität

Zugriffskontrolle



- Subjekte: aktiv (Personen, Gruppe, Prozesse, Rechner, ...)
- Objekte: passiv (Dateien, Ports, Speicherbereiche)
- Zugriffskontrolle muß geschützt sein

Zugriffsmatrix-Modell

	...	O_j	...
...		...	
S_i	...	***r***	...
...		...	

- Zugriffsrechte: read, write, execute...
- s darf o lesen, falls read ∈ M[s,o]
- Statische / dynamische ZM
- Speicherung: vertikal (ACL) / horizontal (capabilities)

Dynamische Matrix

- Harrison/Ruzzo/Ullman 1976
- Elementaroperationen: create, destroy, enter, delete
- Kommandos:

```
COMMAND revoke_r (S1,S2,X)
   IF owner ∈ M[S1,X] AND r ∈ M[S2,X]
   THEN delete r from M[S2,X]
END;
```

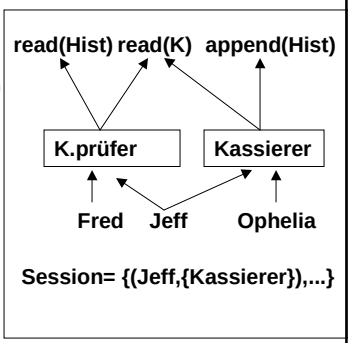
- Fragestellungen:
  - Z.B. Safety-Problem; Soll-Ist-Vergleich/ implizite Rechte

Zugriffsmatrix - Bewertung

- Flexibel einsetzbar (Granularität)
- Sehr einfach
- Probleme:
  - Inkonsistente Schutzzustände möglich
  - Rollenbasierung nur als Gruppenkonzept
  - Keine Informationsflußkontrolle
  - Keine systembestimmte Zugriffskontrolle
- Implementierung: z.B. UNIX, NT, DCE

Rollenbasierte Zugriffskontrolle

- R. Sandhu, 1996
- RBAC = (S,O,RL,P,sr,pr,session)
  - RL : Menge der Rollen (z.B. Angestellter, Kassierer, Kassenprüfer)
  - P : Rechte
  - sr : $S \rightarrow 2^{RL}$: Rollenmitgliedschaft
  - pr : $RL \rightarrow 2^{P}$: Zugriffsrechte f. Rolle
  - Session $\subseteq S \times 2^{RL}$: aktive Sitzungen
- Sicherheitseigenschaften:
  - $\forall s \in S: R_i \in session(s) \Rightarrow R_i \in sr(s)$
  - $\forall s \in S: exec(s,p) \Rightarrow \exists R_i \in RL: R_i \in session(s) \wedge p \in pr(R_i)$
- Vorteil:
  - Rollenprofile langlebig, gut für verteilte Systeme, need-to-know Prinzip
- Erweiterungen:
  - Rollenhierarchie, Aufgabentrennung



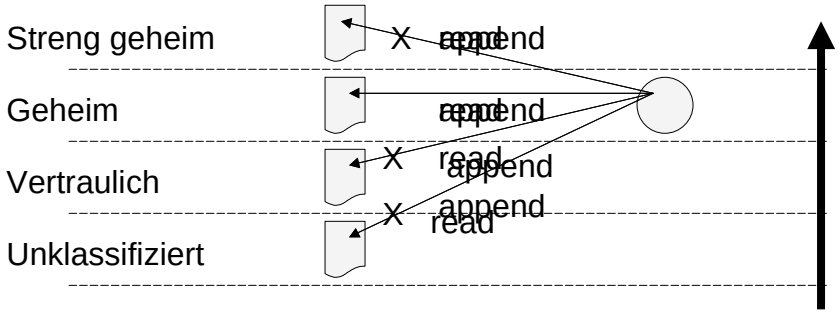
MAC/DAC

- Discretionary AC (DAC):
  - Eigentümer eines Objekt verantwortlich für Schutz
  - Probleme:
    - Widersprüchliche Rechtevergabe
    - Keine Kontrolle von Informationsfluß
    - Trojanische Pferde
- Mandatory AC (MAC)
  - Systembestimmte Zugriffskontrolle
  - Dominiert über DAC
  - $\Rightarrow$ Bell-LaPadula

Bell-LaPadula

- Bell, La Padula 1973
- Matrix-Modell mit P = {read,append,exec,rw}
- Verband SC = L $\times$ C von Sicherheitsklassen
  - aus Sicherheitsmarken L und Kategorien C
  - Z.B. L = {unklassifiziert, vertraulich, geheim, streng geheim}
    C = {},{Patient},{Arzt},{Arzt,Schwester}
  - $(L_1,C_1) \geq (L_2,C_2) \Rightarrow L_1 \geq L_2 \wedge C_1 \supseteq C_2$
- Regeln: no read up, no write down

Bell-LaPadula (2)



- No-read-up:
  - $r \in \{read,exec,rw\} \wedge r \in M[s,o] \Rightarrow sc(s) \geq sc(o)$
- No-write-down:
  - $r \in \{append,rw\} \wedge r \in M[s,o] \Rightarrow sc(s) \leq sc(o)$

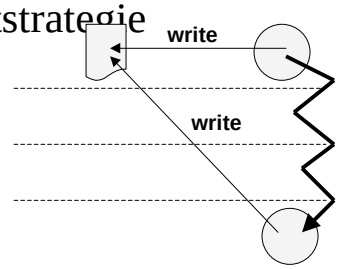
Informations-Fluß

Bell-LaPadula Diskussion

- Einschränkungen:
  - Zu restriktiv ($\Rightarrow$ trusted processes)
  - Nicht modellierbare Szenarien (z.B. Komposition)
  - Blindes Schreiben (Integritätsproblem)
  - Entferntes Lesen
- Integrität: Biba-Modell
- Eingesetzt z.B. in Trusted Solaris

Verdeckte Kanäle

- Übertragung von Information unter Umgehung der Sicherheitstrategie
  - Storage channels (e.g. locks, disk full)
  - Timing channels (e.g. CPU-Last)



- Gegenmaßnahmen
  - Schwierig...
  - V.a. Bandbreitenreduktion; Elimination durch Informationsflußkontrolle

Informationsflußkontrolle

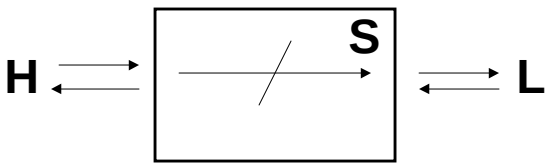
- Modelliert (un-)zulässige Informationsflüsse zwischen Objekten
- Idee: non-interference-Relation
  - Wenn $O_1 \not\rightarrow O_2$, dann
    - Vertraulichkeit für O_1
    - Integrität für O_2

Verbands-Modell

- Erweiterung von Bell-LaPadula
- Informationsfluß Objekt $x \rightarrow y$ erlaubt, wenn $sc(y) \geq sc(x)$
- Bestimmung von Informationsfluß durch Programmanalyse:
  - y:=x $\Rightarrow$ $x \rightarrow y$ (explizit)
  - z:=x;y:=z $\Rightarrow$ $x \rightarrow y$ (indirekt)
  - If x=0 then y=0 $\Rightarrow$ $x \rightarrow y$ (implizit)

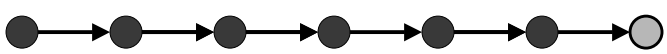
Non-Interference

- Goguen/Meseguer 1982; Ryan/Schneider, Gorrieri/Focardi, Lowe, ...



- Idee:
  - 2 Klassen: High, Low
  - High-Inputs dürfen Low-Sicht nicht beeinflussen
  - In CCS:

NNI: $(S \setminus_I Act_H) / Act_H \approx_T S / Act_H$



Non-Interference (2)

- Beispiel: kaputter Access-Monitor, ein Low-Objekt (Initialwert 0)

  γ = access_w(H).write(42).access_r(L).$\overline{\text{value}}$(42)

  γ / Act_H = access_r(L).$\overline{\text{value}}$(42) $\notin$ (S \ Act_H) / Act_H

- Vielzahl von Non-Interference-Begriffen
  - Basierend auf CCS,CSP,Automaten,...
  - NI,NNI,SNNI,BSNNI,BNDC,... (Sync., Deadlocks, ...)
  - Analog Äquivalenz von Prozessen
- Anwendung: z.B.
  - Analyse von Security-Kernels (e.g. Referenzmonitor)
  - Protokoll-Checking (Non-Interference Intruder->System)

NI und Refinement

- Failure-basiertes NI, CSP:
  - $INV_L = \forall tr,tr' \in traces(S): tr\uparrow Act_L = tr'\uparrow Act_L \Rightarrow$ $refs\text{-}after_L(S,tr) = refs\text{-}after_L(S,tr')$
- Zwei Prozesse:

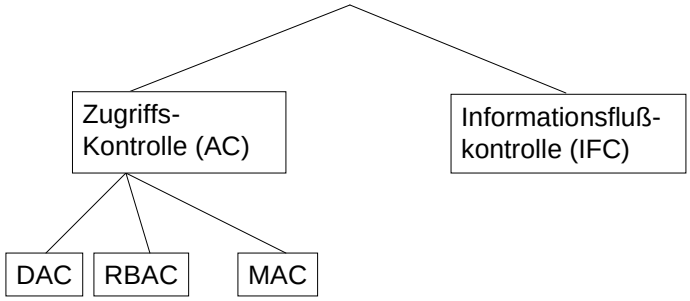
  λ_1 λ_2 λ_4 λ_3

  $Q_1 = h \rightarrow (l \rightarrow STOP \sqcap STOP) \quad (l \rightarrow STOP \sqcap STOP)$

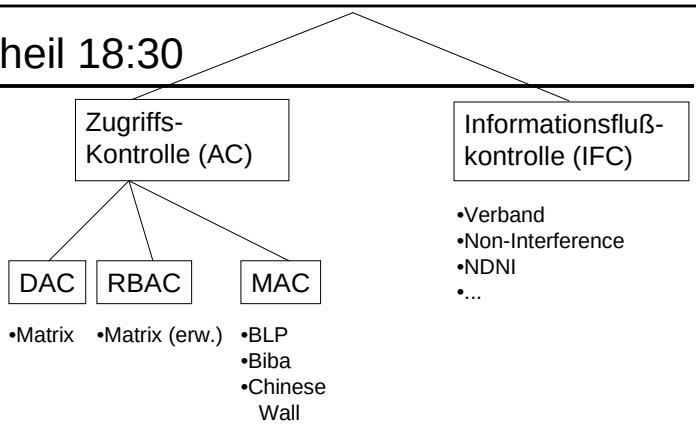
  $Q_2 = (h \rightarrow l \rightarrow STOP \quad l \rightarrow STOP) \sqcap h \rightarrow STOP$

- Lösungskonzept:
  - Scheduler wird in Semantik aufgenommen
  - Neuer NI-Begriff: $\forall$Scheduler: INV_L

Zusammenfassung



- Schwerpunkt: Vertraulichkeit / Integrität
- Sichtweise: geschlossene Systeme (BS,DB)
- ⇒Erweiterungen: Netzwerke, Kryptographie (Protokolle, PKI,...)
- Anwendungsbezogene Modelle (e.g. Smartcards, Prozessoren)
- 19.7.01, 17ct: Vortrag Lawrence Paulson (induktive Protokollmodellierung)

Schluß

Steinheil 18:30