14 Zugriffskontrolle: Einleitung (weitergeführt), Bell LaPadula

Satz 14.1 (Safety-Problem)**.** *Gegeben ein Sicherheitsmodell, eine (initiale) Zugriffsmatrix M und ein Zugriffsrecht r. Die Entscheidung, ob M sicher in Bezug auf r ist, ist unentscheidbar.*

Proof. Wir nutzen das Halteproblems für Turingmaschinen (unentscheidbar):

- Turingmaschine $A = (Q, \Sigma, \Gamma, q_0, \delta, F)$
  Übergangsfunktion $\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{\text{links}, \text{rechts}\}$
- O.B.d.A. linksseitig beschränktes Band, nur ein Endzustand q_f
- Halteproblem: Hält A bei Eingabe mit leerem Band (in q_f)
  (nur Blanksymbole B in den Zellen des Bandes)
- Start: A ist im Zustand q_0, Kopf steht auf erster Zelle.

Idee: Sicherheitsmodell soll A simulieren:

- Zugriffsrechte $R = Q \cup \Gamma \cup \{\text{own}, \text{end}\}$
- Subjekte, Objekte: $S = O = \{c_1, c_2, \dots\}$ (Zellen von A)
- Zugriffsmatrix wird durch δ modifiziert (siehe unten)
- Ziel: Recht q_f wird hinzugefügt gdw. A hält (in q_f)

Anfangskonfiguration des Sicherheitsmodells $M_1 =$

	c_1
c_1	$\{q_0, B, \text{end}\}$

Bedeutung: A ist in Zustand q_0, liest B und hat bisher nur Zelle 1 besucht.

Weiteres Beispiel: A hat Zellen 1-4 mit wxyz beschrieben
und steht im Zustand p auf Zelle 2

	c_1	c_2	c_3	c_4
c_1	{w}	{own}		
c_2		{x, p}	{own}	
c_3			{y}	{own}
c_4				{z, end}

Bedeutung own: own $\in m_{c_i c_j}$ gdw. $j = i + 1$ (aufeinanderfolgende Zellen)

Beschreibung der Matrixoperationen über die Übergangsfunktion:

- Für $\delta(q, x) = (p, y, \text{links})$:

  command $c_{qx}(c, c')$
  if own $\in m_{cc'}$, $q, x \in m_{c'c'}$ then
  delete q, x from $m_{c'c'}$
  enter y into $m_{c'c'}$
  enter p into m_{cc}
  end if
  end

- Für $\delta(q, x) = (p, y, \text{rechts})$ zwei Fälle (TM geht in alte oder neue Zelle)

  command $c_{qx}(c, c')$ (schon besuchte Zelle wird erneut besucht)
  if own $\in m_{cc'}$, $q, x \in m_{cc}$ then
  delete q, x from m_{cc}
  enter y into m_{cc}
  enter p into $m_{c'c'}$
  end if
  end

  command $c'_{qx}(c, c')$ (neue Zelle wird besucht)
  if end, $q, x \in m_{cc}$ then
  delete end, q, x from m_{cc}
  enter y into m_{cc}
  create c'
  enter end, p, B into $m_{c'c'}$
  enter own into $m_{cc'}$
  end if
  end

Es gilt: Recht q_f zu M hinzugefügt gdw. A Zustand q_f erreicht.
Also: Alg. für Safety-Problem würde auch Alg. für Halteproblem liefern. □

Zwei Kategorien von Sicherheitsmodellen

- benutzerbestimmt (Discretionary Access Control, DAC):
  Nutzer legen Zugriffsrechte ihrer Dateien fest (gängige OSs)
- systembestimmt (Mandatory Access Control, MAC):
  Meißt für kritische Systeme (Geheimdienste, Militär)

Bell-LaPaduda Sicherheitsmodell

David Bell and Leonard LaPadula (1973) im Auftrag der US Air Force

- Erstes verifiziertes Sicherheitsmodell
- Erweiterung des Matrixmodells um systembestimmte Eigenschaften
- Sicherheitsziel: Vertraulichkeit (Informationsflusskontrolle)

Einfaches Bell-LaPaduda-Modell:

- Zugriffsrechte $R = \{\text{read}, \text{write}, \text{execute}, \text{control}\}$
- Sicherheitsmarken $SM = \{\text{unklassifiziert}, \text{vertraulich}, \text{geheim}, \text{streng geheim}\}$
  mit entsprechender Ordnung $\leq$: unklassifiziert $\leq$ vertraulich $\leq$ geheim $\cdots$
  Man spricht daher auch von einem Multi-Level-Security-Modell (MLS)
- Menge von Subjekten S und Objekten O mit
  - clearance $cl : S \longrightarrow SM$ (Subjekte erhalten Ermächtigung)
  - classification $cl : O \longrightarrow SM$ (Objekte erhalten Einstufung)
  - Zugriffsrechte von Subjekten an Objekten: Matrix $M = (m_{s,o})_{\substack{s \in S \\ o \in O}}$
- Zugriffskontrolle:
  - (i) Kontrolle von Zugriffen über Zugriffsmatrix M
  - (ii) systembestimmt I: Simple Security Property, no read-up
    $\text{read} \in m_{so} \Rightarrow cl(o) \leq cl(s)$: lesen nur mit entsprechender Ermächtung
  - (iii) systembestimmt II: $\star-$Property, no write-down
    $\text{write} \in m_{so} \Rightarrow cl(s) \leq cl(o)$: kein Informationsfluss nach unten

(*iv*) systembestimmt III: Strong Tranquility Property:
Nur vertrauenswürdige Personen können M und cl ändern
(z.B. Sicherheitsbeauftragte)

Formalisierung:

- Zustände: Tripel (b, M, cl) mit
  - $b = ((s_1, o_1, r_1), (s_2, o_2, r_2), \dots)$:
    Aktuelle Zugriffe mit r_i durch Subjekt s_i auf Objekt o_o
  - M: aktuelle Zugriffsmatrix
  - cl : aktuelle Ermächtigungs-/Einstufungsfunktion
- Sicherer Zustand: Alle Regeln werden beachtet
- Zustandsübergänge, Änderung des Tripels (b, M, cl)
- Sichere Zustandsübergänge:
  - Änderung von b durch Nutzer unter Beachtung von $(i), (ii), (iii)$
  - Änderung von M, cl unter Beachtung von (iv)

Satz 14.2 (Sicherheitstheorem)**.** *Nachfolgezustand ist sicher, wenn*

- *Vorgänger sicher ist, und*
- *ein sicherer Übergang genutzt wurde.*

Problem 1: Need-to-Know-Prinzip aufwändig umsetzbar
Nur in Zugriffsmatrix: Für alle Subjekte und Objekte

Lösung: Festlegung von Zuständigkeitsbereichen

Erweiterung des Modells:

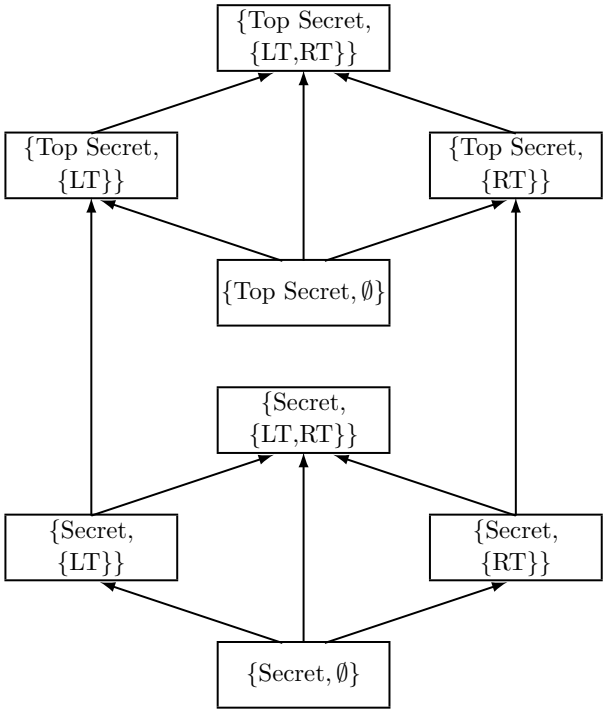
- Menge von Zuständigkeitsbereichen Z
  z.B. Links-, Rechtsterrorismus, Org. Kriminalität, Islamismus, ...
- Sicherheitsklassen $SK = SM \times \mathcal{P}(Z)$ mit partieller Ordnung $\leq$
  $(A, B) \leq (A', B') :\Longleftrightarrow A \leq A'$ und $B \subseteq B'$

- Erweiterung der Funktionen cl zu
  - $sc : S \longrightarrow SK$. Bedeutung $sc(s) = (A, B)$
    Subjekt s hat die Ermächtung A
    s ist Subjekt im Zuständigkeitsbereich b für alle $b \in B$.
  - $sc : O \longrightarrow SK$: Bedeutung $sc(o) = (A, B)$
    Objekt o hat die Eintstufung A
    o ist Objekt im Zuständigkeitsbereich b für alle $b \in B$.

Übung: Wie müssen die Eigenschaften (i) und (iii) angepasst werden?

Beispiel. Zwei Sicherheitsmarken: $SM = \{\text{secret}, \text{top secret}\}$
Zwei Zuständigkeitsbereiche: $Z = \{\text{Linksterrorismus}, \text{Rechtsterrorismus}\}$.

Darstellung von $(SM \times \mathcal{P}(Z), \leq) : ((A, B) \to (A', B')$ heißt $(A, B) \leq (A', B')$



Formal: SK zusammen mit $\leq$ bildet einen Verband:

- $(SK, \leq)$ ist eine partielle Ordnungsrelation
  (transitive, reflexive und antisymmetrische)
- Für je zwei Elemente $A, B \in SK$ gilt:
  - es gibt eindeutiges Infimum $\text{glb}(A, B) \in SK$ (greatest lower bound)
    d.h. $\text{glb}(A, B) \leq A, B$ und für alle $L \leq A, B$ gilt $I' \leq \text{glb}(A, B)$.
  - es gibt eindeutiges Supremum $\text{lub}(A, B) \in SK$ (least upper bound)
    d.h. $\text{lub}(A, B) \geq A, B$ und für alle $U \geq A, B$ gilt $U \geq \text{lub}(A, B)$.
- Informationsfluss nur über die Halbordnung
- Erweitertes BLP: Bsp. für verbandbasiertes Informationsflussmodell

Problem 2: Kein Schreiben von oben nach unten
Vorgesetzter kann keine Anweisung nach unten erteilen

Lösung: Zeitlich beschränkte untere Sicherheitsklasse
Es gibt für Subjekte zwei Funktionen

- $sc_s(s)$: Sicherheitsklasse von Subjekt s
- $sc_c(s)$: aktuelle Sicherheitsklasse von Subjekt s
  Aktuelle Klasse wird vom Nutzer selbst festgelegt
- Es gilt stets $sc_c(s) \leq sc_s(s)$
  Sicherheitsklasse dominiert die aktuelle Sicherheitsklasse

Übung: Wie muss die Eigenschaften (iii) angepasst werden?

Problem 3: Kein Integritätsschutz

- $\star-$property: Erlaubt schreiben nach oben
  und damit Änderung eingestufter Informationen
- Beispiel für Sicherheitsmodelle für Integritätsschutz: Biba (Übung)

Problem 4: Keine Erfassung verdeckter Informationskanäle
Kanäle, die nicht für Informationsfluss vorgesehen sind

Beispiel.

- Ressourcen-Konflikt: High-Level-Prozess erzeugt eine Datei, so dass ein Low-Level-Prozess eine Fehlermeldung erhält, wenn eine Datei mit gleichem Namen erzeugt wird

- High-Level-Prozess kann gemeinsam genutzte Ressourcen (Position des Festplattenkopfes, Inhalt des Caches) in einem Zustand hinterlassen, so dass der Low-Level-Prozess anhand von Antwortzeiten zusätzliche Informationen gewinnen kann (Seitenkanalangriff)

Lösung: Nächste Woche