

Sicherheit von Informationssystemen

Dr. Adrian Spalka

Institut für Informatik III, Universität Bonn

Übersicht:

- Einführung
- Grundlegende Konzepte
- Kryptographie
- Zugriffskontrollsysteme
- Datenbanken

Allgemeine Begriffsklärung

Ziel der Computersicherheit:

Schutz des Wertes, den die Daten und Programme haben

Methodik:

- i)* **Sicherheitspolitik:** Programmatische Zielsetzung
 - Informelle Bestimmung des Wertes und der Schutzbedürfnisse Daten

- ii)* **Sicherheitsmodell:** Deklarative Spezifikation
 - Übersetzung der Sicherheitspolitik in den Kontext eines Computermodells

- iii)* **Implementierung:** Operationale Umsetzung
 - Praktische Umsetzung eines Sicherheitsmodells auf einem Computer(-Netz)

Probleme der Umsetzung

Sicherheitspolitik → Sicherheitsmodell:

Sicherheitsmodell verfehlt die Absicht der Sicherheitspolitik:

- Politik an sich mehrdeutig
- Politik stützt sich auf implizite Annahmen
- Modell ist zu ausdrucksschwach

Sicherheitsmodell → Implementierung:

Implementierung weist die Eigenschaften des Modells nicht auf:

- Modell berücksichtigt zu wenige Gefahrenfaktoren
- Deklarative Beschreibung vernachlässigt Seiteneffekte der Operationen

Wertbestimmung

Art der Wertbestimmung in der Sicherheitspolitik:

- **Verbindliche** (engl. mandatory):
 - Legt alle Werte und ihre Schutzbedürfnisse verbindlich fest
- **Nach freiem Ermessen** (engl. discretionary):
 - Legt fest, in wessen freiem Ermessen die Festlegung der Werte und ihrer Schutzbedürfnisse liegt

Faktoren der Wertbestimmung:

- **Verfügbarkeit:** Schutz vor Verzögerung des Zugriffs auf Daten
 - Z.B.: Ein Systemausfall darf nicht länger als 4 Stunden dauern
- **Integrität:** Schutz vor unbefugter Modifikation von Daten
 - Z.B.: Kein Angestellter darf sein eigenes Gehalt verändern
- **Vertraulichkeit:** Schutz vor Bekanntgabe der Daten an Unbefugte
 - Z.B.: Kaufabsichten dürfen nicht vor dem Kaufzeitpunkt bekannt werden

Wertbestimmung: Beispiele

Bereich	Werte	Schutzfaktoren
Bahnverkehr	Kollisionsfreiheit der Züge	Integrität
Banken	Schutz vor Betrug	Integrität
Flug-Tower	Leben der Passagiere	Verfügbarkeit / Integrität
Militär	Erfolg von Operationen	Vertraulichkeit
Medizin	Gesundheit der Patienten	Integrität / Verfügbarkeit / Vertraulichkeit

Maßnahmen zur Werterhaltung

Klassifikation der Schutzmaßnahmen zur Werterhaltung:

- i)* **Vorbeugung:** Sollen Wertverlust verhindern
 - z.B. Zugangskontrollen

- ii)* **Wiederherstellung:** Sollen Wertverlust nachträglich beheben
 - z.B. Aufzeichnungen und Protokolle

- iii)* **Verlustbegrenzung:** Sollen Wertverlust begrenzen
 - z.B. Einbruchmelder

Zugangskontrollen

Kontrolle des Zugangs:

- **Identifikation:** Angabe einer individuellen Kennung
 - Z.B. Benutzer- / Rollen- / Knotenname
- **Authentisierung:** Bestätigung der Echtheit der Kennung durch z.B.:
 - **persönliches/systemeigenes Wissen:** Paßwort (vergessen, aufgeschrieben)
 - **persönlicher Besitz:** Smartcard (beschädigt, dupliziert)
 - **persönliche Eigenschaft:** Fingerabdruck, Unterschrift (nicht akzeptiert, unzuverlässig)

Arten der Identifikation und Authentisierung

- **Einseitige:** Benutzer muß sich vor dem Rechner ausweisen
- **Gegenseitige:** Benutzer und Rechner müssen sich gegenseitig ausweisen

Sicherung der Verfügbarkeit

Verfügbarkeit:

- **Beschränkung** der Betriebsmittelnutzung
 - statische/dynamische Quotas

- **Redundanz** der Betriebsmittel und Daten
 - **zeitgleiche**: lokal (dual-disk Systeme) / im Netz (Mehrwegverbindungen)
 - **zeitversetzte**: (periodische) Archivierung

- **Externe physikalische Maßnahmen**
 - Schutz gegen Stromausfall, Feuer, Diebstahl

Sicherung der Integrität

Integrität:

- **Hardware-Integrität:** Veränderung der Rechner
- **Einhaltung der Zugangswege:** anwendungsspezifischer Zugang
- **Unverletzlichkeit der Datenwege:** Modifikation des Datenstroms
- **Beschränkung** von Kompetenzen und **Aufteilung** von Vollmachten: Rechtesysteme
 - SQL-GRANT/REVOKE-Befehle
- **Semantische Integrität:** unsinnige Daten
 - Integritätsbedingungen

Sicherung der Vertraulichkeit (1)

Generelle Problematik: präzise Definition von Vertraulichkeit

Frage: Haben Sie ein Konto

- i)* bei der **Sparkasse**?
- ii)* bei der **Castle Bank of Nassau**?

Antworten:

i) Ja.

ii) Vielleicht. / Sage ich nicht. / Kein Kommentar.

iii) Nein.

iv) Ich verstehe die Frage nicht.

Sicherung der Vertraulichkeit (2)

Vertraulichkeit:

- Zugang kann kontrolliert werden: **Zugangsbeschränkung**
 - Betriebssysteme: Login-Prozedur
 - Datenbank:
 - SQL-GRANT/REVOKE
 - Sichten (engl. views)
- Zugang kann nicht kontrolliert werden: **Chiffrierung**
 - offene Netze
- Benutzer kennt Fragmente des Umfelds: **Fehlinformationen**

Inferenz: Benutzer folgert vertrauliche Informationen aus eigenem Wissen + Antworten der Datenbank

 - statistische Datenbanken
 - Datenbanken mit Integritätsbedingungen

Rechtevergabe

Grundlage der Rechtevergabe:

Vertrauenswürdigkeit des Benutzers bzgl. Vertraulichkeit/Integrität/Verfügbarkeit

⇒ **Bedacht**: Explizite Annahmen über das **Benutzerverhalten**

⇒ **Oft unbedacht**: Implizite Annahmen über das **Programmverhalten**

- ggf. nicht vertrauenswürdig, z.B. **Trojanisches-Pferd-Programm**

Präventive Gefahrenbeseitigung

- Prüfung der Vertrauenswürdigkeit, d.h. **Verifikation**, daß ein Programm:
 - keine undokumentierte Funktionalität aufweist (engl. **security**)
 - die zugesicherte Funktionalität besitzt (engl. **safety**)

- **Reduktion der Kompetenzen** von Programmen:
 - Einschränkung der Vergabe von Lese- und Schreibrechten
 - Trennung und Strukturierung von Gruppenumgebungen

- **Pauschale Trennung** von Programmen in
 - vertrauenswürdige (\Rightarrow [engl.] Trusted Computing Base, TCB)
 - nicht vertrauenswürdige

Wiederherstellung und Verlustbegrenzung

Wiederherstellung:

- i)* **Nachvollziehbarkeit** von Handlungen
- ii)* **Zuordnung** zu den Auftraggebern und Ausführenden
 - Protokollierung und nachträgliche Auswertung (engl. audit and control)

Verlustbegrenzung:

- i)* **Überwachung** laufender Aktivitäten
- ii)* **Alarmierung** bei vermuteten Verstößen,
d.h. bei zulässigen jedoch ungewöhnlichen Aktivitätsmustern
 - **Einbruchmelder** (engl. intrusion detector)
⇒ Problem: Privatsphäre und Datenschutz

Netzwerkspezifische Sicherheitsprobleme

- Nicht vertrauenswürdige Knoten
- Unsichere Kommunikationswege
- Interaktive Kommunikation mit passiven und aktiven Komponenten
 - Dienstanbieter: entfernte Zugangsbeschränkung und Zugriffskontrolle
 - Kunde: vertrauenswürdige Dienstleistung (Java, ActiveX)
- Nachbildung der Postdienste
- Verteilte Überwachung: Globale Protokollierung und Analyse
- Erkennung von Eindringlingen

Grundbegriffe der Chiffrierung (1)

Problem:

- A sendet an B Nachrichten über einen **ungesicherten Kanal**
- C kann unrechtmäßig

den Kanal abhören, Nachrichten herausnehmen, einfügen oder verändern

Lösungsansatz:

- A verfaßt Nachricht X
- A benutzt eine **Verschlüsselungsfunktion** V : Nachricht $X \Rightarrow$ Nachricht Y
$$V(X) = Y$$
- A sendet Y an B
- B benutzt eine **Entschlüsselungsfunktion** E : Nachricht $Y \Rightarrow$ Nachricht X
$$E(Y) = X$$
- B liest X

Grundbegriffe der Chiffrierung (2)

Erwünschte Eigenschaften von V und E :

- Die Bestimmung von X bei gegebenem Y ist **praktisch nicht durchführbar**
- V und E sind **schnell und leicht** benutzbar

Wege:

- **Steganographie**: Verschleierung der Verschlüsselung (\Rightarrow digitale Wasserzeichen)
- **Geheimhaltung der Funktionen V und E**
- Erweiterung der bekannten Funktionen V und E um Schlüsselparameter,

$$V(X, A_1, \dots, A_k) = Y$$

$$E(Y, B_1, \dots, B_l) = X$$

und **Geheimhaltung der Schlüssel**

Symmetrische kryptographische Verfahren

Ansatz: Verwirrung stiften, z.B. DES

Berechnung: primitive Operationen, z.B. Bit-Rotation, -Permutation und -Substitution

Weg:

- **Gemeinsamer geheimer Schlüssel S** zur Ver- und Entschlüsselung

$$V(X, S) = Y \text{ und } E(Y, S) = X$$

- **Vorteile:** schnell (Hardwareimplementierung)
- **Nachteile:**
 - Schlüsselübermittlung
 - hohe Schlüsselanzahl
 - komplizierte Schlüsselverwaltung

Asymmetrische kryptographische Verfahren (engl. public key)

Ansatz: Schwierige mathematische Probleme

- **Primfaktorzerlegung** natürlicher Zahlen (RSA) – heute > 150 Stellen
- Berechnung **diskreter Logarithmen** in endlichen Körpern – heute > 10^{120} Elemente

Berechnung: komplexe mathematische Operationen

Weg:

- **öffentlicher Schlüssel** (engl. public key) zum Verschlüsseln

$$V(X, S_1) = Y$$

- **geheimer Schlüssel** (engl. private key) zum Entschlüsseln

$$E(Y, S_2) = X$$

- **Vorteile:** einfache Schlüsselverwaltung und geringe Schlüsselanzahl
- **Nachteile:** langsam, Echtheit öffentlicher Schlüssel

Nutzen und Grenzen der Kryptographie

Nutzen:

- Vertraulichkeit und Integrität auf ungesicherten Kommunikationswegen
- Verifikation von Wissen
- Zugangskontrolle ohne Benutzerkonten im Betriebssystem

Sicherheit \neq Kryptographie

Grenzen:

- Verfügbarkeit
- Vertraulichkeit und Integrität bei Trojanischen Pferden
- Vertraulichkeit in Datenbanken
- Schlüsselverwaltung

Zugangsbeschränkung durch Zugriffskontrollsysteme

Ziele: Gewährleistung von

- **Vertraulichkeit:** Rechte zur Feststellung der Existenz und des Zustands
- **Integrität:** Rechte zur Änderung des Zustands und zum Löschen

Allgemeine Komponenten:

- *O*: Menge der **Schutzseinheiten** (auch Objekte), z.B. Dateien
- *R*: Menge objektspezifischer **Rechte**, z.B. Lesen, Schreiben, Versenden
- *S*: Menge aktiver Einheiten (auch Subjekte), Teilmenge von *O* = **Prozesse**
- *U*: Menge der **Benutzer**

Spezifische Komponenten:

- *Z*: Zugriffskontrollregeln
- *V*: Zusatzvereinbarungen über Beziehung: Benutzer \longleftrightarrow Subjekte

Zugriffskontrollsysteme nach freiem Ermessen

(Engl. **discretionary access control**, DAC)

Zugriffskontrollregeln

- Jedes Objekt hat einen Benutzer als **Besitzer**
- Rechtevergabe: im **freien Ermessen** des Besitzers

Momentaufnahme der Rechtezuordnung:

- **Zugriffsmatrix** (ungeordnet):
- **Zugriffskontrollliste** per Objekt: $R(o_i) = \{(u_j, R_{i,j}), \dots\}$
- **Fähigkeitenliste** per Benutzer: $R(u_i) = \{(o_j, R_{i,j}), \dots\}$

	o_1	\dots	o_n
u_1	o	\dots	rw
\vdots	\vdots	\ddots	\vdots
u_m	\emptyset	\dots	rd

Zusatzvereinbarung: (Bisher stets **stillschweigend**)

Die Vertrauenswürdigkeit eines Benutzers überträgt sich auf seine Subjekte
⇒ Kein Schutz gegen Trojanische-Pferde-Programme
⇒ ABER: Schwäche im Design – kann behoben werden!

Verbindliche Zugriffskontrollsysteme

(Engl. **mandatory access control**, MAC)

Idee: Übergeordnete Instanz legt Spielraum der Benutzer verbindlich fest

Zugriffskontrollregeln:

- Besitzer eines Objektes ist ein Benutzer oder die übergeordnete Instanz
- Rechtevergabe: im Rahmen des festen Spielraums

Momentaufnahme der Rechtezuordnung:

- **Generell:** wie bei Zugriffskontrollsystemen nach freiem Ermessen
- **Speziell:** unter Ausnutzung von Gruppenstrukturen

Zusatzvereinbarung: (Bisher oft **ausdrücklich**)

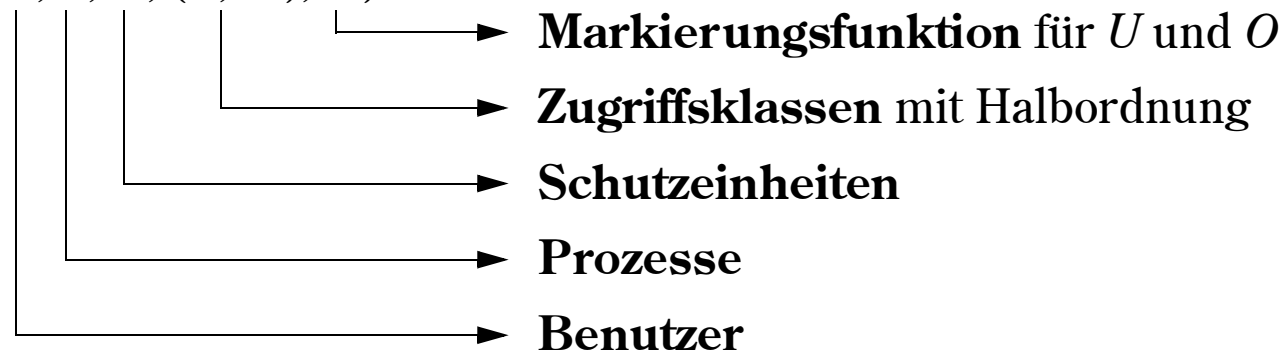
Die Vertrauenswürdigkeit eines Benutzers überträgt sich **nicht** auf seine Subjekte
⇒ Heute: **Schutz** der Vertraulichkeit gegen einige Trojanische-Pferde-Programme
⇒ **ABER:** Noch kein Schutz der Integrität und Verfügbarkeit

„Multilevel“-Systeme (1)

Ziel: Gewährleistung der **Vertraulichkeit**

Idee: Einordnung der Benutzer und Schutzeinheiten in eine **Hierarchie**

Weg: $BLP = (U, S, O, (L, \leq), G)$



- **Schutzeinheit o** erhält von G eine **Stufe der Vertraulichkeit** $G(o)$
- **Benutzer u** erhält von G eine **Stufe der maximalen Vertrauenswürdigkeit** $G(u)$

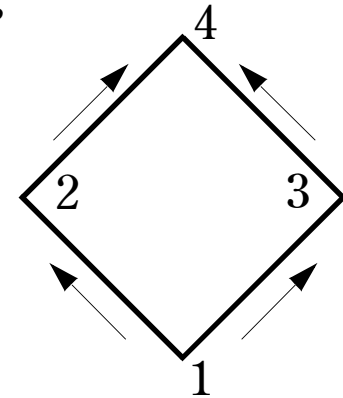
Anforderung:

- Schutzeinheit o muß vor Benutzer u vertraulich gehalten werden, falls $G(u) \geq G(o)$ nicht gilt.

„Multilevel“-Systeme (2)

Arbeitsregeln:

- Beim Login: Benutzer u wählt Stufe l , $l \leq G(u)$, für **Arbeitssitzung**
- Alle Prozesse s des Benutzers u erhalten die Stufe l , d.h. $G(s) = l$



Zugriffskontrollregeln:

- Besitzer eines Objektes ist eine **übergeordnete Instanz**
- Neue Objekte o erhalten die Stufe des erzeugenden Prozesses s , d.h.

$$G(o) = G(s) = l$$

- Zugriffsrechte während einer Arbeitssitzung: bestimmt von

Stufe des Subjektes, d.h. $G(s) = l$, und Stufe des Objektes:

- Lesezugriff nur „**nach unten**“ gestattet, d.h. $G(s) \geq G(o)$ (engl. simple-security-p.)
- Schreibzugriff nur „**geradeaus**“ gestattet, d.h. $G(s) = G(o)$ (engl. *-property)

„Multilevel“-Systeme (3)

Vorteil:

- Schützt lokal die **Vertraulichkeit** gegenüber Trojanischen Pferden
ABER nur in freien Containersystemen

Schwachstellen:

- Beschränkte **Funktionalität** der Arbeitssitzung
- Kein Schutz der **Integrität** gegen Trojanische Pferde
- Falls Zugang zum Internet ⇒ Kein Schutz der **Vertraulichkeit**
- Falls Datenbank mit Integritätsbedingungen ⇒ Kein Schutz der **Vertraulichkeit**
- Kein Schutz gegen **verdeckte Kanäle** (engl. covert channels)

Statistische Datenbanken

Ausgangslage:

- Relation $R(\text{Name}, A_1, \dots, A_n)$
- Benutzer darf über R nur statistische Anfragen stellen, z.B. count, sum und avg, und Bedingungsteil (SQL: WHERE) darf nur A_1, \dots, A_n benennen

Ziel: Benutzer darf kein vollständiges Tupel rekonstruieren können

- Unterbindung der Zuordnung eines Namens zu seinen Attributwerten

Problem: Benutzer kennt eine Invariante (Integritätsbedingung) der Form

$\exists x_1 x_2 \dots : r(a_1, a_2, \dots, x_1, x_2, \dots)$, d.h. z.B. $R(\text{Maier}, \dots, a_i, \dots, a_j, \dots)$
⇒ er kann ALLE Attributwerte rekonstruieren

Schutzmethoden:

- Keine universelle Lösung. Vorschläge stets mit Nachteilen verbunden, z.B.
 - Einschränkung der Abfragen, fehlerbehaftete Antworten, enormer Verwaltungsaufwand

Sicherheit in SQL (1)

Idee:

- Nachbildung von Zugriffskontrollen im freien Ermessen in Betriebssystemen für Datenbanken
- Semantische Integrität

Ansatz:

- Keine Betrachtung von expliziten Vertraulichkeitsforderungen
- Statt dessen zwei *low-level* Mechanismen:
 - Rechtesystem auf Relationen: Grant/Revoke
 - Einführung neuer Prädikate: Sichten (engl. views)
- Integritätsbedingungen in SQL: nur zwei Arten
 - Primärschlüssel: Namen für Tupel
 - Fremdschlüssel: Namentliche Zuordnung von Tupeln in zwei Relationen

Sicherheit in SQL (2)

Vorteile:

- Einfach zu implementieren

Nachteile:

- *High-level* Konzepte lasten auf Benutzer/Designer
- Rechte nur auf Relationen, nicht aber auf Tupeln \Rightarrow Views als Ausweg
- Views sind Ausdrücke \Rightarrow Kein Ausschluß individueller Tupel
- Update-Anomalien und -Probleme bei Views
- Ausdrucksschwach: keine Vertraulichkeitsforderungen für fehlende Tupel
- Generell: Views als Instrument der Tupelvertraulichkeit

Vergabe von Rechten in SQL

Rechte auf Dateneinheiten:

```
GRANT {ALL [PRIVILEGES][column_list] | permission_list [column_list]}  
ON {table_name [(column_list)] | view_name [(column_list)] | stored_procedure_name}  
TO {PUBLIC | name_list }  
[WITH GRANT OPTION]
```

- *permission*: SELECT, INSERT, DELETE, UPDATE, REFERENCES und EXECUTE

Rechte auf SQL-Befehle:

```
GRANT {ALL | statement_list} TO { PUBLIC | name_list}
```

- *statement*: CREATE DATABASE, CREATE DEFAULT, CREATE PROCEDURE, CREATE RULE, CREATE TABLE, CREATE VIEW, DUMP DATABASE und DUMP TRANSACTION

Entzug von Rechten in SQL

Rechte auf Dateneinheiten:

```
REVOKE [GRANT OPTION FOR]
{ALL [PRIVILEGES] | permission_list } [(column_list)]
ON { table_name [(column_list)] | view_name [(column_list)] | stored_procedure_name
| extended_stored_procedure_name }
FROM {PUBLIC | name_list} [CASCADE]
```

Rechte auf SQL-Befehle:

```
REVOKE {ALL | statement_list} FROM {PUBLIC | name_list}
```

Bemerkungen:

- Die Semantik der Rechte ist abhängig von der Reihenfolge, in der die Vergabe und der Entzug stattgefunden haben. Gültig ist immer der zuletzt ausgeführte Befehl.
- Der Erzeuger eines Objektes wird sein Besitzer.
- Der Besitzer behält immer die Rechte auf seine Objekte.

Beispiel: MS SQL Server

Stored procedures:

- Mittel der Zugriffskontrolle als wohlgeformte Transaktion
- Benutzer kann auf *stored procedures* Zugriff bekommen ohne Zugriff auf Rels zu haben

Trigger: Definiert bzgl. einer Relation

- Einsatz: Definition von Integritätsbedingungen / Laufende Berechnung von Werten

Alarmierung:

- Reaktion auf Ereignisse: Ausführung eines Programms / Versendung einer Nachricht
- Ereignisarten: *Performance monitor* Grenzwerte / *NT Event Log* Einträge

Identifikation und Authentisierung:

- Integrierte Sicherheit:
MS SQL Server verläßt sich auf Mechanismen von MS Windows NT
- Standardsicherheit:
MS SQL Server übernimmt selbst die Identifikation und Authentisierung der Benutzer

Verfügbarkeit im MS SQL Server

Datenimport/-export aus/in Dateien: bcp, *bulk copy program*

Backup/Restore: DUMP/LOAD-Kommandos

- Quelle: master-DB, msdb (schedule)-DB und alle Benutzer-DBs + TA-logs
- Optionen:
 - Einmalig/periodisch
 - überschreibend/anhängend

Replikation: Automatische Verteilung von Lesekopien von DB(-Teilen)

- Paradigma: Verleger – Verteiler – Abonnent
- Jeder DB-Server kann zugleich Verleger/Verteiler und Abonnent sein
- Replikationsarten: Enge-synchrone / lose-zeitversetzte
- Abonnementarten:
 - Push: Verteiler verschickt Änderungen; Verwaltet wird der Verteiler
 - Pull: Abonnent fordert Änderungen an; Verwaltet wird der Abonnent

Konzept der Vertraulichkeit

Ausgangspunkt:

- Vertraulichkeitsforderung

„Benutzer A möchte X vor Benutzer B vertraulich halten“

im Kontext einer Datenbank interpretieren.

Fragen:

- Was kann X sein? (Definition der Schutzeinheiten)
 - Datenbank, Relation, Tupel, Attribute, Attributwerte
- Was bedeutet es, X vertraulich zu halten? (Definition des erwünschten Zustands)
 - Inhaltsteile, Existenz, Wissen um die Existenz
- Wie kann der erwünschte Zustand erreicht werden? (Umsetzung)
 - Statische und dynamische Semantik

Traditionelle „multi-level“ Datenbanken

Schritt in die semantische Dunkelheit: Denning *et al* (1988):

SeaView = Offene relationale DB + *mandatory access control*

SeaView *multi-level relational data model* (ML-RDB):

- *Multi-level Relation*: $R(A_1:D_1, L_1, \dots, A_k:D_k, L_k, L_T):L_R$
- Integritätsbedingungen: nur Primär- und Fremdschlüssel (Fehlerhafte Annahme)
- Anfragebeschränkung: nur Daten „von unten“
- Änderungsbeschränkung: nur Daten auf gleicher Stufe

Ziel: Gute Sicherheitseigenschaften der Betriebssysteme auf Datenbanken übertragen

Resultat: (bereits 1987 erkannt)

- Keine statische Semantik: Kein intendiertes Modell / Def. konsistenten Zustands
- Keine dynamische Semantik: Was ist der Effekt einer Transaktion?

Fehler: (BS-motiviert) Sicherheitsstufen als Zugriffsklassen interpretiert

Lösung: Alles nochmal von vorne \Rightarrow Axiomatischer Neuansatz

SeaView-Beispiel

	Starship	AC	Objective	AC	Destination	AC
1.	Enterprise	high	fun	high	Asgard	high
2.	Enterprise	low	explore	low	Muspelheim	low
3.	Voyager	low	cargo	low	Midgard	high
4.	Voyager	low	cargo	low	Nifelheim	low

- Tupel (1) und (2): Tupel-Polyinstanziierung
- Tupel (3) und (4): Element-Polyinstanziierung

Entstehung der Konfusion:

- *Low*-Perspektive:
Prüfung der Integritätsbedingungen darf nur *sichtbare* Tupel einbeziehen
- *High*-Perspektive:
Transaktionen dürfen keine Auswirkung „unterhalb“ der aktuellen Stufe haben

Axiomatischer Neuansatz

Grundlegende Eigenschaften:

- i)* Globale Konsistenz:
Die offene globale Datenbank ist immer konsistent.
- ii)* Lokale Konsistenz:
Eine lokale Datenbank ist immer konsistent.
- iii)* Unterordnende Konsistenz:
Lokale Konsistenz impliziert globale Konsistenz.
- iv)* Respektierung der Benutzerkompetenzen (Wahrheitstreue von Transaktionen):
Falls lokal akzeptiert, werden Transaktionen auch global akzeptiert.
Sonst werden sie abgelehnt.
- v)* Bewahrung lokaler Wahrheitstreue:
Ein Benutzer bewertet seine Datenbank als
wahrheitsgetreues Abbild des Weltausschnitts.

Axiomatik der Vertraulichkeit in Datenbanken

Definition der Schutzeinheiten und des erwünschten Zustands:

- **Relation:** Existenz einer Beziehungsart
 - Relation aus dem Namensraum des Benutzers entfernen
- **Attribut:** Existenz eines Elements in einer Beziehungsart
 - Relation aus dem Namensraum des Benutzers entfernen
- **Tupel:** Nicht-/Existenz einer konkreten Beziehung
 - Tupel in/aus Relation einfügen/entfernen
- **Element:** Existenz eines Objektes
 - Element nicht aus dem Namensraum des Benutzers auswählen

Umsetzung:

- **Strukturierte Namensräume:** Relationen / Attribute / Elemente
- **Aliase:** Tupel

AS-DB, Alias-basierte sichere Datenbank: Aufbau

Offene Datenbank: unverfälschter Zustand

Je Benutzer:

- lokale Datenbank: verfälschter Zustand gemäß Vertraulichkeitsforderungen
- Änderungsrechte auf Tupeln = Kompetenzen
- Wissen über Tupel = Vertraulichkeit sinnlos

Explizite Vertraulichkeitsforderungen:

- Relation und Attribut: Verfälschung des Schemas
- Tupel: Einsatz falscher Tupel
- Element: Wahl eines unzugänglichen Namensraums

Verwaltung der Verfälschungen der Tupel: Alias-Log

- Beziehung: vertrauliches Tupel \Leftrightarrow seine Aliase
- Beziehung zwischen allen Einträgen

Beispiel: Umsetzung einer Vertraulichkeitsforderung

- Relation $p(X, Y)$ mit Primärschlüssel auf X :

$$\forall XYZ: \neg p(Z, X) \vee \neg p(Z, Y) \vee eq(X, Y)$$

- Tupel $p(a, b)$ wird in Datenbank M eingefügt und soll vor Benutzern der Datenbank N vertraulich gehalten werden:

$$M(p(a, b)) = \text{true} \text{ gilt und } N(p(a, b)) = \text{false} \text{ soll gelten}$$

- Ausführung von $TA = (p(a, c), \text{INSERT})$ auf N führt zu Problemen:

	$p(a, b)$	$p(a, c)$	$eq(b, c)$	$\neg p(a, b) \vee \neg p(a, c) \vee eq(b, c)$	
M	true	false	false	true	O.K.
N	false	false	false	true	
$TA = (p(a, c), \text{INSERT})$ auf N					
M	true	true	false	false	Gefahr!
N	false	true	false	true	

Schlußbetrachtung

Forschung an der Universität Bonn:

- Vertraulichkeit und Integrität in Informationssystemen
- Schutzmechanismen gegen Trojanische Pferde
- Sicheres Arbeiten im WWW

