

IT Sicherheit I v3, Ü1

Prof. Claudia Eckert

- **Sprechstunden:** Di 13:00 - 14:00 in S202/D204, sowie nach den Vorlesungen u. nach Vereinbarung
- Büro: Hochschulstr. 10, S202/D204 Telefon: 16 6591
- E-Mail: eckert@sec.informatik.tu-darmstadt.de
- **Sekretariat,** Frau Walter, Telefon 16 6592
Raum: S202/D204
- **Übungsleitung und –Organisation:**
Dipl. Inf. Lars Fischer
E-Mail: fischer@sec.informatik.tu-darmstadt.de



Materialien zur Vorlesung:

- (1) **Folien:** im Netz vorlesungsbegleitend vorab unter:
<http://www.sec.informatik.tu-darmstadt.de/de/lehre/WS04-05/itsec1>
- (2) **Buch:** IT-Sicherheit, 3-te Auflage (2004), Oldenbourg-Verlag
Studienausgabe erscheint Anfang November!
- (3) **Übungsblätter** und ggf Begleitmaterialien

Prüfungen

- Einordnung der Lehrveranstaltung in den
Prüfungsbereich: Informatik II,
- FAQ zu Prüfungsangelegenheiten:
<http://www.sec.informatik.tu-darmstadt.de>



Prüfungen (cont.)

- semesterbegleitende Prüfung:
schriftliche Klausur am Semesterende,
keine Semestralprüfung sondern **Prüfung für Master- bzw. Diplomstudiengang** (alt)
- **IT-Sicherheitszertifikat**: Vorlesung als geprüfter Bestandteil
Für Studierende: erfolgreiche Teilnahme an Klausur
Für Weiterbilder: mündliche Prüfungen nach Absprache
- **Sonstige Leistungsnachweise** über die Veranstaltung
für Studierende **anderer Fachrichtungen**: idR über Klausur
Abweichungen nur nach vorheriger persönlicher Absprache



Einordnung und Ziele der Vorlesung(en) IT – Sicherheit I und II

Ziele:

- Kenntnisse über wichtigste **Sicherheitsprobleme** in heutigen IT-Infrastrukturen und deren **Ursachen**
- Überblick über die gängigen **Techniken zur Erhöhung** der IT-Sicherheit
- **Einschätzung der Grenzen** der im Einsatz befindlichen Techniken (Vorteile, Nachteile)
- **Modelle und Methoden** zur systematischen Konstruktion sicherer Systeme bzw. zur Erstellung von Sicherheitskonzepten
- Verdeutlichung der Konzepte und Methoden anhand von **Fallbeispielen** und **konkreten Protokollen**





Berührungspunkte mit u.a.:

- Betriebssystemen, Kommunikationsnetzen
- Datenbanken, Verteilten Anwendungen
- Software-Engineering-Methoden

Ergänzende Lehrveranstaltungen im WS04/05:

- Einführung in die Kryptographie: J. Buchmann
- Kommunikationsnetze II: R. Steinmetz
- Dependability I: Reliable Distributed Systems: N. Suri
- Beweisbar sichere Kryptographie: T. Takagi
- Virtuelle Private Netze (VPN): Böhmer
- Netzwerksicherheit: Wolthusen



Literatur u.a.

C. Eckert *IT – Sicherheit*, 3-te Auflage, Oldenbourg-Verlag, 2004,
Studentenversion (reduzierter Umfang ab Nov.)

R. Anderson, *Security Engineering*, John Wiley, 2001

M. Bishop, *Computer Security: Art and Science*, Addison-Wesley-Longman

J. Viega und G. McGraw, *Building Secure Software*, Addison-Wesley, 2002

G. Hoglund und G. McGraw, *Exploiting Software, how to break code*,
Addison Wesley, 2004

S. Garfinkel und G. Spafford *Practical Unix & Internet Security*, O'Reilly

B. Schneier *Secrets & Lies*, dpunkt-Verlag, 2000

J. Buchmann *Einführung in die Kryptographie*, Springer, 2001

W. Stallings *Cryptography and Network Security*, 2nd Edition

E.D. Zwicky, S. Cooper, B. Chapman *Building Internet Firewalls*, O'Reilly



Inhaltsverzeichnis – Grobüberblick (IT-Sicherheit I und II)

1. Einleitung

- Motivation (Bedrohungen, Angriffe)
- Begriffsklärung, Schutzziele etc.

2. Technische Grundlagen

- Internetprotokolle und -dienste:
wesentliche Funktionsweisen, Nachrichtenformate etc.
- Betriebssystem-Dienste:
grober Überblick über Aufbau und Funktionsweise



3. Standardsicherheitsprobleme heutiger Rechner

- Schwachstellen in Standard-Betriebssystemen und –Servern
 - Buffer-Overflows Angriffe
 - Viren, Würmer, Trojanische Pferde, mobiler Code:
- Sicherheitsprobleme der Internet-Protokolle, Beispiele
- Sicherheitsprobleme bei drahtloser Kommunikation

4. Security Engineering

- Sicherheitsprozess: Überblick über die Phasen
- Schutzbedarfsfeststellung, Bedrohungs- und Risikoanalyse
- Sicherheit im laufenden Betrieb (Beispiel eSI-Tool)



5. Kryptographische Basis-Technologien

- Symmetrische und asymmetrische Verschlüsselung
 - Konzepte und Einsatzbereiche
 - Block- und Stromchiffre, Betriebs-Modi
 - Beispiele: AES und RSA-Verfahren
- Schlüsselaustauschverfahren, Diffie-Hellman-Verfahren
- Kryptographische Hashfunktionen, MACs, Digitale Signatur

6. Sichere Basis und Sicherheitsmanagement

- Schlüsselerzeugung, -aufbewahrung, -Recovery
- Smartcards: Aufbau, Sicherheitsdienste, Einsatzbereiche
- Trusted Computing: TPM



7. Authentifikation: und Trust Management

- Passwortverfahren: klassisch und One-time Passworte
- Challenge-Response und Single-Sign On
- Biometrische Verfahren: Funktionsweisen und Grenzen
- Trust-Management: Zertifikate, PKI, Reputationssysteme

8. Identitäts- und Rechtemanagement

- Sicherheits-Modelle: u.a. Rollenmodelle, Bell LaPadula
- Rechteverwaltung: Policies, Zugriffskontrolllisten, Capabilities
- Beispiele: Zugriffskontrolle in UNIX/Linux und Windows
- Verschlüsselnde Dateisysteme
- DRM und Nutzungskontrollen



9. Firewall-Technologie

- Firewall-Konzepte: Packetfilter, Application-level Proxies
- Firewall-Architekturen, Einsatzbeispiele und Grenzen

10. Sicherheitsprotokolle

- Virtuelle private Netze (VPNs)
- Kommunikationsebene: IPsec und SSL /TLS
- Anwendungsebene: Sichere E-Mail, Sichere Web-Services

11. Sicherheit in mobilen und drahtlosen Netzen

- GSM/GPRS/UMTS
- WLAN und Bluetooth

12. Sicherheitsbewertung

Common Criteria mit Protection Profiles



TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 1

Kapitel 1 Einführung

1.1 Motivation

heutige Systeme: vernetzt, mobil, heterogen

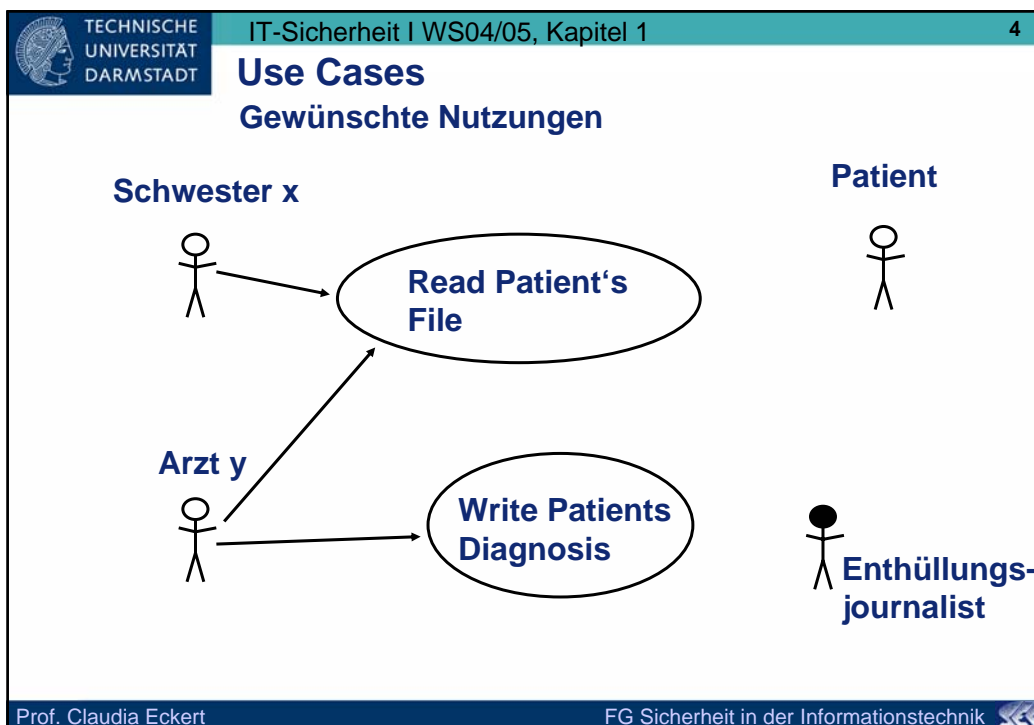
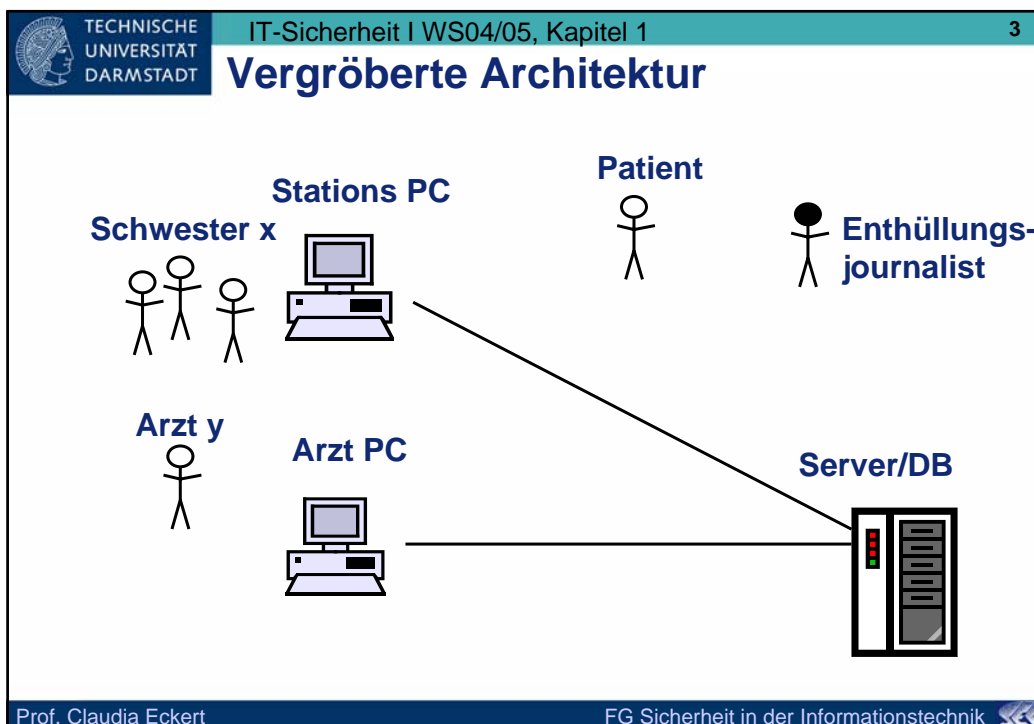
Prof. Claudia Eckert
FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 2

1.2. Beispiel-Szenario: Krankenhaus

<p>Akteure</p> <ul style="list-style-type: none"> •Ärzte •Schwestern •Patienten •Besucher •Laboranten •... 	<p>Begleitendes Beispiel</p>	<p>Angreifer</p> <ul style="list-style-type: none"> •Krimineller Hacker •\$TERRORIST •Versicherungsbetrüger •Teeny-Fan •Enthüllungsjournalist •...
<p>Kommunikationspartner</p> <ul style="list-style-type: none"> •Versicherungen •Banken •Andere Krankenhäuser •... 	<p>Orte</p> <ul style="list-style-type: none"> •Kliniken •Stationen •Verwaltung •... 	

Prof. Claudia Eckert
FG Sicherheit in der Informationstechnik



TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 5

Mis-Use Cases

Unerwünschte, zu verhindernde Nutzungen

Schwester x

Arzt y

Patient

Enthüllungsjournalist

Read Patient's File

Write Patients Diagnosis

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 6

Angriffsmöglichkeiten

Schwester x

Arzt y

Patient

Enthüllungsjournalist

Read Patient's File

Write Patients Diagnosis

- Spoofing
- Spionieren
- Social Engineering
- Virus
- Buffer Overflow
- Sniffing
- Trojaner

Systematische Untersuchung der
Angriffswege z.B. Angriffsbäume:
Verfeinerung der Ziele durch
immer genauere Vorbedingungen

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 7

Bsp.: Angriffsbaum A. liest Patientenakte

1. Spoofing (OR)
2. Spionieren (OR)
3. Social Engineering (OR)
4. Virus (OR)
5. Buffer Overflow (OR)
 1. Code einschleusen (AND)
 1. Zugang zu Endgerät (OR)
 1. Fehlende/fehlerhafte Zugangssicherheit (OR)
 2. Via offener Netzchnittstellen (OR)
 1. Mobiles Gerät an lokalem Netz (OR)
 1. Fehlerhafte/fehlende Geräteauthentisierung
 2. Internetzugriff (OR)
 2. Programm ist anfällig (AND)
 1. Fehlerhafte Eingaberoutine (AND)
 1. Programmierfehler z.B.: Fehlendes Bound-Checking
 2. Fehlende/fehlerhafte BO. Detection (AND)
6. Sniffing (OR)
7. Trojaner (OR)

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 8

Hauptursachen für heutige Sicherheitsprobleme

- **Design-Mängel:** u.a. fehlende Kontrollen bei Zugriffen,
 - fehlende, schwache Authentifikation (u.a. Passworte),
 - Vergabe zu vieler Rechte, allmächtiger Superuser
- **Implementierungsfehler:** u.a. Pufferüberläufe,
 - Schwacher Mechanismen (z.B. schw. Zufallszahlengen.)
 - direkte Zugriffe ohne Kontrollen (z.B. privilegierter Modus)
- **Administrative Fehler:** u.a. Kennung ohne Passwort,
 - falsch konfigurierte Firewalls, zu viele offene Ports,
- **Fehlende Awareness,** mangelhafte Kenntnisse
 - u.a. Root-Passwort über Jahre unverändert
 - Social Engineering: Weitergabe von Passwörtern

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 9

Fazit

- Für das Verständnis der Sicherheitsprobleme sind zumindest rudimentäre Kenntnisse über **Betriebssysteme und Kommunikationsprotokolle (Netze)** notwendig
- Viele Probleme durch ‚Social Engineering‘: Kenntnisse der
 - **Ursachen** von Problemen, **Zusammenhänge** und
 - **Auswirkungen** notwendig
- Viele Probleme durch Unkenntnis der tatsächlichen Güte und Grenzen **vorhandener Sicherheitskonzepte**, z.B. „Wir haben eine Firewall, also sind wir abgesichert.“
- Viele Probleme durch mangelhafte **Methodik, Modelle**

Vorlesung soll Verständnis vermitteln!

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 10

Herausforderungen für die IT-Sicherheit? Quo vadis?

1.3 Einige **Trends in der Technologie-Entwicklung** und ihre Bedeutung für die **IT-Sicherheit (von morgen und übermorgen)**

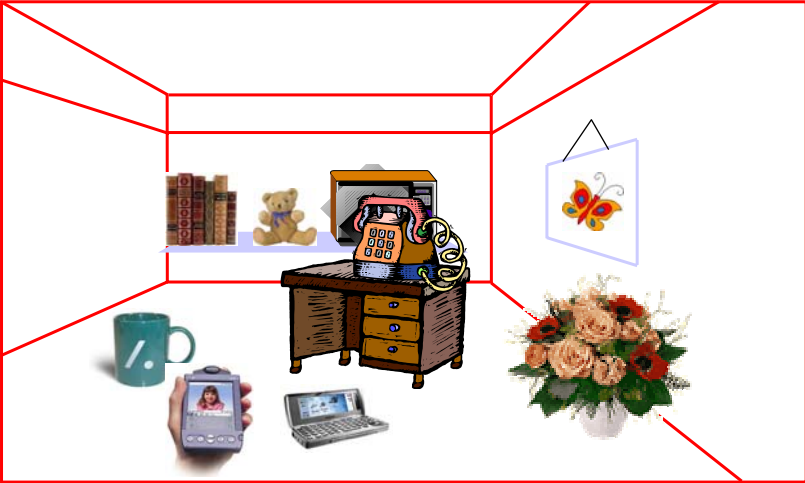
(1) Von zentralen raumfüllenden Rechnern...



Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 11

...zu **allgegenwärtigen (ubiquitären)** Rechnern, die in den Gegenständen des täglichen Lebens, in der Umgebung verschwinden



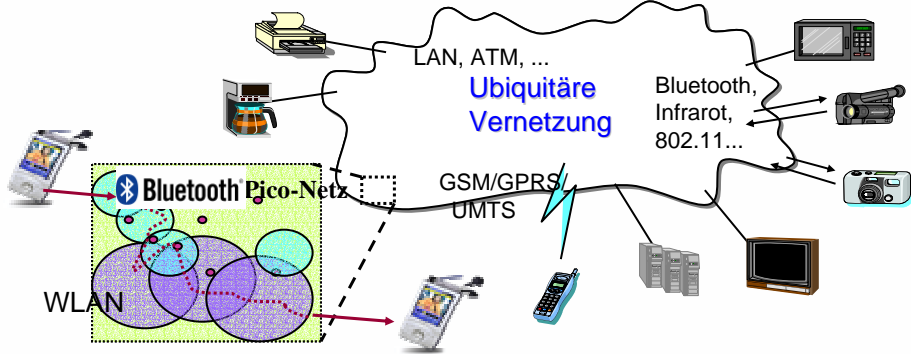
Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 12

Trend

(2) **Ubiquitäre Netze:** jeder mit jedem, jederzeit, in jeder Form

- heterog: Festnetz, drahtlos, mobil, Funk, Sensor, ...
- unterschiedliche Bandbreite, Kosten, Abdeckung, QoS, ...



Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 13

Trend: (3) Zusammenwachsen von Infrastrukturen

- Unternehmensübergreifende digitale Geschäftsprozesse
- Ablaufoptimierungen: u.a. Fernsteuerung, Überwachung von Produktions- und Steuerungsanlagen, Autos, Flugzeugen, ...

Steuerung von Signalanlagen, ...

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 14

Fernwartung, Monitoring

Steuerung der Energieversorgungsanlagen

Steuerung von Signalanlagen

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

Datenban

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 15

Trend: (4) Mobilität, Heterogenität, Dynamik (Erweiterbarkeit)

- mobile Geräte, Personen, Netze, Dokumente/Content, ...
- heterogene Geräte/Systeme:
Mix aus ressourcenstark und –schwach

Heimnetz Ad-hoc Netz Gastnetz

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik


TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 16

Konsequenzen: u.a.

- vermehrte **Maschine-zu-Maschine** Kommunikation, ohne Interaktionsmöglichkeiten durch den Menschen
- Vermehren der **Angriffspunkte** (Personen, Anlagen, ..)
- Anstieg an **Angriffen** (u.a. durch automatisierte Tools),
- Ausweitung **pot. Angreifer**: Kriminelle, Wirtschaftsspionage, ...
- Erhöhtes **Risiko** durch Abhängigkeiten (Stromausfall!)
- keine definierten **Grenzen**: Innen und Außen verschwimmt
z.B. PDA, der sowohl im internen Netz als auch außen agiert
- Dezentrale **Kontrollen** (z.B. Peer-to-Peer, GRID, Content),
Probleme der Vertrauenswürdigkeit, Accounting, Billing etc.
- Vielzahl von **Identitäten**: Privacy-Probleme, Usability

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

Dat

 TECHNISCHE UNIVERSITÄT DARMSTADT


IT-Sicherheit I WS04/05, Kapitel 1 17


1.4 Herausforderungen an F&E im Bereich IT-Sicherheit in den nächsten Jahren

Bericht der Computing Research Agency, siehe: www.cra.org:
Entwicklung **vertrauenswürdiger IT-Systeme** als eine der **fünf großen Herausforderungen** für die IT-Technologie bis 2013

Key Challenges im Bereich IT – Sicherheit

- (Weiter)-Entwicklung von **Konzepten u. Methoden**, um IT-Systeme vertrauenswürdig, sicher und nutzbar zu gestalten.
- Konstruktion von **sicherer Software**, Sicherheit per Design, Ergänzung bzw. Neuentwicklung von Konzepten, Analysen, ...

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik 

 TECHNISCHE UNIVERSITÄT DARMSTADT


IT-Sicherheit I WS04/05, Kapitel 1 18

Key Challenges im Bereich IT – Sicherheit (cont.)

- (Weiter)-Entwicklung von **Auditing, Monitoring-Tools**: statische (z.B. Schwachstellen) und dynamische Analysen
- Konzepte zur besseren **Nutzbarkeit, Handhabbarkeit**
- Konzepte für **Selbst-organisierende** Systeme: Einhaltung von Regeln prüfen, System automatisch rekonfigurieren

These: Paradigmenwechsel sind notwendig

- Von **Perimeterschutz** zur **integrierten Systemsicherheit** gestaffelte Abwehrmaßnahmen, nicht nur ad-on (z.B. Firewall)
- von **zentralen Zugriffskontrollen** durch Server etc. zu **dezentralen Nutzungskontrollen** z.B. Kontrolle von digitalem Content (MP3, Dokumente etc.)

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik 

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 19

Noch einmal zur Verdeutlichung:
Perimeterschutz meist durch **Firewall** als erste Hürde:
alle Zugriffe sind an einem zentralen Punkt kontrollierbar
Aber: mobile Rechner, mobiler Code:
Informationsflüsse **an dem zentralen Kontrollpunkt vorbei!**

Heimatnetz Ad-hoc Netz Gastnetz

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 20

These: Paradigmenwechsel sind notwendig (cont.)

- vom Ziel der ‚**perfekten**‘ Sicherheit (nothing bad will happen) zu **angemessener und ‚managbarer‘** Sicherheit
Angriffe sind unvermeidbar, System muss damit umgehen!
- von der Betrachtung der Sicherheit als ein **Produkt** zur Sichtweise der Sicherheit als ein **dynamischer Prozess**
 - **Vermeidung** von Schwachstellen durch Design
 - **Angriffserkennung, -bewertung** (Entscheidungsfindung)
 - **Angriffsabwehr** (Handeln, Reagieren)
 - **Reduktion von Schwachstellen** (u.a. Rekonfiguration)

Aber: bevor man Neues entwickelt muss man wissen:

- **welche** Konzepte und Methoden, Verfahren **gibt es**
- **wie** gut/schlecht sind sie, worin liegt Verbesserungspotential

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 21

1.5 Gegenstand der Vorlesung(en):

Basis:
Technische Grundlagen von BS und Protokollen

- **Probleme:**
Bedrohungen und Funktionsweisen von Angriffen
- **Lösungen:**
Realisierungskonzepte, -Dienste, Methoden, Modelle
- **Bewertung:**
Stärke und Schwächen der verschiedenen Konzepte und Verfahren analysieren, diskutieren
- **Methoden und Modelle:**
systematische Vorgehensweisen, Ansätze

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 1 22

Das Sicherheits-Gebäude:

Plan, Qualitätssicherung

Angriffe:
Viren, Trojaner, Buffer-Overflows, Sniffen, DoS, Spoofen, ...

Modelle, Sicherheitsprozess, Zertifizierung, ...

Dienste/Protokolle: Kerberos, IPsec, SSL, ...

ACLs Schutzdomäne

Firewall-Konzepte

Verschlüsselung, Hashverfahren, Signaturen, ...

Passworte, Biometrie, Smartcards

Bausteine

Basis: Betriebssystem-Konzepte, -Strukturen, -Modi, Protokoll-Eigenschaften

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 1

Kapitel 2 Begriffe und grundlegende Definitionen

2.1 Trusted Systems

Informell: Was ist Vertrauen (trust)?!

Trust is the extend to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible!

Unter einem **Trusted System** verstehen wir ein System, dem man vertraut, dass es spezifizierte Dienste, bzw. eine spezif. Funktionalität **auch beim Auftreten von Störungen oder Fehlern korrekt** und **zuverlässig erbringt**.

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 2

Ursache für Störungen/Fehler können sein:

- **unabsichtliche** Fehlnutzungen, aber insb. **gezielte** Störversuche: meist von außen initiiert: **Security-Probleme**
- **fehlerhafte unzuverlässige funktionale Abläufe** (operational) : meist von innen, durch das System: **Dependability-Probleme**

Vertrauen

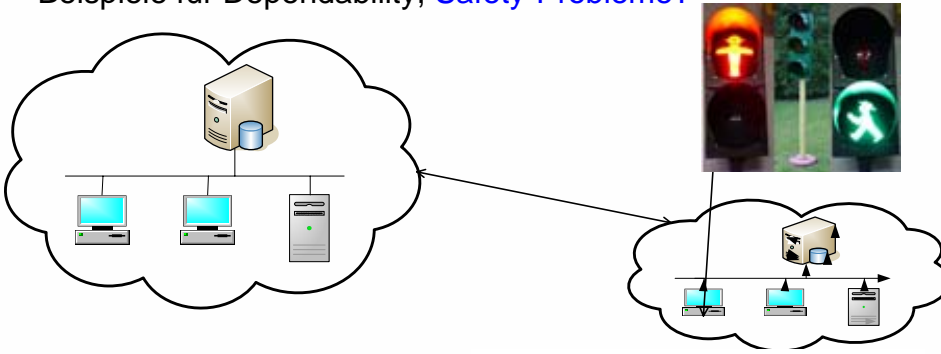
- ist schwer meßbar und damit auch **schwer quantifizierbar**,
- deshalb wird Vertrauen häufig **qualifizierend** erfasst, Bsp.?
- ist **abhängig von Kontexten** und **Kenntnissen**: Beispiele?
- erfordert eine **Einschätzung der Bedrohungs- und Risikolage**
- erfordert eine **Festlegung der Ziele**

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 3

Beispiel-Szenario: Vernetztes IT-System

- Funktionalität: u.a. korrektes Schalten der Signale
- Beispiele für **Security-Probleme?**
- Beispiele für Dependability, **Safety-Probleme?**



Steuerung und Überwachung von Signal-Anlagen

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 4

Security und Dependability:
Abgrenzungen und Gemeinsamkeiten

Security: Erkennen u. Abwehr insbes. von **gezielten Angriffen:**
nach ISO (International Standards Organization):
Minimierung der Verwundbarkeit von Werten u. Ressourcen

- Spezifikation der **ungewünschten Aktionen**
- **Nothing bad can happen**
- **Bad** ist aber nicht unbedingt eine fehlerhafte Funktion, sondern z.B. eine unerwünschte Informations-Weitergabe
- gezielte Angriffe, ‚intelligente‘ Angreifer
- **Beispiele** für Angriffe auf die IT-Sicherheit des Systems:
Sniffen, Spoofen, Hijacken, ... **Was bedeutet das?**

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

Datenba

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 5

Dependability, Safety: Erkennen und Abwehr von Störungen, die die korrekte Funktionalität, die Betriebssicherheit eines Systems beeinträchtigen

- Spezifikation der **gewünschten Funktionalität**
- Erkennen von Abweichungen vom gewünschten Verhalten
- Störungen treten meist nicht gezielt sondern zufällig auf
- Beispiele für Safety-Probleme:
 - <http://www.heise.de/newsticker/meldung/51351>
Software-Probleme führten zum Totalausfall des Funksystems im Flugkontrollzentrum von Palmdale nahe Los Angeles. Auslöser der Panne war ein Designfehler im Windows-2000-Server-Netz des Los Angeles Air Route Traffic Control Center.
- weitere?

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 6

Gemeinsamkeiten zwischen Security, Dependability: (u.a.)

- Formulierung von gewünschtem Verhalten, um Abweichungen zu erkennen:
Modelle, Regelwerke sind notwendig, um dies zu beschreiben
Beispiele für Modelle? für Regelwerke?
- Erkennung von Abweichungen erfordert:
 - Erfassung von Zustandsinformationen (z.B. Wer ist aktiv) über geeignete **Datenstrukturen, Sensoren** etc.
Beispiele für solche Datenstrukturen?
Beispiele für Sensoren?

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 7

- Erkennung von Abweichungen erfordert:
 - **Algorithmen**, um zu erkennen, dass eine Abweichung vorliegt, z.B. Filterregeln, Identifikationsüberprüfung, Weitere Beispiele?
- Erkennung und Behandlung von Fehlersituationen erfordert
 - **Systemarchitekturen**, die diese Dienste anbieten, z.B. integrierte Passwortkontrolle, redundante Komponenten

Trusted Systems: Kanonik-Vorlesungen und Vertiefungen
Kennen lernen wichtiger Modelle, Methoden, Konzepte und Architekturen

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 8

Trust im **Security-Umfeld**:
idR abhängig von der **Identität** und Einhaltung von **Regelwerken**

Trust im **Safety-Umfeld**:
idR abhängig von **korrektem Verhalten** eines Systems

Unterschiede zwischen Security und Safety:

- **unterschiedliche Teilziele** mit **unterschiedlichen**
- **Methoden, Modelle, Verfahren** zur Erreichung der Ziele

Vorlesung IT-Sicherheit: Konzentration auf den Bereich Security!

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 9

2.2. Grundlegende Begriffe

Verdeutlichen an Krankenhaus-Szenario

Ziele: Schaffen eines einheitlichen Begriffsrahmens,
wird in den folgenden Unterabschnitten fortgesetzt

- **Rechensysteme**
 - geschlossene oder **offene** (verteilte, vernetzte),
 - **dynamische** Systeme mit der Fähigkeit zur Speicherung und Verarbeitung von **Information**
- **Information** ist abstrakt, wird in einem Rechensystem durch **Objekte** (Datenobjekte) konkretisiert, erfordert Interpretation
- **Objekte/Informationen:** die zu schützenden Einheiten (Asset)
- **Subjekte** (Principals): aktive Einheiten (im Benutzer- oder Systemauftrag), die auf Objekte/Informationen **zugreifen**

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 10

Einige wichtige Aufgaben sicherer Systeme:

Aufgabe: Zugriff auf Objekte/Informationen **beschränken**

- **Rechte:** zur Nutzung von Objekten, Rechte an Information

Aufgabe: nur **autorisierte** Zugriffe zulassen und ermöglichen

- **Regelwerk (Policy)** für Berechtigungen/Verbote
- Verfügbarkeit von Diensten sicherstellen

Aufgabe: Zugriffsbeschränkungen **kontrollieren**

- **Schutzmechanismen** (z.B. Kryptographie) und
- **Sicherheitsdienste** und **-Protokolle**

Beispiel-Szenario: Krankenhaus-Szenario

- Objekte? Subjekte? Rechte?
- Mögliche Schutzkonzepte? Notw. Sicherheitsdienste

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 11

2.3. Schutzziele

Basis: funktionale Korrektheit wird vorausgesetzt
Schutzziele sind das Gewährleisten der

- (1) **(Daten)Integrität** (engl. integrity)
- (2) **(Informations)Vertraulichkeit** (engl. confidentiality)
- (3) **Verfügbarkeit** (engl. availability)
- (4) **Verbindlichkeit** (engl. accountability)
- (5) **Authentizität** (engl. authenticity)
- (6) **Privatheit** (engl. privacy)

idR wird eine Kombination mehrerer Schutzziele gefordert
z.B. Homebanking: **Integrität** des Kontostandes, **Vertraulichkeit** der Kontodaten, **Verfügbarkeit** der Bankdienste,

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 12

Die Schutzziele im Einzelnen:

(1) Datenintegrität
Schutz vor **unautorisierter** und **unbemerktter** Daten-**Modifikation**
Angriffe: Beispiele?

Maßnahmen: Festlegen und kontrollieren von Regeln:
Wer (Subjekt) darf **Was** (Rechte) unter **Welchen** Bedingungen
Beispiel für Regel (z.B. Krankenhaus-Szenario)?

Schutzkonzepte: u.a.

- Objekt-bezogene Zugriffskontrolllisten mit Zugriffsrechten,
- Berechnen von eindeutigen Prüfsummen von Objekten,
- Firewalls zum Filtern von Datenpaketen, Gewaltenteilung, ...

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 13

Die Schutzziele im Einzelnen (cont.):

(2) Informationsvertraulichkeit

Schutz vor unautorisierter **Informationsgewinnung**
Angriffe: Beispiel?

Maßnahmen: Festlegen u. kontrollieren von Informationsflüssen:
Wer darf auf **welche** Informationen zugreifen, bzw. davon Kenntnis erlangen
Beispiel eines berechtigten Informationsflusses?

Schutzkonzepte: u.a.

- kryptografische Verfahren, Verschlüsselte Dateien etc.
- Sicherheitsklassifikation (labeling) von Objekten/Subjekte

Beispiel für Labeling?

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 14

Wichtig: Kontrolle der Objektzugriffe \neq Kontrolle der Information!

Beispiel-Szenario:

Forderung: Subjekt **Joe** darf keine Kenntnis über vertrauliche Informationen erlangen, Bill aber sehr wohl!

Annahme:
Subjekt Chef schreibt vertrauliche Informationen in Datei_1 (Objekt),
Policy: Joe hat kein Recht, auf Datei_1 zuzugreifen!
Gut, aber reicht nicht!

Annahme:
Bill liest die Information aus Datei_1 und schreibt sie in Datei_2
 \Rightarrow **Forderung verletzt**

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 15

Informationskanäle:

Legitimer Kanal: für Informationsaustausch vorgesehen, Bsp?

- **Speicher Kanal:** indirekte Kommunikation, Beispiel?
- **Verdeckter Kanal** (covert channel):
nicht für Informationsaustausch vorgesehen

Problem: Verhindern verdeckter Kanäle ist **sehr schwer**
Meist nur Beschränkung der Bandbreite möglicher Kanäle

Beispiel: Kanal über Datei-Locking-Konzepte

Das Diagramm zeigt den zeitlichen Ablauf zwischen einem Server und einem Kollaborator. Der Server lockt abwechselnd Dateien für '1' (schraffiert) und '0' (weiß). Die Bitstream, der über den Kanal gesendet wird, ist '1 1 0 1 0 1 0 0'. Ein roter Kreis unterstreicht den Bitstream.

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 16

Die Schutzziele im Einzelnen (cont.):

(3) Verfügbarkeit
Schutz vor unbefugter **Beeinträchtigung** der Funktionalität von Komponenten, Diensten etc.

Angriffe: Beispiele?

Maßnahmen: u.a.

- Überwachung der Systemnutzung, Monitoring,
- Überwachen des Ressourcenverbrauchs, Accounting

Schutzkonzepte: u.a.

- Festlegen von Quotas (z.B. max. Anzahl von Prozessen)
- Protokollieren von Aktionen, Reaktion auf unübliche Aktionsmuster (z.B. Blockieren von Datenpaketen)

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 17

Die Schutzziele im Einzelnen (cont.):

(4) Verbindlichkeit
kein unzulässiges **Abstreiten** durchgeführter Handlungen
Notwendig **beispielsweise** für:

- Abschließen von elektronischen Kaufverträgen
- Digital unterschriebene Mahnbriefe

Angriffe: Beispiele?

Maßnahmen: u.a.

- Nachweise erstellen, Audits, Log-Bücher, ...
- Beweissicherungen durchführen, (ggf. bis hin zu Forensik)

Schutzkonzepte: u.a.

- digitale/elektronische Unterschriften, Signaturen
- Einrichten vertrauenswürdiger Notariatsstellen (Trust Center)

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 18

Die Schutzziele im Einzelnen (cont.):

(5) Authentizität
Nachweis der **Echtheit** und **Glaubwürdigkeit der Identität**
eines Objekts/Subjekts

Angriffe: Beispiele?

Maßnahmen:

- Eindeutige Identifikation von Subjekten und Objekten und
- Nachweis der Korrektheit der Identität

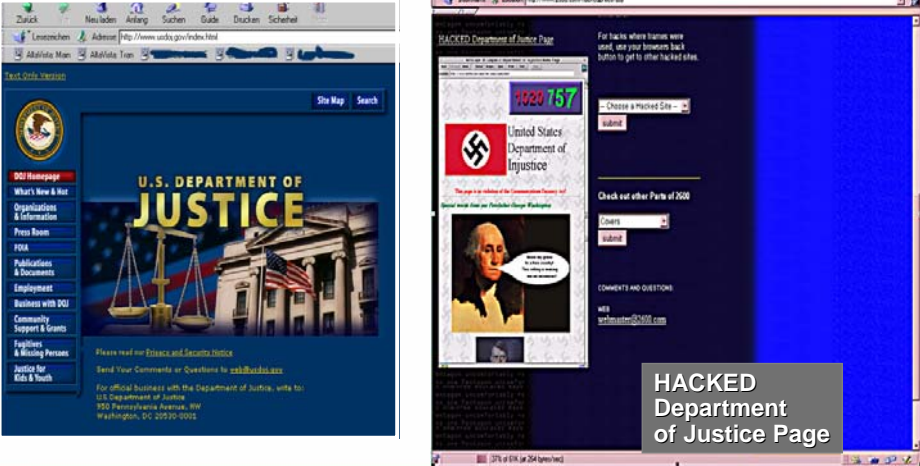
Schutzkonzepte: u.a.

- Passworte, PINs, Biometrie
- Smartcards, Security-Tokens
- Signierter Code (was für eine Authentizität damit erreichbar?)

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 19

**Bsp.: Spoofen, Fälschen einer Homepage,
Departement of Justice: Nazi-Seite als Department of Injustice**



Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 20

Die Schutzziele im Einzelnen (cont.):

(6) Privatheit
Schutz der **personenbezogenen Daten**, Schutz der Privatsphäre,
Informationelles Selbstbestimmungsrecht
Angriffe: Beispiele?

Maßnahmen: u.a.

- Verschleiern, Anonymisieren von Identitätsdaten
- Einhaltung rechtlicher Rahmenbedingungen (u. a. BDSG)

Schutzkonzepte: u.a.

- Pseudonyme oder Verschlüsseln der Identitäten (Tunneln)

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 21

Schutzziele sind **abhängig** von der zu schützenden Anwendung bzw. den Funktionen des zu schützenden Systems

Beispiel-Szenarien:

- (1) Krankenhaus-Umfeld: Schutzziele?
- (2) Web-Services, Content-Anbieter und Nutzer: Schutzziele?

2.4 Sicherheitsstrategie (Security Policy)

- Festlegen der **Schutzziele** und der Menge von technischen und organisatorischen **Regeln** und **Verhaltensrichtlinien**
- Maßnahmen zur Gewährleistung der Schutzziele, um ein angestrebtes Sicherheitsniveau zu erzielen
- Festlegen von **Verantwortlichkeiten** und **Rollen**

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 22

Beispiel: Passwort-Policy (Auszug)

All system-level passwords (e.g., root, enable, NT admin, application administration accounts, etc.) **must be changed** on at least a quarterly basis.

- All production system-level passwords **must be part** of the InfoSec administered global password management database.
- All user-level passwords (e.g., email, web, desktop computer, etc.) **must be changed** at least every six months. The recommended change interval is every four months.
- User accounts that have system-level privileges granted through group memberships or programs such as "sudo" **must have a unique** password from all other accounts held by that user.
- Passwords **must not be inserted** into email messages or other forms of electronic communication.
- ...

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 23

Sicherheitsstrategien

i.d.R. in Form von Text-Dokumenten festgelegt (siehe Folie 23)

Probleme: u.a.

- ggf unpräzise, missverständlich, inkonsistent
- Vergleich/Abgleich von Policies ist sehr schwierig (z.B. kooperatives Arbeiten, Austausch von Dokumenten) jeder hat eigene Policy: was ist der gemeinsame Nenner?!

Formalisierung der Policies ist wünschenswert,

- bislang nur wenige Ansätze hierfür
- z.B. **SAML** (Security Assertion and Markup Language) für Web-Service Security entwickelt (OASIS) Assertions (u.a. Authentifikation): wer, wann, womit

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 24

Authentication Assertion Example

Bsp. SAML-Spezifikation

```
<saml:Assertion
  MajorVersion="1" MinorVersion="0"
  AssertionID="186CB370-5C81-4716-8F65-F0B4FC4B4A0B"
  Issuer="www.example.com"
  IssueInstant="2001-05-31T13:20:00-05:00">
  <saml:Conditions
    NotBefore="2001-05-31T13:20:00-05:00"
    NotAfter="2001-05-31T13:25:00-05:00"/>
  <saml:AuthenticationStatement
    AuthenticationMethod="password"
    AuthenticationInstant="2001-05-31T13:21:00-05:00">
  <saml:Subject>
    <saml:NameIdentifier>
      <SecurityDomain>"www.example.com"</SecurityDomain>
      <Name>"cn=Alice,co=example,ou=sales"</Name>
    </saml:NameIdentifier>
  </saml:Subject>
  </saml:AuthenticationStatement>
</saml:Assertion>
```

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 25

2.5 Bedrohungen, Angriffe, Risiken

Eine **Schwachstelle** (weakness) ist eine Schwäche des Systems oder ein Punkt, an dem das System **verwundbar** sein kann.

Beispiel?

Eine **Verwundbarkeit** (vulnerability) ist eine Schwachstelle des Systems, über die die Sicherheitsdienste des Systems **umgangen, geändert** oder **getäuscht** werden können.

Beispiel?

Bedrohungen (threat) nutzen eine oder mehrere Schwachstellen oder Verwundbarkeiten aus, um ein oder mehrere Schutzziele zu gefährden.

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 26

Bedrohungen ergeben sich durch **Angriffe** auf das System

Potentielle Angreifer

- Über 50 % aller bekannten Angriffe erfolgen durch **Mitarbeiter**
Ursachen: u.a. Nachlässigkeit, Bereicherung, Frust
- Social Engineering Angriffe
- Stark zunehmend: **Hacker/Cracker/Skript-Kiddies** Angriffe:
Systematische Zerstörung, 'Spieltrieb',
Unterschied: Hacker, Cracker, Script Kiddies?
- Bem.: > 90% aller Angriffe sind **Wiederholungstaten!**
Bedeutung?
- Stark zunehmend: **Wirtschaftsspionage**, geheimdienstliche Tätigkeiten, Kriminelle (z.B. Erpressung)

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 27

Abhängig von der Funktionalität und Einsatzumgebung des Systems besitzen Bedrohungen ein **unterschiedliches Gewicht**.

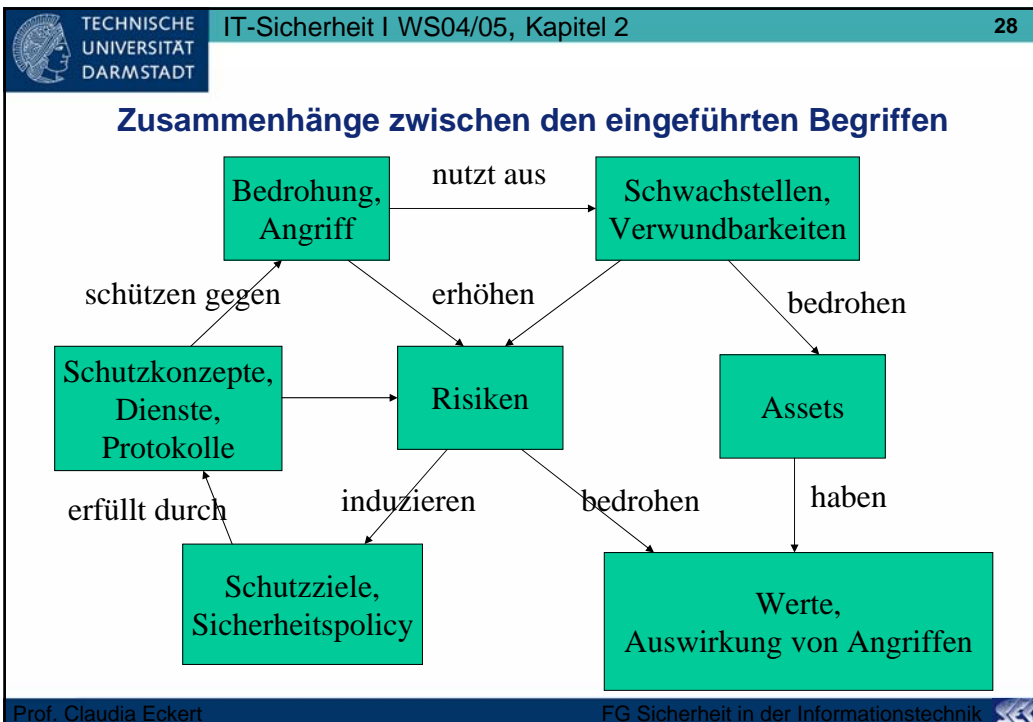
Beispiel-Szenarien:

- Bank, öffentliche Datenbank, Web-Dienste, Uni
- Krankenhausinformationssystem, ...

Zur Bestimmung der Gefährdungslage muss das **Risiko** bestimmt werden, das mit den Bedrohungen verbunden ist.

- Das **Risiko** (risk) einer Bedrohung ist die **Wahrscheinlichkeit** des **Eintritts** eines Schadensereignisses und die **Höhe** des potentiellen **Schadens**, der daraus resultieren kann.

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik



TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 29

Maßnahmen zur Gewährleistung der Schutzziele

sind sehr vielfältig, ggf. widersprüchlich: Bsp.?

- sind **drei groben Klassen** zuzuordnen:
 - Maßnahmen zu **Vermeidung, Verhinderung** von Angriffen
 - Maßnahmen zur **Erkennung** von Angriffen
 - Maßnahmen zur **Schadensbegrenzung** (mitigation)

Beispiele für die jeweilige Klasse?

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik

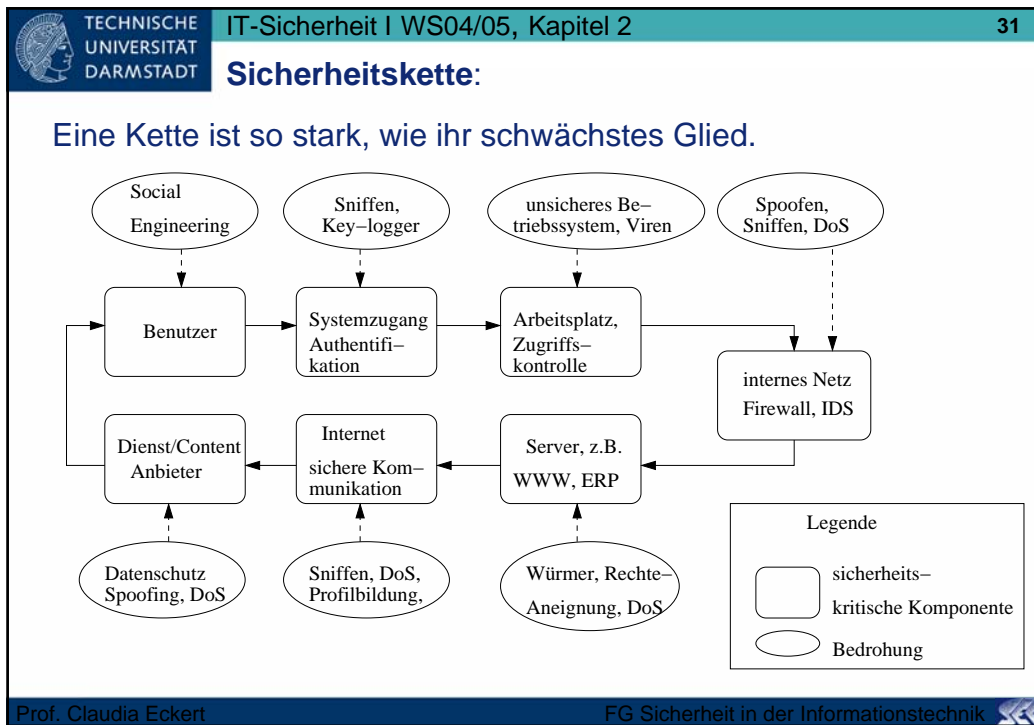
TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit I WS04/05, Kapitel 2 30

Erforderlich: Methoden/Verfahren zur Erhöhung der Sicherheit

Aber:

- ein **Absichern eines Teilbereichs ist unzureichend**,
 - z.B. Firewall zur Filterung des Datenverkehrs aus dem Internet; **aber**
was ist verschlüsselten Daten, die den Firewall tunneln?
 - z.B. verschlüsselte Kommunikation, **aber**
was ist mit den Zugriffen auf die unverschlüsselten Daten auf den Rechnern?
- Systemsicherheit erfordert einen **ganzheitlichen Ansatz**:
 - systematischer Security Engineering Prozess

Prof. Claudia Eckert FG Sicherheit in der Informationstechnik



TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 1

Kapitel 3 Technische Grundlagen

Ziel:
Überblick über einige Grundlagen aus den Bereichen **Betriebssysteme** und **Kommunikationsnetze**

3.1 Betriebssysteme

3.1.1 Aufgaben/Schichten eines Betriebssystems (siehe Folie 2)

- BS als **Vermittler** zwischen Hardware einschließlich E/A-Geräten und Anwendungen
- Betriebssystem bietet über **Systemschnittstelle (API) Dienste** an, um u.a. Prozesse zu erzeugen, Dateien zu verwalten, auf E/A Geräte zuzugreifen
- **Multiprogramming BS**: mehrere ausführbare Programme liegen gleichzeitig im physikalischen Speicher

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 2

Beispiel: Unix-Betriebssystem (Windows-Familie analog)

```
graph TD; Users[Users] -- User interface --> Utils[Standards utility programs  
(shell, editors, compilers etc)]; Utils -- Library interface --> Lib[Standard library  
(open, close, read, write, fork, etc)]; Lib -- System call interface --> OS[UNIX operating system  
(process management, memory management,  
the file system, I/O, etc)]; OS --- HW[Hardware  
(CPU, memory, disks, terminals, etc)];
```

Das Diagramm zeigt die Schichten eines Unix-Betriebssystems. Von oben nach unten sind dies: **Users** (User interface), **Standards utility programs (shell, editors, compilers etc)** (Library interface), **Standard library (open, close, read, write, fork, etc)** (System call interface), **UNIX operating system (process management, memory management, the file system, I/O, etc)** und **Hardware (CPU, memory, disks, terminals, etc)**. Rechts daneben sind die Modi **User mode** und **Kernel mode** mit den entsprechenden Übergängen dargestellt.

Prof. C. Eckert FG Sicherheit in der Informationstechnik

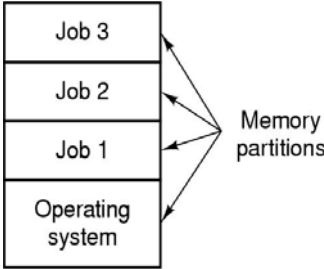
TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 3

3.1.2 Speicherlayout

- Speicherbereich für das BS und Programme

Problem:

- **Schutz der Programme** gegeneinander
- **Schutz des BS** vor den Programmen



Lösungen: Hardware-Unterstützung, Adressräume, Betriebsmodi

Hardware-unterstützte Trennung von BS u. Prozessen:

- durch **Register u. Adresslogik** (HDW)

Speicher-Adressraum:

- aufgeteilt in **System (kernel) Space**, niedrige Adressen
- Benutzer (**user) Space**, höhere Adressen

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 4

- Prozesse können in **zwei Betriebsmodi** ausgeführt werden

Betriebsmodi: Modus in der Hardware durch **1 Bit** festgehalten

- **Benutzer-** und **Systemmodus**,
- Prozesse im Benutzermodus laufen im User-Space,
- Prozesse im System-Modus laufen im Kernel-Space ab
- **kontrollierter Modus-Wechsel** über System-Aufrufe (Traps)

System-Modus (engl. kernel, system, supervisor mode):

- **Privilegierte Befehle** sind verfügbar,
u.a. Zugriff auf Hardware nur mit privilegierten Befehlen,
- **BS-Dienste** werden idR im Systemmodus ausgeführt: d.h. **besondere Berechtigungen** für diese Systemdienste!

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 5

Benutzer-Modus (engl. user mode)

- Anwendungsprozesse laufen im Benutzer-Modus ab,
- es sind **nur nicht privilegierte** Befehle verfügbar,
- Systemdienste, die privilegierte Befehle ausführen dürfen, sind über **Systemaufrufe** nutzbar (**siehe 3.1.5**)
- ein **Systemaufruf** führt zu einer **Unterbrechung** (Interrupt) mit
 - **kontrolliertem Einsprung** in den Betriebssystemkern (Trap)
 - und mit dem **Wechsel** in den **Kernel-Modus**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 6

Benutzer- und Systemadressraum

The diagram illustrates the flow of control and data between different layers of a system:

- User space:** Contains a **User program** (User process) and **User-Prozesse im Benutzer-Modus (auch einige Systemdienste)**.
- Kernel space:** Contains the **Rest of the operating system** and **System-Prozesse im System-Modus, privilegiert**. It also includes **Treiber-Code, privilegiert** (highlighted with a red oval), specifically **Printer driver**, **Camcorder driver**, and **CD-ROM driver**.
- Hardware:** Contains **Printer controller**, **Camcorder controller**, and **CD-ROM controller**.
- Devices:** Contains a printer, a camcorder, and a CD-ROM drive.

Arrows indicate the flow from the user program through the OS and drivers to the hardware controllers and finally to the physical devices. A red oval highlights the driver code, with the text **Treiber-Code, privilegiert** and **Konsequenzen?** next to it.

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 7

Sicherheitsprobleme: u.a.

- **Geräte-Treiber** sind Bestandteil des BS, (siehe Folie 6)
 - werden mit **besonderen Privilegien** ausgeführt
 - werden **beim Booten von der Festplatte in Speicher** als **ausführbare** Programme geladen
 - können aber auch **dynamisch** (Plug&Play) **nachgeladen** werden
 - **Vertrauenswürdigkeit** von Treiber-Software? **Viren?**

3.1.3 Virtuelle Adressräume (BS mit virtueller Speicherverwaltung)

- jeder Prozess besitzt einen **virtuellen Adressraum** (VA) idR 4GByte
- **Abstrahieren** von Größenbeschränkungen des realen Speichers
- VA ist eingeteilt in **Seiten oder in Segmente** mit Seiten, die werden in den Hauptspeicher idR on-demand von der Platte eingelagert

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 8

Aufteilung des virtuellen Prozess-Adressraumes:

4 GByte

Stacksegment

Heapsegment

Codesegment

0 GByte

lokale Variablen, Register, Übergabeparameter, Rücksprungadressen, Stack **wächst nach unten**

globale Daten und Konstanten, reservieren von Speicherbereichen, die zur Laufzeit mit *malloc()* angefordert werden. Heap kann zur Laufzeit **nach oben wachsen**

Maschinenbefehle des Programms, Größe, Inhalt fest, idR gegen Überschreiben geschützt, *Segmentation violation*

Stack: **für die Ausführung von Prozeduren/Funktionen**

Bem.: bei monolithische Betriebssysteme: z.B. UNIX/Linux liegt der gesamte Code des BS-Kerns und alle Datenstrukturen **in einem** virtuellen Adressraum (u.a. Treiber)

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 9

Beispiel: Prozessraumaufteilung unter Windows2000

- **Dynamic link libraries (dll): ausführbare Programme,**
 - Code der dll existiert nur in 1 Kopie im Speicher,
 - von mehreren Prozessen **gemeinsam** nutzbar
- jeder Prozess besitzt **eigene lokale Daten** und **eigenen Code**

Virtueller Adressraum von A Prog1.exe Virt. Adressraum von B

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 10

Bem.: welche Schutzziele verletzt??

Sicherheitsprobleme: u.a.

- bei Prozeduraufrufen (auch bei System calls siehe unten):
 - Ablage von **Rücksprungadresse** und **Parameter** auf dem **Stackbereich** des Prozess-Adressraums
 - Ansatzpunkt von **Bufferoverflow-Angriffen**, um Code (z.B. Trojaner) auf Stack einzuschleusen und ausführen zu lassen (**mit den Rechten des Prozesses** (gerne Root!))
- **Viren**-Infektion von dlls als gemeinsame Programme
- Windows-Familie: Browser, E-Mail-Client Outlook:
- Ausführung häufig als Prozesse mit **Root-Rechten**, damit
 - z.B. **unbeschränkter Zugriff** auf Dateisystem,
 - Ausführung von Script-Befehlen mit Root-Rechten, ...

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 11

3.1.4 Prozessverwaltung (nur kleiner Auszug)

Prozess:

- ein Prozess ist ein **Programm in Ausführung**, z.B.
- mehrere Kopien des Editors öffnen: jeweils ein Prozess,
- aber zugehöriges Programm (Code) idR nur einmal im Speicher

- **Erzeugen**: z.B. fork, exec (Unix), CreateProcess (W2K)
- Prozess: durch **Prozess-Kontroll-Block** (PCB) beschrieben:
u.a. Puid, offene Dateien, Rechte, PC, Register, Seitentabelle
- **Anlegen** eines virtuellen Adressraums,
z.B. beim fork: Kopie des Adressraums des Vaters für Kind
- **Eintrag** für Prozess in Prozesstabelle (feste Länge)

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 12

Sicherheitsprobleme: u.a. Bem: welche Schutzziele verletzt??

- **Überlauf der Prozesstabelle** durch zu viele forks:
Ansatzpunkt von Denial of Service Angriffen (**DoS-Angriff**)
- Prozesse besitzen uU **zu viele Rechte** (Root-Rechte!)
Attraktives Angriffsziel, falls Prozess Buffer-Overflow-Schwachstelle aufweist: **Prozess-,Hijacking** mit unautorisierter Rechteübernahme

3.1.5 Interrupts und Systemaufruf

Interrupts (Unterbrechungen)

- Heutige BS arbeiten idR **Ereignis (event)** getrieben
- Auftreten eines Ereignisses wird durch **Interrupt** angezeigt:
 - bei Hardware über Leitungen/Busse,
 - bei Software über **Systemaufruf**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 13

Vergrößerter Ablauf beim Interrupt:

- Anzeigen des Interrupts (z.B. über spezielle Interrupt-Leitungen)
- CPU unterbricht aktuellen Prozess, rettet dessen Status (Register) und
- CPU wechselt in **Kernel-Modus**
- Suche nach Interrupt-Behandlungsroutine:
 - Tabelle: **Interrupt-Vektor**,
 - enthält Anfangsadressen der Handler,
- Ausführung der **Handler-Routine** im privilegierten Modus

Das Diagramm zeigt den Ablauf eines Interrupts in einem Prozessor. Ein Prozess wird durch einen Interrupt unterbrochen. Die CPU speichert den Status (Current instruction, Next instruction) und dispatcht den Interrupt an einen Handler. Nach der Ausführung des Handlers erfolgt ein Return zum nächsten Befehl.

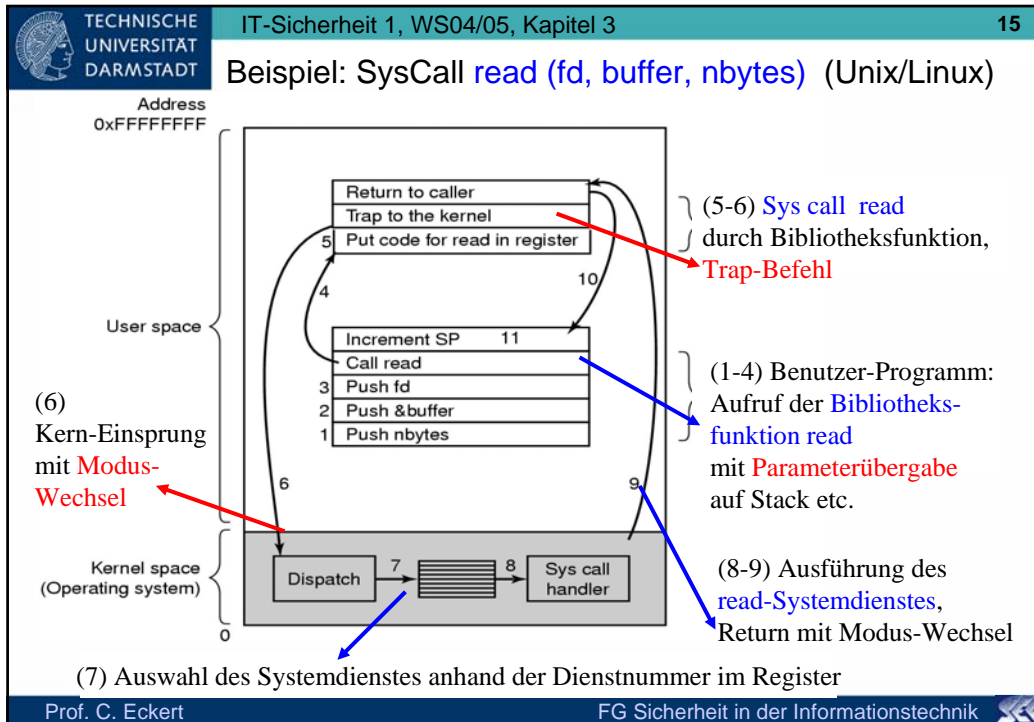
TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 14

Systemaufruf (engl. system call)

- **Schnittstelle** zwischen Prozessen und BS
- **Unix:** Systemaufrufe direkt aus C, C++ Programmen
- **Windows:** Aufruf u.a. über Win32 API (Nutzung von DLLs)

Ablauf: starke Analogie zum Prozeduraufruf,

- **Parameterübergabe** vom User-Prozess zum Kernel über: Prozess-Stack (siehe Folie 8), Register, Speicherbereiche (Pointer)
- **Nummer** des Systemaufrufs wird idR in ein Register geschrieben
- **Modus-Wechsel u. Kontrolltransfer** an BS-Kern: **Trap**
- Dienstnummer dient als Index in den **Interrupt-Vektor**
- Nach Beendigung des Systemaufrufs: Kontroll- u. Modus-Wechsel **zurück**, Prozess wird **fortgesetzt**



TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 16

Bem:

- Interrupt-Tabelle **idR Speicher-resident** (niedrige Adressen)
- Anfangsadresse der Tabelle z.B. fest oder über ein Register
- Interrupt-Handler für E/A-Geräte:
sind **Bestandteile der Treibersoftware**
- Tabelle: wird **beim Booten** mit den Anfangsadr. initialisiert

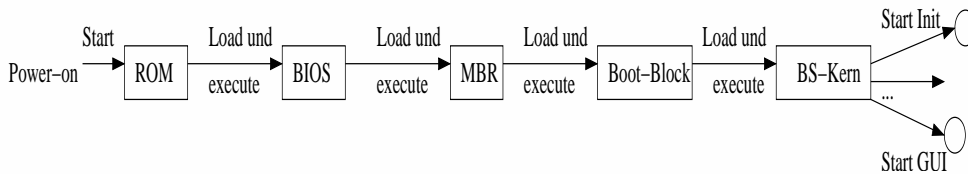
Sicherheitsprobleme: u.a.

- Interrupt-Vektor meist nicht vollständig belegt, Platz u.a. für **Viren**, die dann sogar **resident** sind
- **modifizierte** Tabellen-Einträge:
Sprung zu Virencode, statt zum korrekten Handler, Virus wird im **Kernel-Mode** ausgeführt

Prof. C. Eckert FG Sicherheit in der Informationstechnik

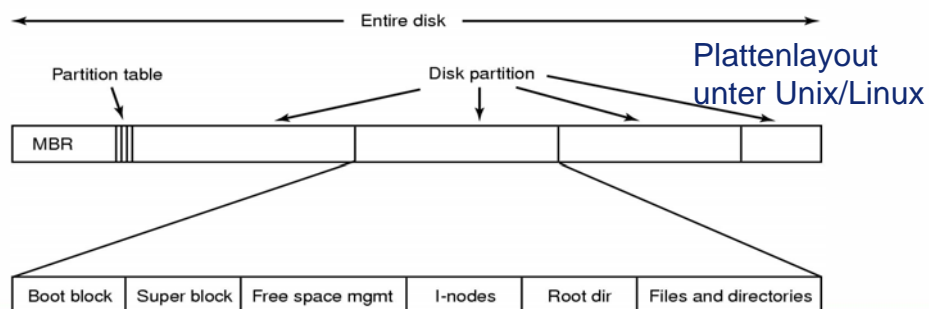
3.1.6 Booten

- Systemstart: Ausführung des **Bootstrap-Programms (BP)**



- idR im **ROM** oder EEPROM initiale Befehle des BP gespeichert
 - Bem.: ROM: **keine Modifikation (z.B. durch Viren) möglich!**
z.B. früher (MS-DOS etc) BIOS vollständig in ROM
- **Aber:** gewollte Modifikation erfordert Veränderung des ROM-Hardware-Chips, deshalb
- nur initialer Teil des **BP** steht im ROM, dieser Teil lädt das BIOS

- **BIOS** steht idR im in RAM (nicht-flüchtig, veränderbar)
Ausführung des BIOS:
 - Power-on-Self-Test (POST), prüft einige Basiseigenschaften
 - Laden des eigentlichen **Bootstrap**-Programms von der Platte
- Start des Bootstrap-Programm steht an fester Stelle auf Platte:
 - im Platten-Sektor 0 im **MBR** (Master Boot Record)
- der Rest des Bootladers befindet sich im **Boot-Block** (Boot Partition),



TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 19

- **MBR** wird in Hauptspeicher geladen, MBR enthält:
 - Anfangs-Code des BP und
 - Tabelle mit Anfangsadressen der Plattenpartitionen
- Anfangs-Code-Stück des BP wird ausgeführt und lädt Inhalt des **Boot-Blocks** der aktiven Partition
- Boot-Block enthält Rest des Bootstrap-Programms,
- Ausführung des **Bootstrap-Codes**
 - Laden und starten des **BS-Kerns**,
 - BS-Code befindet sich auf aktiver Plattenpartition
z.B. unter W2K: Datei *ntldr* lädt Kern (laden von Boot.ini etc.)

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 20

- Ausführen des BS-Kerns: u.a.
 - **Initialisierung** des Rechners
Z.B. der CPU-Register, EA-Controller, Speicher, ...
 - Einlesen von Informationen über das Dateisystem:
Superblock, Dateibeschreibungen (z.B. Unix-inodes), ...
 - **Starten des ersten Prozesses**, z.B. *init-Prozess*
- Startprozess: u.a.
 - lädt alle **Gerätetreiber** von angeschlossenen Geräten
 - Starten weiterer **System-Prozesse** und u.a. der Login-GUI

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 21

Beispiel: initiale Prozesse beim Booten von W2K

Process	Description
idle	Not really a process, but home to the idle thread
system	Creates smss.exe & paging files; reads registry; opens DLLs
smss.exe	First real proc; much initialization; creates csrss & winlogon
csrss.exe	Win32 subsystem process
winlogon.exe	Login daemon
lsass.exe	Authentication manager
services.exe	Looks in registry and starts services
Printer server	Allows remote jobs to use the printer
File server	Serves requests for local files
Telnet daemon	Allows remote logins
Incoming email handler	Accepts and stores inbound email
Incoming fax handler	Accepts and prints inbound faxes
DNS resolver	Internet domain name system server
Event logger	Logs various system events
Plug-and-play manager	Monitors hardware to see what is out there

- Prozesse oberhalb der Linie werden **stets gestartet**
- Prozesse unterhalb der Linie sind Beispiele für solche, die gestartet werden **können**

Prof. C. Eckert FG Sicherheit in der Informationstechnik



TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit 1, WS04/05, Kapitel 3 22

Bem: welche Schutzziele verletzt?

Sicherheitsprobleme beim Booten: u.a.

- Boot: BIOS liest MBR ein:
 - Angriffspunkt für **Boot-Sector-** oder **MBR-Viren**
 - Beim Booten wird Virus in RAM geschrieben,
 - Ausführung in **Kernel-Modus**, ohne MMU-Schutz, ohne Kontrollen des BS, ohne Antiviren-Programme,
 - Virus kann z.B. alle Interrupts abfangen
- **Zerstören** des Boot-Blocks und Superblocks:
System ist ggf nicht mehr bootbar, Dateien nicht mehr zugreifbar
- Booten eines **modifizierten BS-Kerns**
manipulierte Standardsystemdienste: u.a. login, ls, netstat, ...
- Registrieren **nicht-vertrauenswürdig**er Dienste als Services, die automatisch gestartet werden, ggf mit Kernel-Privilegien

Prof. C. Eckert FG Sicherheit in der Informationstechnik

 <p>TECHNISCHE UNIVERSITÄT DARMSTADT</p>	IT-Sicherheit 1, WS04/05, Kapitel 3	23
<p>Spezial-Aspekt: Registry unter Windows</p> <ul style="list-style-type: none">• Menge von Datenbanken, <i>hive</i>, <i>system-hive</i> für das Booten• enthält eine Fülle von Konfigurationsinformationen, u.a. auch SAM-Verzeichnis (sicherheitsrelevante Infos pro Nutzer)• nach erfolgreichem Booten wird aktuelle <i>system-hive</i> auf Platte gesichert (last-known-good)• über API können Prozesse als Services registriert werden, automatisches Starten beim Booten, z.B. bei NT als vertrauenswürdige Prozesse: mit allen Rechten <p>Sicherheitsprobleme: u.a.</p> <ul style="list-style-type: none">• Einschleusen von Schadsoftware in Registry-Verzeichnisse: automatisches Starten der Schadsoftware		
Prof. C. Eckert	FG Sicherheit in der Informationstechnik	

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 24

3.2 Kommunikationsprotokolle

Motivation: übliches Szenario:

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 25

Ziel: ‚Primer‘ zu Kommunikationsprotokollen, Funktionsweisen, Paketformate, Adressierung

3.2.1 ISO/OSI-Referenzmodell

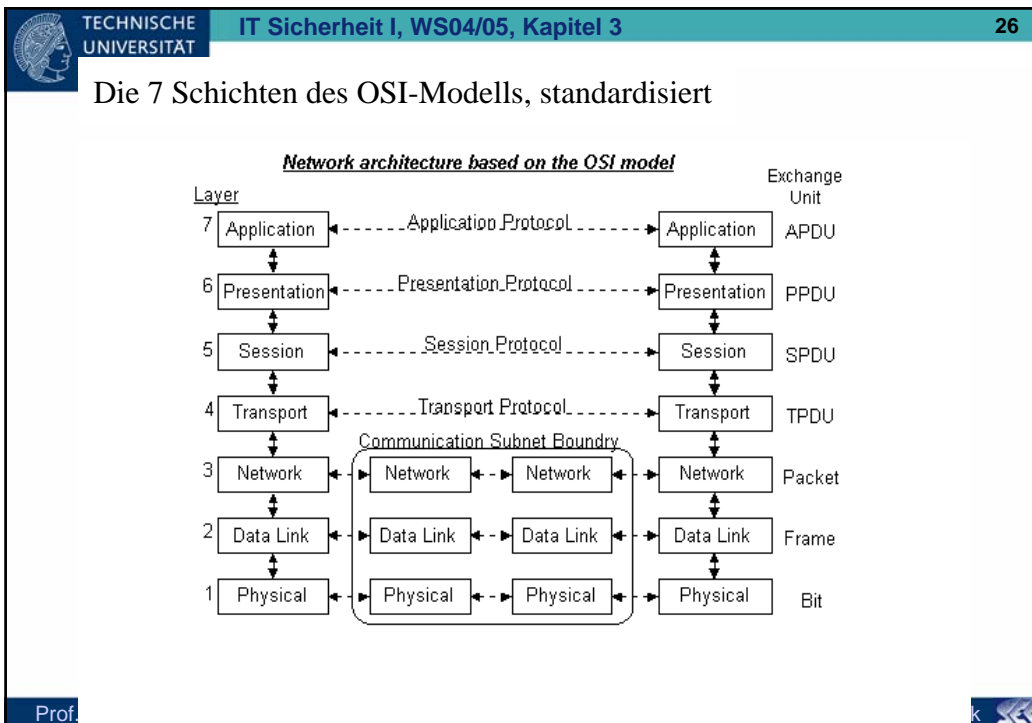
OSI: Open System Interconnection

Schichtenansatz (layer): Prinzip

- Jede Schicht **kapselt** Eigenschaften und Dienste
- Schicht **i** nutzt zur Erbringung ihrer Dienste nur Dienste der Schicht **i-1**
- Dienste der Schicht **i** sind über **Schnittstellen** von der Schicht **i+1** nutzbar

Bsp. Dienste: Fehlererkennung, Verbindungsaufbau, Wegewahl

Prof. C. Eckert FG Sicherheit in der Informationstechnik



- TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 27
- (1) **Bitübertragung** (physical layer):
- physikalischen Verbindung zwischen zwei **direkt verbundenen** Kommunikationspunkten
 - Übertragung **unstrukturierter Ströme von Bits**
 - legt die Charakteristika der **physischen Verbindung** fest, z.B. Spannungspegel, Dauer eines Impulses für 1 Bit, etc
- (2) **Sicherungsschicht** (data link layer):
- Austausch von Daten zwischen **direkt verbundenen** Knoten
 - bündelt Bitströme zu Einheiten, genannt **Frames**
 - strukturiert Bitstrom u. fügt **Kontrollinformationen** zu: u.a
 - **Prüfsummen** zur Erkennung von Übertragungsfehlern
idR CRC (cyclic redundancy check)-Codes
- Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 28

Frame Flag= 0111 1110 Steuerinfo Nutzdaten Prüfsumme Flag

- **Vorsicht:** Prüfsummen erfüllen **nicht Integritäts**-Anforderungen gemäß der Schutzziele, **Warum nicht?**
- **Flusskontrolle** (Sliding Window etc.):
Anpassen von Sende- und Empfangsraten
z.B. Drosseln der Senderate: **mögliche Schwachstelle?**
- Spezielle Teilaufgabe bei **Broadcast**-Medien (z.B. Ethernet)
 - gemeinsam genutztes Medium: jeder mit jedem, **mögliche Schwachstelle?**
 - Medium-Access Control (MAC) z.B. CSMA/CD Verfahren
Vorsicht: hat **nichts** mit einer **Berechtigungskontrolle** zu tun
bereinigt nur Konflikte bei gleichzeitiger Nutzung

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 29

Adressierung auf Data link-Layer:

- Ebene 2: direkt verbundene Rechner,
 - **aber** Adressierungsschema bei **Broadcast-Medien** ist **notwendig**: z.B. Ethernet, Wireless LAN (WLAN), Funk
 - tatsächlicher Empfänger muss identifizierbar sein
- Eindeutige Identifikation des Empfangsrechners über 48-Bit Adressen, **MAC-Adressen**, weltweit eindeutig
- MAC-Adresse: **im programmierbarem Speicher** (EEPROM),
- d.h. **Adresse ist also änderbar**, Anleitungen hierzu online
 - Treiber muss MAC-Adress-Änderung unterstützen
 - z.B. Ändern der MAC-Adresse unter **Linux**:
`ifconfig <interface> hw ether <neue MAC Adresse>`
 - Bei **Windows** z.B. durch Ändern des Registry-Eintrags
- **dennoch:** Geräte-Authentifik. über MAC-Adresse üblich, **wo z.B.?**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 30

(3) **Netzwerkschicht** (network layer):

- verknüpft und kontrolliert Teilnetze,
- **Ende-zu-Ende** Kommunikation zwischen Kommunikationspartnern, die **nicht direkt benachbart** sein müssen: d.h.
 - Weiterleiten der Daten-**Pakete** über verschiedene Knoten,
 - Vermittlung von Leitungen bei circuit switched Netzen
- **Routingaufgabe** (Wegewahl):
 - Anlegen von Routing-Tabellen (u.a. Pfad von A nach B)
 - Routingalgorithmen: statisch, dynamisch
 - **mögliche Schwachstellen?**
- Erkennen, Auflösen von **Stausituationen** (congestion control)
- **Beispiel: IP** (Internet Protocol siehe später mehr)

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 31

(4) **Transportschicht** (transport layer):

- **Ende-zu-Ende** Kommunikation idR zwischen Prozessen
- in der Regel **zuverlässiges, verbindungsorientiertes** Protokoll:
 - Verbindungsauf- und -abbau (logische Verbindung), z.B. Zustandsinformation in Datenstrukturen ablegen
 - **korrekte Reihenfolge**: Vergabe von Sequenznummern,
 - **erneutes Versenden** bei Paketverlusten, nach Timeout
 - **Bestätigen** empfangener Pakete (Acknowlegment)
 - Beispiel: TCP (später mehr)
 - **mögliche Schwachstellen erkennbar?**
- auch verbindungslose, unzuverlässige Dienste
 - Beispiel UDP (später mehr)
 - **mögliche Schwachstellen?**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 32

(5) Sitzungsschicht (session layer):

- Unterstützung von **Sitzungen über längere Zeiträume** hinweg:
 - **Synchronisation** der Datenübertragung über mehrere Verbindungen
 - **Zustandsinformation** ist notwendig, z.B. Sicherungspunkte, zur Weiterführung eines unterbrochenen Transfers
- Bem: Sitzungsschicht wird selten unterstützt

(6) Darstellungsschicht (presentation layer):

- Aushandeln **gemeinsamer Datenformate**, Syntax, Semantik
Beispiele: ASN.1 (Abstract Notation No 1), XML
- Absprache der **Informationsdarstellung**: Byte-Ordering, Zahldarstellung (u.a. Komplement, Gleitkomma-Darstellung)

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 33

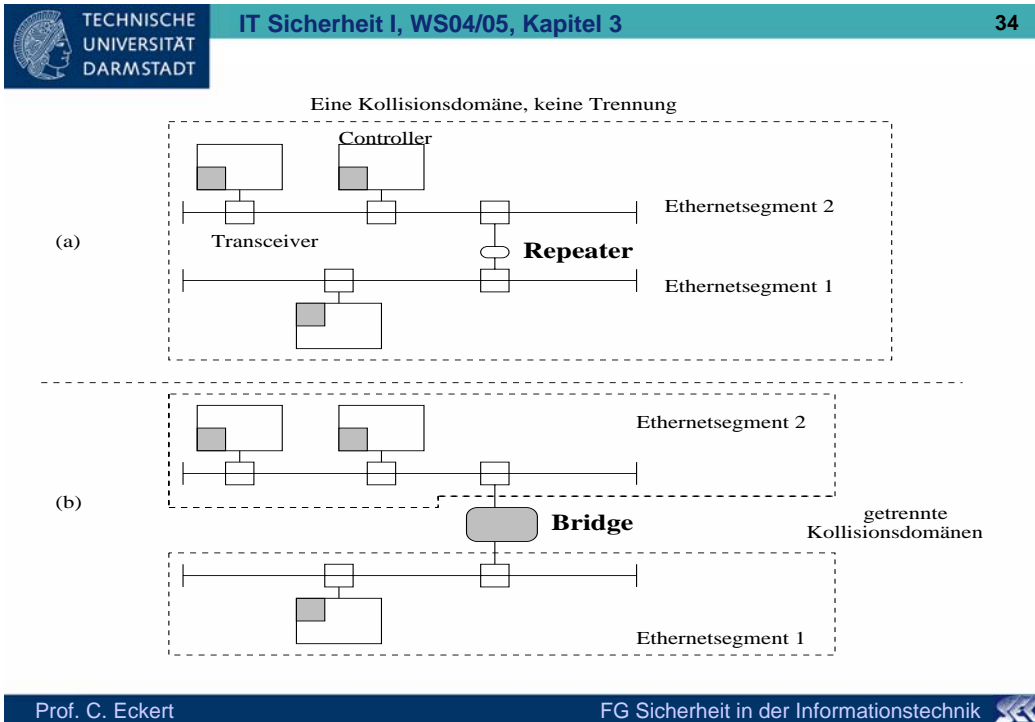
(7) Anwendungsschicht (application layer)

- Vielzahl von Protokollen, die von Anwendungen genutzt werden können
- **Beispiele**: http (www), smtp (e-mail), ftp (Datei-Transfer),

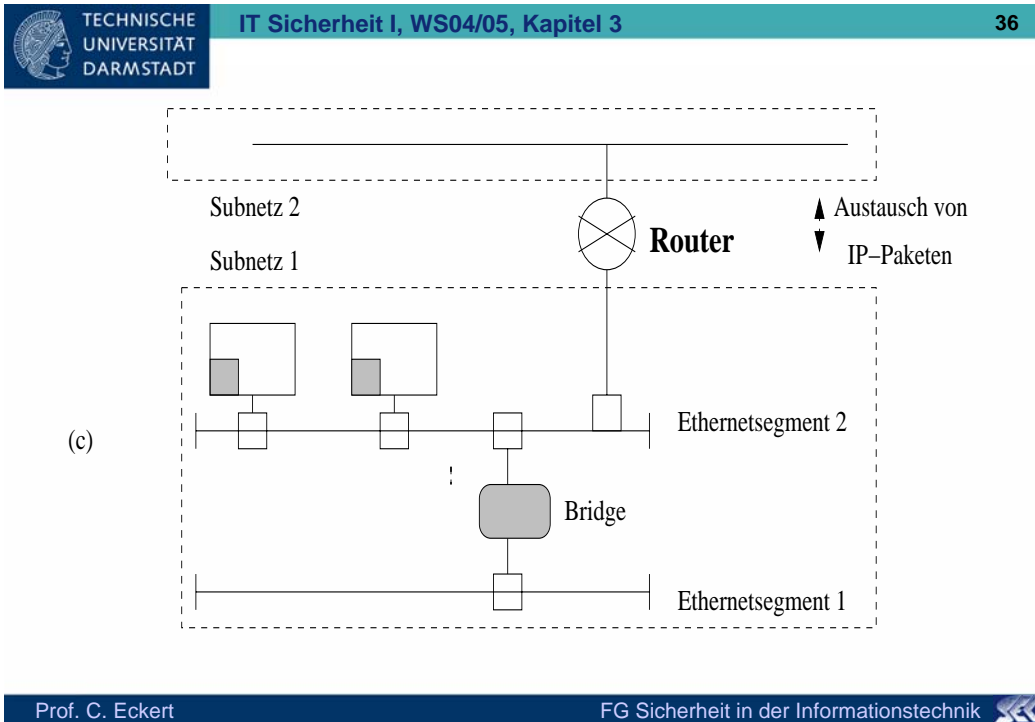
3.2.2 Netzwerk Komponenten

- **Repeater**: auf Schicht 1:
 - Verstärken eines Signals, bitweises Kopieren von einem Netz zum nächsten
 - Verbindung gleichartiger Medien
- **Hub**: Multiport Repeater: verbinden mehrerer Netze
 - eingehendes Signal an einem Port wird aufbereitet und **an alle ausgehenden** Ports weitergeleitet

Prof. C. Eckert FG Sicherheit in der Informationstechnik



- TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 35
- **Bridge**: Verbindung von Netzen auf der Data Link Layer,
 - z.B. Verbindungen von Ethernet-Segmenten,
 - Weiterleiten von Frames an MAC-Adressen, Verwaltung von Tabellen zum Weiterleiten der Frames
 - Verbindung auch unterschiedlicher Medien
 - **Switch**: Multiport Bridge, Weiterleitung von Paketen nur an Ports, mit MAC-Adress-Eintrag (s.u.) des Empfängers, **Filterung möglich!**
 - **Router**: Verbindung von Netzen auf Netzwerk Ebene
 - Routing-Tabellen für Wege-Wahl
manuell oder automatisch erstellt, aktualisiert
 - häufig dezidierter Rechner für Routing Aufgaben
 - **Gateway**: Verbindung von Netzen, auch unterschiedl. Typen
 - z.B. IP-Netz und Apple-Talk
- Prof. C. Eckert FG Sicherheit in der Informationstechnik



TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 37

Im Folgenden: Internet-Protokolle

3.2.3 Internet Protocol (IP)

- Paketvermittelndes, **unzuverlässiges**, **verbindungsloses** Protokoll, Schicht 3
- Best Effort-Prinzip: d.h. Paketverluste, Paketmodifikationen etc. möglich
- Ende-zu-Ende Kommunikation zw. **Rechnern**
- Identifikation über logische **IP-Adressen**
 - IPv4: 32-Bit Adressen, 4er Gruppe a 1Byte z.B. 130.149.25.97 (Dottet dezimal Notation)
 - IPv6 128 Bit Adressen
- Strukturierung durch **Subnetz-Bildung**
Adresse: Netzadressteil und Endpunkt-Teil

TCP/IP-Modell

FTP, SMTP, DNS

TCP, UDP

IP

Data Link z.B. PPP

Bitübertragungsschicht

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 38

3.2.3.1 Subnetze

- Aufteilung eines Adressraums auf verschiedene Subnetze
- Kopplung mittels Routern, effiziente Weiterleitung (Masken)

Schnittstelle nach außen 130.149.0.0 Router 130.149.8.0 130.149.17.0 130.149.25.0 Interne Struktur

3.2.3.2 Vergabe von IP-Adressen

- IANA (Internet Assigned Number Authority): zentrale Vergabe von Adressbereichen an **Registrare**,
- Registrare: Verteilung der Adressbereiche z.B. an ISPs

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 39

- **ISP** (Internet Service Provider): Zuweisen von Adressen bzw. Adress-Bereiche an Unternehmen, Unis, Privatpersonen ...
- **statische Adressvergabe**: z.B. bei Web-Servern
- **dynamisch Adressvergabe**: z.B.
 - durch **Einwahldienste** (AOL etc.) von ISPs,
 - durch **DHCP** (Dynamic Host Configuration Protocol):
 - DHCP-Server verwalten Adressbereiche
 - Client erhält IP-Adresse mit begrenzter Gültigkeit
 - Server verwaltet Vergabeinformation
 - durch **NAT** (Network Address Translation)
 - viele interne (nicht gleichzeitig genutzte) IP-Adressen
 - auf wenige feste öffentlich bekannte IP-Adressen
 - **aber**: Verändern der Header-Infos notwendig!

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT

IT Sicherheit I, WS04/05, Kapitel 3

40

Beispiel: Arbeitsweise einer NAT-Komponente

Man spricht vom 'Natten' und 'Patten'

nach außen bekannte IP-Adresse

Internet

NAT-Tabelle			
PC 1: 10.0.42.1: 16587	↔	130.3.18.39.20001	FTP-Server
PC 2: 10.0.42.2: 32006	↔	130.3.18.39.20004	Web-Server

Lokale IP-Adressen, nach außen unbekannt

Prof. C. Eckert

FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT

IT Sicherheit I, WS04/05, Kapitel 3

41

NAT-Probleme: U.a.


- NAT-Komponente verwaltet Zustandsinformation
- **NAT-Ausfall** (z.B. durch Angriff) unterbricht alle bestehenden Verbindungen
- **Veränderung der IP-Adressen** durch NAT führt zu Problemen, falls Anwendungsprotokoll die IP-Adresse des Host-Rechners benötigt, z.B. IP-Telefonie

NAT-Vorteile: u.a.

- **Verschleierung** der internen Adreßstruktur!

Prof. C. Eckert

FG Sicherheit in der Informationstechnik

 TECHNISCHE UNIVERSITÄT DARMSTADT	IT Sicherheit I, WS04/05, Kapitel 3	42
---	--	-----------


3.2.3.3 Fragmentierung:


- Max. Länge pro IP-Paket: **65.535 Byte** (16 Bit für Längenangabe)
- beim Versand über Schicht 2-Protokolle: ggf. **Fragmentierung**, d.h. Aufteilung des IP-Pakets auf mehrere Frames nötig
- Bsp. Ethernet-Frame: max Payload-Länge von 1500Bytes
- einzelne Fragmente werden **unabhängig** übertragen

3.2.3.4 IP-Paket-Format

- Identifikation des Pakets
- Fragmentierungsinfos für Empfänger, Angabe der Fragmentposition im Paket
- Header-Prüfsumme
- IP-Adressen
- IP-Optionen: z.B. Routing
Loose Source: Pfad-Angabe


↑ IP-Header ↓	Version	Header- länge	Service-Typ	Länge (max. 65.535)		
	Identifikation			Flags	Fragment-Offset	
	TTL Time to Live		Protokoll	Header-Prüfsumme		
	Sende-Adresse					
	Empfangs-Adresse					
	IP-Optionen					Füllbits
	Eigentliche Nutzdaten Payload					

Prof. C. Eckert	FG Sicherheit in der Informationstechnik 
-----------------	---

 TECHNISCHE UNIVERSITÄT DARMSTADT	IT Sicherheit I, WS04/05, Kapitel 3	43
---	--	-----------

Sicherheitsprobleme bei IP: u.a.

- **Fälschen** von IP-Adressen ist möglich: IP-Address-Spoofing (Folie 44)
- Prüfsumme bezieht sich
 - (a) nur auf Header und ist
 - (b) nur eine einfache Checksumme (CRC-Prüfsumme)
- **Verändern** von Nutz- und Headerdaten, Wiedereinspielungen
- Nutzdaten und Header-Daten im Klartext: **Sniffen**
- **Entfernen** von IP-Paketen aus Nachrichtenstrom ist möglich:
 - es reicht, wenn ein Fragment verloren geht: Empfänger verwirft ganzes Paket ohne Sender zu informieren
 - Angriffe durch zu große Pakete **nach Defragmentierung**
- ggf. gezieltes **Umlenken** von IP-Paketen möglich

Prof. C. Eckert	FG Sicherheit in der Informationstechnik 
-----------------	--

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 44

Absender-
adresse

Empfänger-
adresse

130.149.25.97 130.149.25.98 ...

IP-Paket mit gespoofter
Absender-Adresse

PC des Angreifers
10.0.42.1

Internet

Router

gespoofter Rechner
130.149.25.97

interner PC
130.149.25.98

interne Netzdomäne
130.149.25

IP-Spoofing:

- Angreifer verwendet als Absende-Adresse die seines Opfers: **130.149.25.97**
- bei Erfolg: Rechte eines internen Rechners wahrnehmen

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 45

3.2.4 Steuerungsprotokolle auf IP-Ebene

Internet Control Message Protocol (ICMP):

- **Aufgabe:** Übermittlung von Fehler- und Statusmeldungen
- ICMP-Nachrichten werden als IP-Pakete versandt
- **Beispiele:**
 - Erreichbarkeitsfehler z.B. wenn *don't fragment* gesetzt, aber Ziel nur mittels Fragmentierung erreichbar ist
 - Status-Informationen einer spezifischen Maschine: *ping*

Mögliche Sicherheitsprobleme? u.a.

- Denial-of Service Angriffe z.B. **Smurf-Angriff**
 - Versenden einer ICMP-ping-Anfrage an Broadcast Adresse
 - mit **gefälschter Absenderadresse**: Konsequenz?

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 46

Beispiel: Smurf-Angriff

Angreifer (1) ping-Request (schmalbandiger Zugang, z.B. Modem reicht!) an Broadcast-Adresse

Ping-Anfrage mit Absende-Adresse 130.149.25.97

Router (2) Rechner senden Reply

Router versendet ping-Anfrage im Broadcast an alle!

Opfer (3) Bündel an Reply-Nachrichten an die angebliche Anfrage-Adresse

(4) Opfer-System wird überlastet, Angreifer kann z.B. sich jetzt als Opfer maskieren

Bem.: Ggf. auch Router und Swiches überlastet

IP-Adresse 130.149.25.97

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 47

Address Resolution Protocol (ARP): (in IPv4)

- **Abbilden:** IP-Adressen auf MAC-Adressen (Ethernet)
- Rechner: führt ARP-Adresstabelle in Form eines Caches
 - Cache-Miss: Empfänger IP-Adresse nicht im Cache, Broadcast-Request (Netzsegment): wer hat MAC-Adr.?
 - Rechner mit angefragter Empfänger IP-Adresse: schickt seine MAC-Adresse zurück (ARP-Reply)
 - Cache-Eintrag wird aktualisiert nach wenigen Minuten
- **Problem:** ARP ist ein zustandsloses Protokoll,
 - ankommende ARP-Replies werden vom Router auch ohne vorherige Anfrage angenommen
- **Mögliche Sicherheitsprobleme:** u.a.
 - Spoofing:** Unterschieben gefälschter MAC-Zuordnungen
 - ARP-Broadcasts:** künstliche Broadcasts (DoS)

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT

IT Sicherheit I, WS04/05, Kapitel 3

48

Beispiel: ARP-Broadcast

Situation: H1 sendet Paket an H2:

- IP-Adresse von H2 ist H1 bekannt, aber
- die MAC-Adresse ist nicht bekannt,

Senden eines (1)Broadcast mit Anfrage nach MAC-Adr

H1 IP-Adresse: 212.23.23.40

IP-Adr. 212.23.23.1
MAC 00:00:0C:a:A7:30

212.23.23.5

Bridge

H2 MAC-Adr. 00:00:0C:E8:E4:D3
IP-Adr.: 192.168.0.2

Broadcast-Bereich

Prof. C. Eckert

FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT

IT Sicherheit I, WS04/05, Kapitel 3

49

Beispiel: ARP-Broadcast (Forts.)

Situation: H1 sendet Paket an H2:

(1)Broadcast mit Anfrage

(3) Eintrag in ARP-Cache

ARP-Cache
192.168.0.2 ↔
00:00:0C:E8:E4:D3

H1 MAC-Adresse 00:00:0D:30:5A:4D
IP-Adresse: 212.23.23.40

(4) Nachfolgend sendet H1 Pakete
Direkt an die MAC-Adresse
00:00:0C:E8:E4:D3
(wer auch immer dahinter sich verbirgt!)


Bridge

(2) Broadcast-Reply mit MAC-Adresse von H2 (wirklich die von H2?!)

H2 MAC-Adr. 00:00:0C:E8:E4:D3
IP-Adr.: 192.168.0.2

Prof. C. Eckert


FG Sicherheit in der Informationstechnik


 TECHNISCHE UNIVERSITÄT DARMSTADT	IT Sicherheit I, WS04/05, Kapitel 3	50
---	-------------------------------------	----

3.2.5 Transmission Control Protocol (TCP)

- Schicht 4-Protokoll, setzt auf dem IP-Protokoll auf


IP-Header	TCP-Header	Nutzdaten
-----------	------------	-----------
- zuverlässig**: einhalten der **korrekten Reihenfolge**
 - Vergabe von **Sequenznummern** an Pakete,
 - Absender bestimmt **ISN** (Initial Sequence Number),
 - gefordert, dass ISN möglichst nicht vorhersagbar ist
 - Sliding-Window für Koordinierung: Sender/Empfänger
- zuverlässig**: **Re-Transmission**, falls Fehler erkannt
 - Erhalt eines TCP-Pakets wird quittiert (**Ack-Nummer**)
 - Sender setzt **Timer**, speichert Pakete in Warteschlange
 - falls kein rechtzeitiges Ack: Re-Transmission des Pakets

Prof. C. Eckert	FG Sicherheit in der Informationstechnik 
-----------------	---

 TECHNISCHE UNIVERSITÄT DARMSTADT	IT Sicherheit I, WS04/05, Kapitel 3	51
---	-------------------------------------	----

TCP-Paket

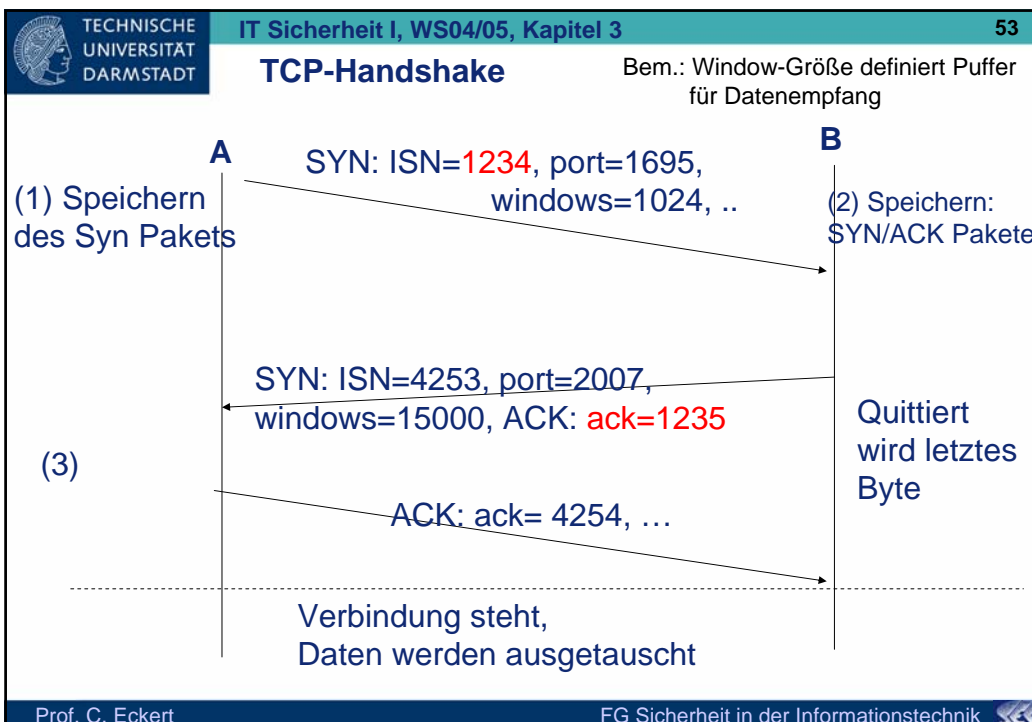
16 Bit Sende-Port	16 Bit Empfangs-Port	TCP-Header 20 Byte
32 Bit Sequenznummer		
32 Bit Acknowledgement		
16-Bit Window-Größe		
16-Bit Prüfsumme für TCP		
Optionen		
Payload		
─────────── 32-Bit ───────────		

Prof. C. Eckert	FG Sicherheit in der Informationstechnik 
-----------------	--

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 52

- **verbindungsorientiert:**
 - Aufbau einer **logischen Ende-zu-Ende Verbindung**
 - Verbindung zwischen **Ports: 16-Bit Adresse**
 - Datentransfer zu Dienst, der an Port gebunden ist
 - Ports < 256 **privilegiert**, nur durch root-Prozesse bindbar
 - Beispiel für Standarddienste an Ports:
Port 80: HTTP; Port 25: SMTP; Port 21: FTP;
- **3-Wege-Handshake** für Verbindungsaufbau
 - (1) Sender: sendet SYN-Paket mit:
ISN, Portadresse für Antwort, Fenstergröße
SYN-Paket: gespeichert (z.B. Linux: Default: 128 Anfragen)
 - (2) Empfänger: SYN/ACK-Paket zurück
legt Paket auch bei sich ab
 - (3) Sender bestätigt seinerseits Empfang des Pakets

Prof. C. Eckert FG Sicherheit in der Informationstechnik



TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 54

Sicherheitsprobleme mit TCP: u.a.

- **Session Hijacking**,
z.B. mit Tool hunt (<http://www.securityfocus.com/tools/834>) unter Linux
 - z.B. Übernahme einer Telnetverbindung zwischen dem Router
 - mit IP-Adresse x.y.z und dem Host mit Adresse u.v.w
- **Angreifer:**
 - mittels **ARP-Spoofing**: Cache-Einträge ändern:
 - im Router : u.v.w <-> MAC-Adresse des Angreifers
 - im Host: x.y.z <-> MAC-Adresse des Angreifers
 - **Effekt**: alle Daten zw. Router und Host gehen über Angreifer
 - Telnet-Session kann protokolliert werden,
 - mit CTRL-C Übernahme der Session durch Angreifer

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 55

Sicherheitsprobleme mit TCP: cont.

- Zuordnung: **Port und Dienst** ist nur Konvention! D.h. blockieren von Ports heißt nicht: deaktivieren des Dienstes
- Privilegierte Ports: jeder **Sysadmin** kann root-Prozesse starten
- **DoS-Angriffe**: z.B. SYN-Flooding auf einen Port XYZ
 - Ausschöpfen des Speichers für nicht bestätigte SYN/ACK Antworten des Empfängers
 - Port XYZ des Empfängers ist nicht mehr erreichbar
- Flooding mit **Absenderspoofing**:
 - SYN-Anfrage des Senders unter gefälschter Adresse
 - SYN/ACK des Empfängers wird nicht beantwortet
 - Empfänger muss nach Timeout Verbindung beenden

Lessons Learned? Was wird benötigt?

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 56

3.2.6 User Datagram Protocol (UDP)

- auch auf Schicht 4, auf IP aufsetzend
- UDP-**Prüfsumme**: über ganzes Paket (nicht nur Header)
- **aber UDP: verbindungslos** und **unzuverlässig**, d.h.
 - Sender hat auf IP und UDP-Ebene keine Informationen, ob Daten beim Empfänger angekommen sind
 - UDP-Pakete können in beliebiger Reihenfolge beim Sender eintreffen,
 - Duplikate werden nicht erkannt
- Kommunikation erfolgt zwischen **Ports** als Endpunkten

Sicherheitsprobleme: u.a.

- **Wiedereinspielungen, Paketverluste, Modifikationen**
- **kein Zusammenhang** zw. Anfrage und Antwortpaketen

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 57

3.3. Mobile, Drahtlose Kommunikation

Klassifikation

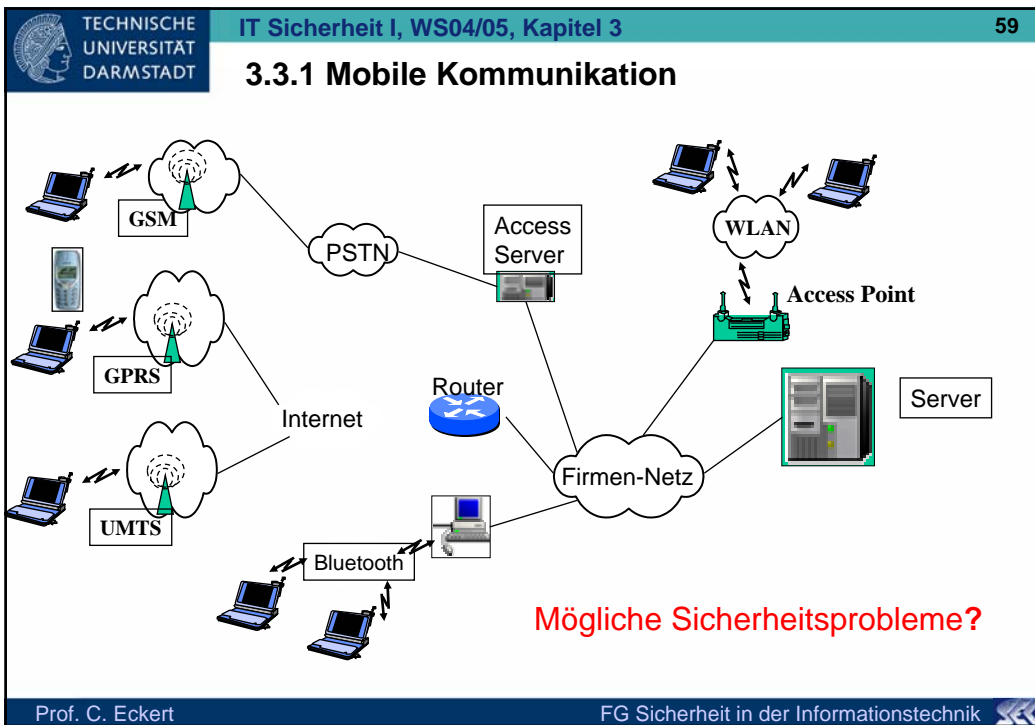
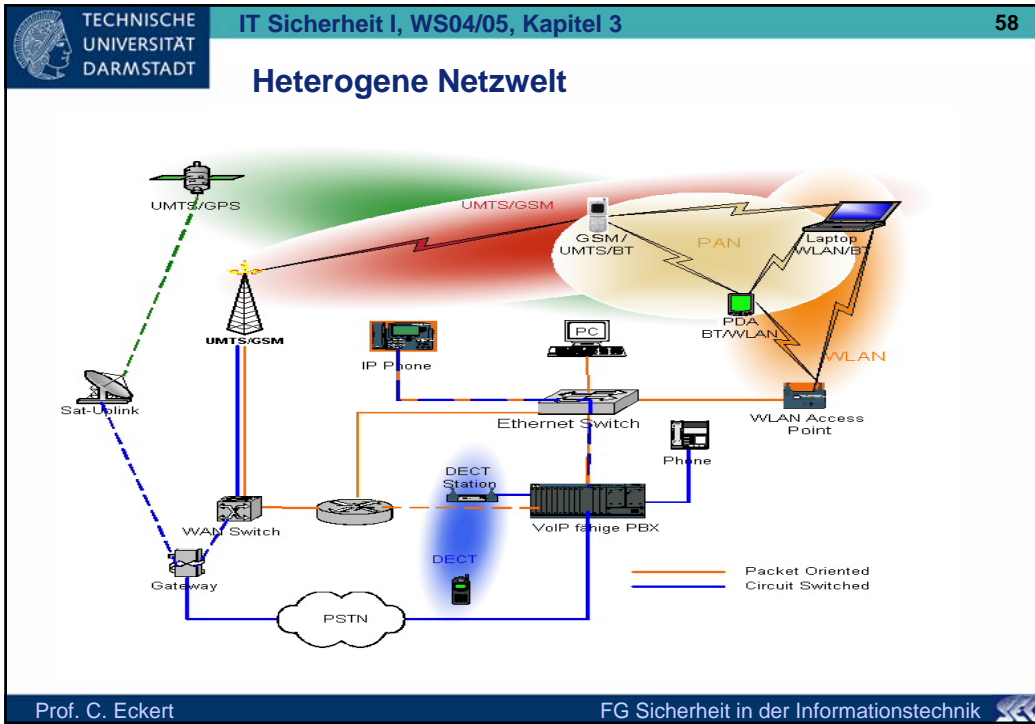
```
graph TD; Root[Systeme mit drahtloser Kommunikation] --> Mobiltelefonie; Root --> Drahtlose[Drahtlose, lokale Netze]; Root --> PAN[Personal Area Networks PAN]; Mobiltelefonie --> Zellularer[Zellularer Mobilfunk]; Mobiltelefonie --> Schnurlose[Schnurlose Telefonie]; Zellularer --> GSM; Zellularer --> EDGE; Zellularer --> GPRS; Zellularer --> imode; Zellularer --> UMTS; Zellularer --> cdma2000; Schnurlose --> DECT; Drahtlose --> HiperLAN; Drahtlose --> WirelessATM[Wireless ATM]; Drahtlose --> HomeRF; Drahtlose --> WirelessLAN[Wireless LAN, 802.11]; PAN --> Funk; PAN --> Infrarot; Funk --> Bluetooth; Infrarot --> IrDA;
```

Weitverkehr (under Mobiltelefonie)

Nahbereich (under Drahtlose, lokale Netze)

Persönliche Arbeitsumgebung (under Personal Area Networks PAN)

Prof. C. Eckert FG Sicherheit in der Informationstechnik



TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 60

GSM-Architektur

Anbindung an Fest- und Datennetze

Heimnetz verwaltet Benutzerinformationen:
Authentifikation des Benutzers,
Zulässigkeit der Netz-Nutzung,
Abbonierte Dienste, Billing, ...

Luftschnittstelle GSM-(Core)-Netz

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 61

GSM-Architektur

Access Server

Heimnetz

- prüft die Identität u. Authentizität des Handys
- verschlüsselte Verbindung: aber nur über Luft-Schnittstelle

Fazit: **GSM-Dienste:**

- Sprach- und Datendienste,
- Authentifikation des Handys
- Verschlüsselte Kommunikation
- Roaming zwischen Providern
- Handover zwischen Zellen

Einbuchen beim stärksten Sender

Luftschnittstelle GSM-(Core)-Netz

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 62

GSM-Sicherheitsprobleme: u.a.

- Verschlüsselung nur bis zur BTS danach alles im Klartext
- Nur Handy authentifiziert sich: u.a.
 - **IMSI-Catcher**: Angreifer maskiert sich als Funkstation (BTS)
 - zwingt Handy seine Identität zu übertragen
 - Handy liefert dem Catcher regelmäßig aktuelle Aufenthaltsdaten, **Erstellung von Bewegungsprofilen!**
 - **Maskierte BTS**: gibt vor, keine Verschlüsselung zu können:
 - Handy setzt daraufhin A5/0 ein: **unverschlüsselte Übertragung**
 - **keine** Maßnahmen für **Datenintegrität**: Modifikationen möglich
 - **keine** Maßnahmen für **Verbindlichkeit**: Abstreiten möglich
 - **SMS**: vollständig im Klartext (Homebanking?!)

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 63

GPRS: General Packet Radio Service

- **paketvermittelnde** Technologie Abrechnung nach: **Datenvolumen**, nicht nach Zeit: **Viren-Problem, Spamming** etc.
- **direkte Internet-Anbindung** von GPRS-Geräten! Jedes Gerät besitzt IP-Adresse: Angriffe über IP-Netze ausgesetzt
- GPRS verwendet gleiche Sicherheit-Architektur und
- gleiche Sicherheits-Maßnahmen wie GS:
gleiche Sicherheits-Probleme

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 64

UMTS (Universal Mobile Telecommunication System)

- all-IP, 2-10 Mb/s Durchsatz,
- Anwendungen: z.B. Rich Telephonie, Audio-, Video-Streaming

Adaption der GSM/GPRS-Sicherheitsdienste

- Authentifizierung des Teilnehmers gegenüber dem Netz
- verschlüsselte Kommunikation auf der Luftschnittstelle,

UMTS-Erweiterungen:

- **Heimnetz authentifiziert** sich auch gegenüber Benutzern
- **Sequenznummern**, um Replay-Attacken abzuwehren
- **Integritätsschlüssel**, um Modifikationen zu erkennen

Aber:

- Absicherung **nur der Luftschnittstelle!**
- **Keine Ende-zu-Ende** Sicherheit
- **All-IP:** Angriffe auf Betreiber-Rechner, Core-Netze? **Vertrauen!**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

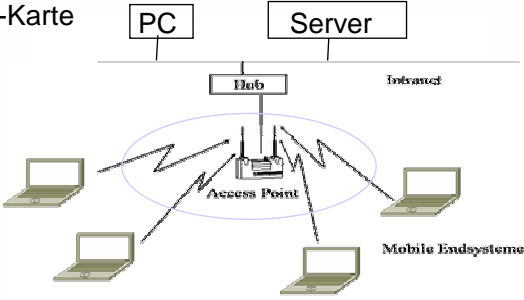

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 65

3.3.2 . Wireless LAN (WLAN) IEEE Standard 802.11

- Nutzung **gebührenfreier** Funkspektren
- verschiedene Standards: 802.11, 802.11b, 802.11g, 802.11a
- **Signalreichweite:** > 150 m, auch durch Mauern etc.
- nächster Standard: WiMAX: Sendereichweite von mehreren km

In der Regel: Infrastruktur-Modus mit Access Point

Z.B. PDA/Laptop mit WLAN-Karte
WLAN-Modem



Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 66

Datenaustausch zw. AP und Geräten: über Funksignale

- **Access Point** sendet regelmäßig **Statusinformationen**
- AP besitzt **SSID** (Identität), falls keine Absicherung:
mit Kenntnis von SSID direktes Nutzen des WLANs möglich
- Gerät kommuniziert mit AP mit **bester Signalqualität**
Versehentliche Fremdnutzung eines Netzes!

Einfachste Angriffe auf WLANs:

- **WarDriving** (Suche nach einem WLAN),
- **Tools verfügbar:** u.a. NetStumbler, MiniStumbler, Airstort
- **Mit Tools: Informationen über:** erreichbare APs, SSID des AP, Hersteller, WEP-aktiviert (y,n), ...

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 67

Ergebnis eines ‚Spaziergangs‘ in DA-Innenstadt (Airstort)

Name des Access Points		Hersteller		Sicherheitsfunktionen aktiviert?				
MAC	SSID	Chan	Vendor	Type	WEP	Beacon Interval	Signal+	Noise-
02EC7FD28309	Test	10		Peer		100	-29	-98
00055DECEF46	default	6	D-Link	AP	Yes	90	-75	-97
0030AB0B8D50	wavevision	6	Delta (Netgear)	AP		100	-68	-97
00022D2F5C5E	CYJYH1495102	11	Agere (Lucent) Orinoco	AP		100	-68	-98
00022D4B086A	wavevision	6	Agere (Lucent) Orinoco	AP		100	-70	-89
00022D41F8CC	wavevision	3	Agere (Lucent) Orinoco	AP		100	-74	-95
0030AB1668DA	Wireless	6	Delta (Netgear)	AP		100	-70	-98
00022D3F4D7C	RBG WaveLAN Net11		Agere (Lucent) Orinoco	AP		100	-91	-96
00022D3F4D76	RBG WaveLAN Net7		Agere (Lucent) Orinoco	AP		100	-70	-97
0004E23990A4	HC02AP	5	SMC	AP		100	-55	-96
00022D3CD28C	HC02AP	5	Agere (Lucent) Orinoco	AP		100	-55	-95
00022D3CD29F	RBG WaveLAN Net1		Agere (Lucent) Orinoco	AP		100	-52	-97
000124F28E1A	dirks home	10		AP	Yes	100	-88	-97
00601D1BDFCE	Frankonia	3	Agere (Lucent) WaveLAN	AP		100	-61	-99

Zusätzliche Maßnahmen sind notwendig!

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 68

Mögliche Schutzmaßnahmen für WLAN?

- Maßnahme gegen unbeabsichtigtes ‚Eindringen‘ in WLAN:
 - **Zugriffskontrolle (AP)** anhand von **MAC-Adressen**
 - **aber:** hoher Administrationsaufwand, manuelle Einträge!
schwacher Schutz: MAC-Spoofing ist einfach
- **WEP** Wired Equivalent Privacy
Ziel: Vertraulichkeit, autorisierte WLAN-Nutzung, Integrität

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 69

„Wirkungsbereich“ von WEP

The diagram illustrates the scope of WEP protection. It is divided into two main sections. The top section, labeled 'Ohne WEP-Schutz' (Without WEP protection), shows a local network environment. It includes a 'Lokaler Server' (Local Server) and 'Lokale Arbeitsplätze.' (Local Workstations) connected to a central 'Hub' and an 'Intranet'. The bottom section, labeled 'WEP-gesichert' (WEP-secured), shows a wireless network environment. It features an 'Access Point' connected to four laptops, one of which is specifically labeled 'Laptop A'. The text 'Mobile Endsysteme' (Mobile Endsystems) is also present, indicating the range of devices protected by WEP.

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 70

Einige Probleme mit WEP:

- **WEP-Authentifikation:** über gemeinsamen Schlüssel
 - Schlüssel: **manuell** im AP und in jedem Client eintragen
 - **nur Gerät** wird authentifiziert, nicht der Benutzer
 - AP authentifiziert sich **nicht**: Spoofing-Angriffe

Alle Geräte verwenden gleichen WEP-Schlüssel!
Schlüssel ist kein Geheimnis mehr!

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 3 71

WEP-Verschlüsselung: u.a.

- Kurze Schlüssel: nominell nur 40 Bit bzw. 104 Bit
- Angriffstools zum Knacken des WEP-Schlüssels K auch mit WEP: **einfaches Abhören möglich!**

WEP-Integrität

- Verwendung linearer CRC-32 Prüfsummen
- Angreifer kann Chiffretexte modifizieren und gleichzeitig die Prüfsumme anpassen, Empfänger erkennt durchgeführte Manipulation nicht

Einschleusen von gefälschten Daten ist möglich

Fazit: WLAN auch mit WEP unsicher! **Verbesserungen notwendig!**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 72

Fazit: Erweiterte Sicherheitsprobleme durch mobile/drahtlose Netze

Netze: Nicht leitungsgebunden (Mobil, Funk, Infrarot):

- **Luftschnittstelle** als Broadcast-Medium: abhören ist sehr einfach, ohne aufwändige/teure Geräte etc. möglich
Probleme der **Vertraulichkeit, Authentizität, Integrität**
- **Netz-Infrastruktur, ad-hoc Netze:** keine vertrauenswürdige Partner, keine zentrale Firewall-Lösung, kein zentrales IDS (Intrusion detection)
Ende-zu-Ende Sicherheit fehlt
Probleme **Billing, Accounting, Authentizität**
- **Heterogene** Netze, Roaming über z.B. GSM, UMTS, GPRS, WLAN, ..
unterschiedliche Bandbreiten, Versorgungsstärken, Abdeckungen
Probleme der **Verfügbarkeit, Erreichbarkeit, Aktualität, DoS**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 3** 73

Literaturhinweise zu Kapitel 3

Betriebssysteme

- A. Tanenbaum
Modern Operating Systems, Prentice Hall, 2001
- A. Silberschatz, J. Peterson, and P. Galvin
Operating System Concepts, Addison Wesley,
- A. Silberschatz, P. Galvin and G. Gagne
Applied Operating System Concepts, John Wiley and Sons

Kommunikationsnetze

- A. Tanenbaum
Computer Networks, Prentice Hall, 1996
- W.R. Stevens
TCP/IP Illustrated, Addison Wesley, 1995 2 Bände

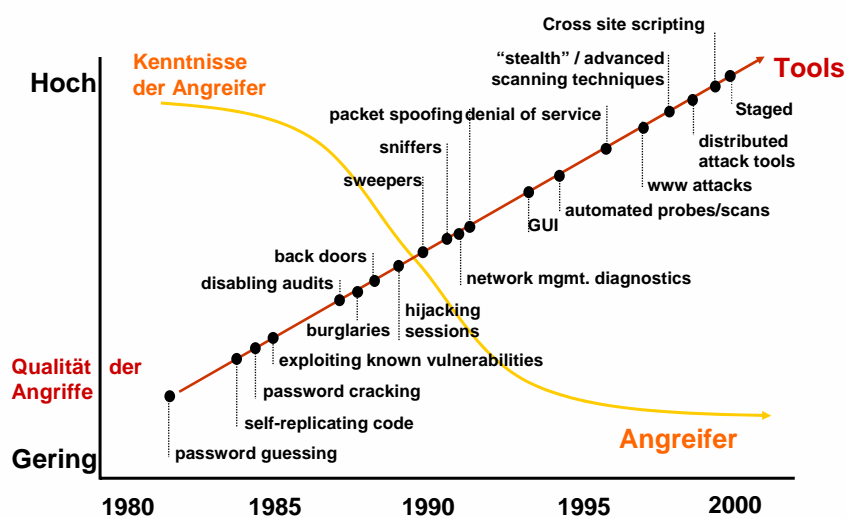
Prof. C. Eckert FG Sicherheit in der Informationstechnik

Kapitel 4 Ausgewählte Sicherheitsprobleme

Angriffsziele:

- Gefährdung von ein einem oder mehreren Schutzzielen
- häufige **Ansatzpunkte**:
 - mangelhafte Identitätsprüfung: **Spoofing-Angriffe**
 - Zustandsinformationen gezielt manipulieren: **Cache-Poisoning** oder **DoS-Angriffe**
 - nicht korrekt überprüfte Eingaben: **Buffer Overflows**, **Cross-Site Scripting**, ...
 - **Social Engineering**: Nutzer ist häufig schwächstes Glied in der Sicherheitskette: **Viren**, **Würmer**, **Trojaner-Angriffe**
 - universelle Interpretierbarkeit: Daten, Code

Qualität der Angriffe vs. Kenntnisse der Angreifer



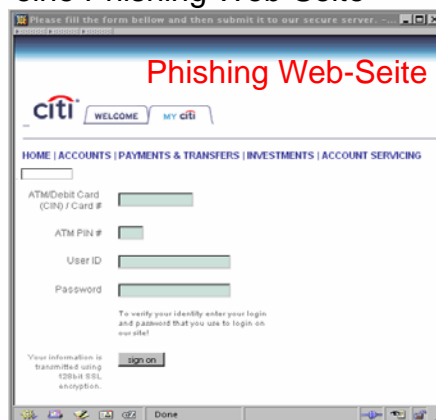
4.1 Mangelhafte Identitätsprüfungen

- Angriffe auf **allen! Schichten** des OSI-Modells anzutreffen
 - Schicht 2: **ARP-Spoofing**
 - Schicht 3 und höher: **IP-Address Spoofing**
 - Schicht 4 (und höher): **Session Hijacking ...**
 - Schicht 7: zugeschnittene Spoofing-Angriffe: u.a.
 - **DNS-Spoofing** (s.u.)
 - gespoofte **E-Mail**-Absendeadressen: SPAM, Viren!
z.B. größeres deutsches Unternehmen:
in 2003: insgesamt **203.102** Viren-Angriffe via e-Mail
in 2004: Jan.: 60.000, Feb. 88.000, März 206.000,
April 476.762 Viren/Würmer in Mails
 - **Web-Spoofing**, inklusive **Phishing**

4.1.1 Phishing (Passwort Fishing)

Benutzer werden mit gefälschten E-Mails und Webseiten getäuscht, um sie zur Eingabe vertraulicher Daten zu verleiten.

Beispiel: Eine Phishing-Mail mit gespooftem Absender und eine Phishing Web-Seite



4.1.2 DNS-Spoofing

- typisches Beispiel für **ungeprüfte Identitätsangaben** und **gezielte Zustandsmanipulation**

Background zum Domain Name Service DNS:

- hierarchischer, verteilter Namensdienst
- Domänen-Hierarchie: hierarchischer Namensraum
- Vergabe von **symbolischen Namen**:
 - z.B. www.sit.fraunhofer.de für Web-Server
 - Einfachere Handhabung für Nutzer als IP-Adressen,
 - **aber**: Zustellung von Paketen (Protokolle) erfordert die Angabe von IP-Adressen

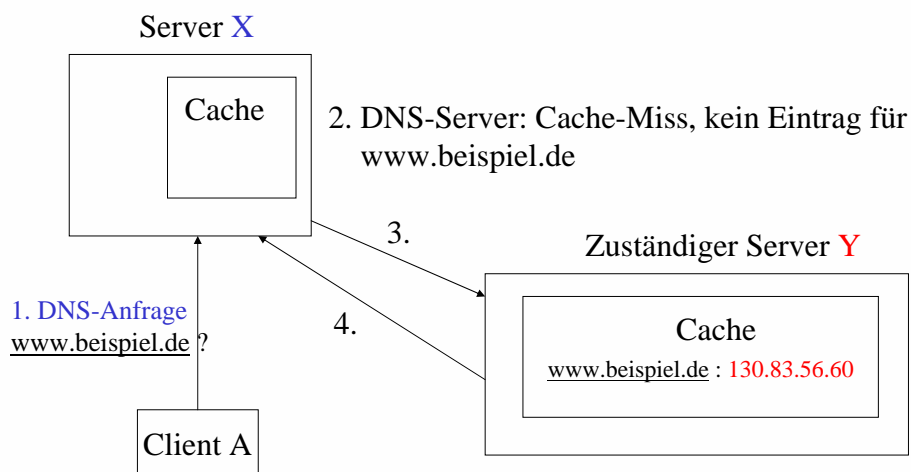
- Aufgabe von DNS: **Abbilden (Resolve)** von **symbolische Rechnernamen** auf deren **IP-Adressen**
- jeder DNS-Server verwaltet einen **Cache** von erfragten Adresszuordnungen
- Über DNS-Anfrage-Antwort Protokoll können Clients von den DNS-Servern IP-Adressen von Rechnern erfragen, von denen nur der DNS-Name bekannt ist
- das Protokoll ist idR **UDP-basiert**, unzuverlässig:
 - Client sendet unbeantwortete Anfragen erneut
 - **Unerwartete** Antwortpakete, z.B. Antworten auf bereits beantwortete Anfragen, werden **einfach ignoriert**

Ablauf bei einer DNS-Anfrage (stark vergrößert):

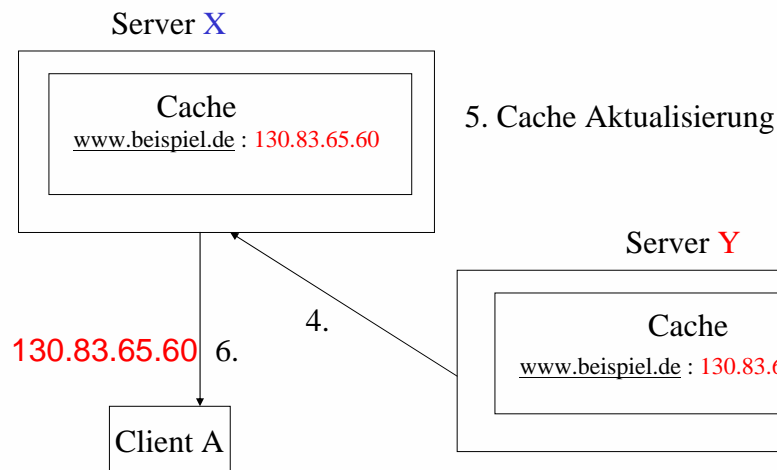
1. Client A stellt eine DNS-Anfrage an den Port 53 eines DNS-Servers X: IP-Adresse von www.beispiel.de?
2. falls Cache-Miss bei Server X,
3. leitet X die Anfrage weiter an zuständigen Server Y
4. falls Cache-Hit, schickt dieser die Antwort, also die IP-Adresse von www.beispiel.de an Server X zurück
5. Server X aktualisiert seinen Cache, um Anfragen nach www.beispiel.de direkt beantworten zu können.
6. Server X liefert Client die IP- Adresse zurück

Ablauf siehe Folien 8,9

Ablauf einer DNS-Anfrage: zusammengefasst



Server Y beantwortet Anfrage (4)

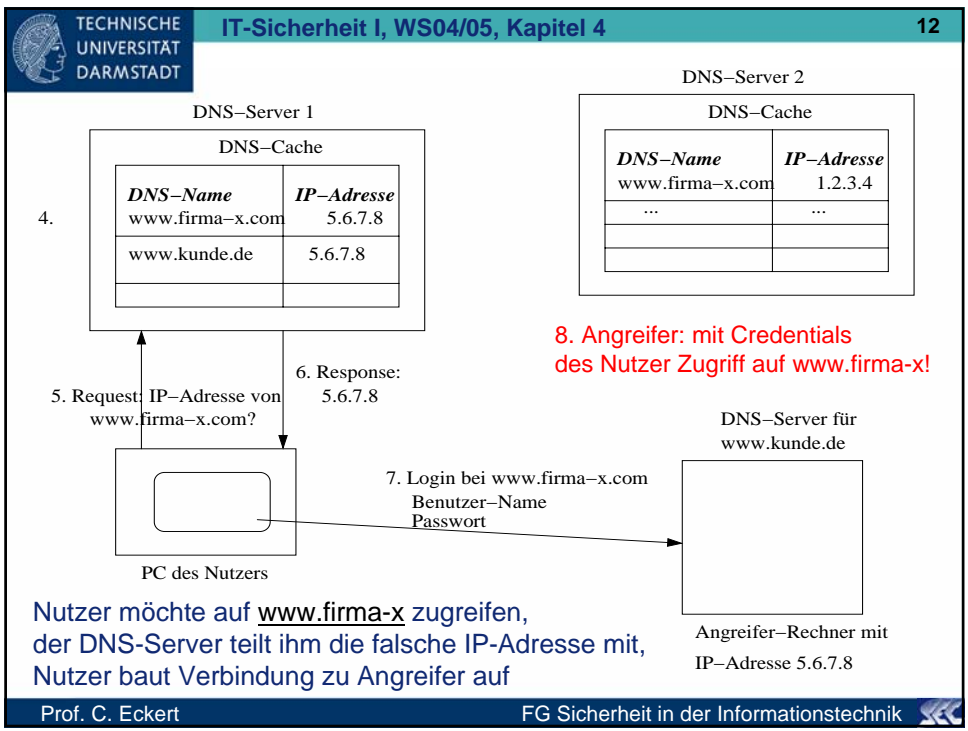
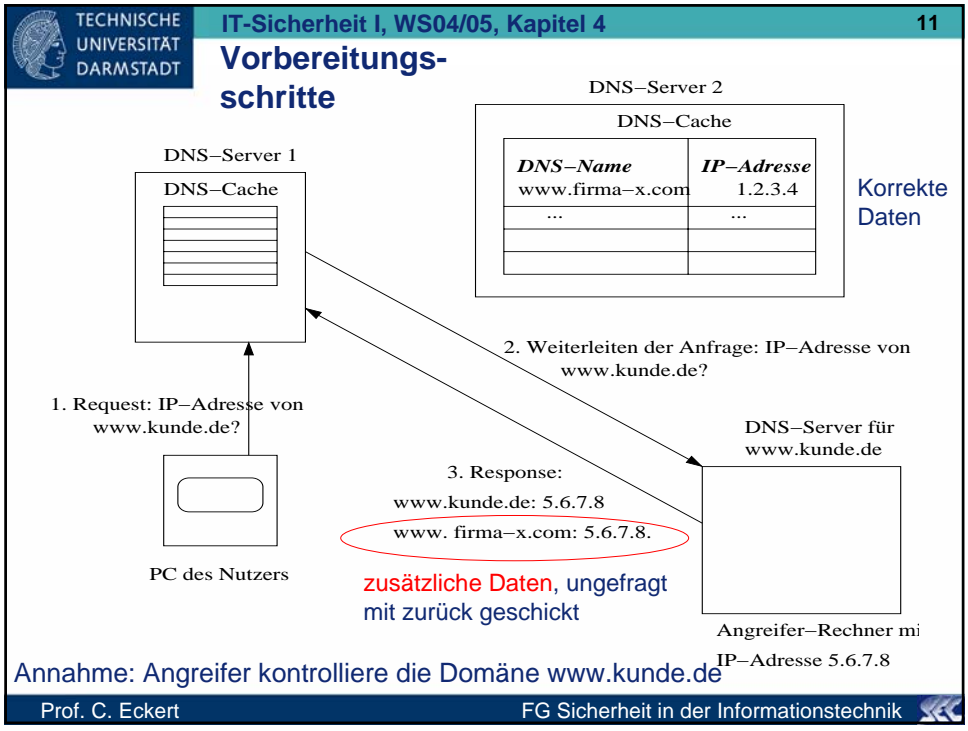


Angriff: Cache-Poisoning: (siehe nächste Folien)

Angriffsziel:

- dem DNS-Server eine **falsche IP-Adresse** zu einem Domännennamen XYZ **unterschieben**
- d.h. dafür zu sorgen, dass ein DNS-Server eine falsche Zuordnung in seinem Cache verwaltet,
- Anfragen von Clients zur Auflösung des Namens XYZ werden mit der falschen IP-Adresse beantwortet
- Anfrager glaubt mit dem korrekten Partner zu kommunizieren,
- **schickt stattdessen seine Daten zum Angreifer**

Abhilfe? Z.B. SSL-verschlüsselte Kommunikation?



Lessons Learned von Problemfeld 4.1?

- in IT-Umfeld verwendete **Identitäten sind idR fälschbar**
- Prüfen der Identität ist deshalb meist unzureichend
- Neben Identitäten benötigt man auch **Authentizitäts-Ausweise, Credentials** (Beispiele?)
- Ausweise dürfen **nicht gefälscht** werden können, bzw. Fälschungsversuche müssen entdeckt werden können

Benötigt werden

- **Authentifizierungsverfahren!** Personen, Rechner, Dienste, ...
- Aber auch ein **sinnvolles Identitätsmanagement**:
Untersuchungen zeigen: 10-70 Passworte pro Nutzer!
- Aber auch: **sichere Verwaltung der Credentials**:
auf der Festplatte? Extern? Wo? Wie sicher verwalten?

4.2 Problembereich: Nicht korrekt überprüfte Eingaben Buffer-Overflow-Exploits als Standardbeispiel

- Häufigste Einbruchmethode in Server, Router oder Clients
- **Ziel:** einschleusen von Code (Viren, Würmer, Trojaner),
Verändern von Daten (z.B. 'Umbiegen' von Funktions-Pointern)
- BO: „Schwachstelle des Jahrzehnts“ (Bill Gates)
- **Ansatz:** Ausnutzen von **Programmierfehlern!**

4.2.1 Vorgehen: (ganz allgemein)

- Überschreiben des Speicherbereichs, die für die Werte einer Variable (z.B. String, Array, Integer) vorgesehen ist, mit **zu großen Werten**, so dass der reservierte Bereich (das ist der ‚buffer‘) überläuft (overflow)

Ursache für erfolgreichen Überlauf:

- **Ungeprüfte Übernahme** von Eingaben/Werten
- häufig bei Programmen/Diensten, in denen Daten über Eingaben in eine Variable eingelesen werden, ohne dass die Größe des Eingabewerts überprüft wird

„Generationen“ von Buffer-Overflow-Angriffen:

- **erste Generation:** Ausnutzen von fehlerhaft programmierten Operationen zum Kopieren von Zeichenketten:
- **zweite Generation:** Überlauf einzelner **Integer-Zahlen**, oder Nutzung von **Schleifen**, die nicht korrekt terminieren z.B. das Zeichenweise Kopieren einer URL (u.a. vom Blaster-Wurm ausgenutzt).

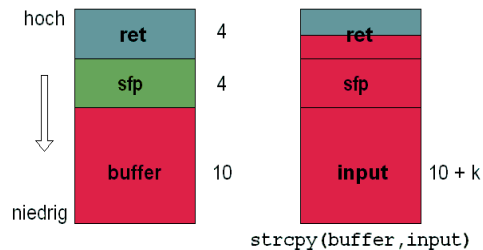
Betroffene Bereiche:

- **Stack-Bereich** des Prozessadressraums: häufigster Angriffsbereich, Stack-Smashing Angriffe (vgl. 4.2.2)
- **Heap-Bereich** des Adressraums: **Angriffe** u.a.:
 - gezieltes Überschreiben von (Funktions-)Zeigern,
 - Überschreiben von mit `malloc()` initialisierten Speicherbereichen mit eigenem Code, ohne eine Speicherschutzverletzung zu produzieren
 - **Effekte:** dadurch kann man z.B. **globale Variable** wie `char[] tmp = „/tmp/prog_swap“` mit `char[] tmp = „/root/.rhosts“` überschreiben und mit **beliebigen Inhalten** füllen
- **Register-Bereiche, Flags** (z.B. im Kernel, Zugriffs-Modi)

4.2.2 Beispiel: Stack-Overflow-Exploit

Beispiel-Code:

```
char input[] = „Windsurfing Hawaii“;
//sizeof(input) = 18
char buffer[10];
strcpy(buffer, input);
```



Effekt:

- strcpy() kopiert Input in Variable **buffer**, ohne Prüfung der Bereichsgröße
- **Überschreiben der Rücksprungadresse**, falscher Wert wird beim Return in Instruction Register geladen

Dieser Programmierfehler könnte einfach beseitigt werden:
`strncpy(buffer, input, sizeof(buffer)-1); buffer[sizeof(buffer) - 1] = '\0';`

Angriffsziel: Einschleusen von Code mittels BO:

Angreifer muss die **genaue Adresse** seines eingeschleusten Codes kennen, um dorthin zu springen, um ihn auszuführen

- **Problem 1:** die genaue Adresse hängt von der Länge, Anzahl der Variablen, Registerinhalte auf dem Stack ab
- **Lösung** z.B. über **NOPs** (No Operations), Prozessor überspringt NOPs bis ausführbarer Code erscheint

Problem 2

Für einzuschleusenden Code steht **nur relativ wenig Platz auf dem Stack** zur Verfügung



Lösung: Platzproblem ist für Angreifer unproblematisch:

- kompakter Assemblercode und vor allem
- **viele Funktionen** stehen Angreifer zur Verfügung, z.B.

Einfache Nutzungen vorhandener Dienste:

- z.B. Windows: viele Bibliotheken (DLLs) stehen **bereits im Speicher** und sind an den befallenen Prozess gebunden
- **Nutzung aller API-Funktionen**, die bereits zum Programm gebunden, z.B. *LoadLibrary* gebunden: **nachladen beliebiger Funktionen** möglich!
- häufig stehen **Bibliotheken an gleichen Stellen** im Speicher, d.h. Angreifer kann sie direkt aufrufen
z.B. zum Aufbau von TCP/IP-Verbindungen,
Erzeugen einer Shell (exec-Befehl), ...

4.2.3 Gefährdungen durch erfolgreiche BO-Angriffe:

Konsequenz einer veränderten Rücksprungadresse:

1. überschriebene Rücksprungadresse enthält **keine sinnvolle** Adresse: **Segmentation Fault**, ggf. bis hin zum Systemabsturz (Verfügbarkeit!)
2. Rücksprungadresse enthält **sinnvolle Adresse**: Programm macht dann **irgendwas**, d.h. es verhält sich nicht mehr gemäß Spezifikation (siehe Folie 21) (Integrität, Vertraulichkeit, Authentizität!!)
3. Rücksprungadresse enthält **sinnvolle Adresse** von auszuführendem Maschinencode, der **vom Angreifer** auf den Stack platziert wurde: **Trojaner, Virus ...**


```
#include <stdio.h>

void function(int a, int b, int c) {

    char buffer1[8];
    char buffer2[16];

    int *ret;

    ret = buffer1 + 12;
    (*ret) += 8;
}

int main() {

    int x;

    x = 0;
    function(1,2,3);

    x = 1;
    printf("%d\n",x);

    return 0;
}
```

Beispiel 2 **Verändern der Programm-Semantik**

Wert der
Rücksprungadresse
wird **um 8** inkrementiert
mit dem Ergebnis, dass
nach Ausführung von
function(1,2,3) der
Befehl **printf("%d\n",x);**
angesprungen wird

Fazit Haupt-Ursache für Buffer-Overflows:

- Programmierfehler insbes. auch in vordefinierten Bibliotheksroutinen in Programmiersprachen wie **C** oder **C++**
- z.B. wie in Beispiel 3.1.2
 - **strcpy()** in **C** mit Parametern: Quell- und Zieladresse
strcpy() kopiert Zeichen für Zeichen von Quell-String zum Zielstring, ohne Bereichs-Prüfung
 - Abhilfe in diesem speziellen Fall:
strncpy(): Begrenzen der Zahl der zu kopierenden Zeichen

Bem.: In C gibt es eine **Vielzahl derartiger Funktionen!**

Bem.: die **meisten BS-Dienste** (Unix, Windows-Welten sind in **C, C++ programmiert!**)

(z.B. Windows XP: ca. 40 Millionen Lines of C-Code!

Richtwert: 5-50 Bugs pro 1000 Lines of Code!)

4.2.4 Gegenmaßnahmen

- Techniken zur **sicheren Programmierung** nutzen:
 - Unbedingt in (C, C++ , ...) Programmen die Eingaben, Bereichsgrenzen prüfen!
 - **Typsichere Sprachen** wie Java verwenden: durch Laufzeitsystem und Compiler erfolgen Bereichs- und Typprüfungen
- Für C, C++: Verwendung **spezieller Bibliotheken**:
 - z.B. Libsafe <http://www.avayalabs.com/libsafe/index.html> (Linux) Wrappen von Standard-C- Bibliotheksaufrufe durch Libsafe-Aufrufe
 - Vorteil: kein erneutes Compilieren des Quelltextes

Falls **Neu-Übersetzung des Quellcodes** möglich:

- z.B. **Programm StackShield** für Linux-Systeme:
sichert bei jedem Funktionsaufruf die **Returnadresse** und korrigiert sie bei Bedarf
- **StackGuard Tool (Unix)**: Modifizierter GNU C-Compiler:
fügt Kontrollzeichen (**canary**) direkt hinter Rücksprung-
adresse ein, prüft Canary vor Rücksprung, schreibt Warn-
meldung in Syslog und terminiert Programm, falls
Änderung erkannt wurde

Bewertung des Ansatzes?

Bem: Stack-Überwachungsprogramme liefern natürliche auch
keinen Schutz vor **Heap-Overflows Exploits**

Betriebssystem-unterstützter Schutz:

- Z.B. unter **Sun-Solaris** konfigurierbar, dass im Stack-Segment kein Code ausführbar ist (stack als **non-executable**)
- Ähnliche **Patches für Linux** <http://www.openwall.org>,
- Und auch für **WindowsNT/2000**
http://securewave.com/products/securestack/secure_stack.html
- **Aber: dieser** Lösungsansatz **hilft nichts**, wenn durch eingeschleusten Code **vorhandener Code** von DLLs oder von Bibliotheksdiensten ausgeführt wird!

Bemerkung:

- BOs sind eine große Klasse von Angriffen, die die Schwachstelle ausnutzen, dass Eingaben nicht geprüft werden,
- Aber bei **weitem nicht die einzigen Angriffe!**
- Weitere große Klassen sind die **CGI- und CSS-Angriffe** siehe Buch, hier nicht mehr ausführlich, Übung!
- **CGI-Skripte: Sicherheitsrisiken für WWW-Server**
 - **ungeprüfte Eingaben** für CGI-Skript, ermöglicht beliebige Code-Injection,
 - **Eingaben** werden von **beliebigen Clients** erstellt,
 - eingeschleuster Code wird auf Server ausgeführt

Cross-Site-Scripting-Angriffe (CSS, oder XSS)

- Angriffsumfeld ist anders als bei BO, aber Ursache ist auch die **fehlende Eingabeüberprüfung**
- Ansatz: **Einschleusen von Code**, der von einem Interpreter ausgeführt wird
- Häufig: Nutzen von **Web-Browsern** als Interpreter:
 - Betroffen (im Gegensatz zu CGI-Problem): sowohl Server als auch alle auf die Seite zugreifenden **Benutzer**
 - Vorgehen: Einbetten von Script-Befehlen in Web-Seite,
 - diese werden vom Browser des Benutzers ausgeführt
- Analoges Vorgehen aber auch z.B. durch Einbetten von URLs in Medien wie MP3, Videos, PS oder PDF-Dateien, Interpreter der Medien führen den Code aus

Lessons Learned

- Buffer-Overflow ist ein fast **allgegenwärtiges Problem**,
- tritt zwar idR bei der Verarbeitung von Prozeduraufrufen auf dem Stackbereich des Prozess- Adressraums auf,
- kann aber auch andere Datenbereiche betreffen!
- Erfolgreiche BO-Angriffe können **alle Schutzziele betreffen!**
- BOs ergeben sich aus **Programmierfehlern!**

Benötigt:

- **Regeln** zur sicheren Programmierung einhalten, **typsichere Sprachen** verwenden (**tritt bei Java/VM kein BO auf??**)
- **Forschung:** neue, erweiterte **Analysetools**, die bereits bei der Compilierung mögliche BO-Schwachstellen erkennen: erste Ansätze gibt es, aber noch nicht ausreichend!

4.2. Schadsoftware (malicious Code, Malware) Viren, Würmer, Trojaner

4.2.1 Viren

In der **Biologie** ist ein Virus

- ein Mikro-Organismus, der auf eine **lebende Wirtszelle** angewiesen ist,
- keinen eigenen Stoffwechsel besitzt und
- fähig ist, sich **zu reproduzieren**

Eigenschaften sind direkt auf Computerviren übertragbar:

Computervirus: von F. Cohen 1984 eingeführter Begriff

Beispiele siehe Buch (u.a. Sobig) und Übungen

Computervirus:

- nicht selbständiges Programm, d.h. es **benötigt Wirt**
Beispiel für Wirt, Bedeutung der Eigenschaft?
- besitzt **Kopierfähigkeit**, ggf. auch mutierend
- enthält i.d.R. einen **Schadensteil**,
d.h. Code zur Durchführung von Angriffen, Beispiel?
- **kann Auslöser** enthalten, d.h. Bedingung zur Aktivierung
des Schadensteils (logische Bombe)
- Enthält i.d.R. eine **Kennung**, z.B. Zeichenketten

- Virus-Code dient häufig **zur** gezielten Angriffs- (Einbruchs)-
vorbereitung: u.a.
 - Infos sammeln, Ports öffnen, Shell-starten, ...

Welche Schutzziele gefährdet?

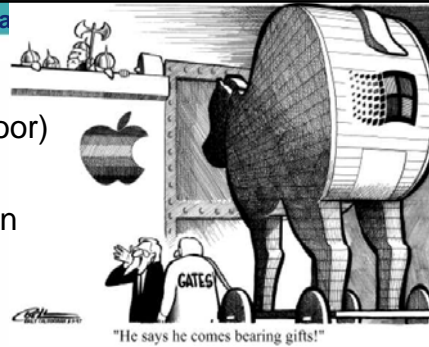
Virentypen

- **Programmiviren** (Link-Viren): infizieren ausführbare Programme (z.B. .exe); Virus-Start mit dem Programm
- **Bootsektor-Viren**: Virus wird resident geladen
- **Makro- und Daten-Viren**: u.a. bei MIME, .ps, .doc, .xls
 - **interpretative Ausführung** von Code
z.B. Starten eines Dateitransfers (von Festplatte) via FTP
 - häufig: **Verbreitung über das Netz**, via E-Mail Attachments, Buffer-Overflow-Angriffe etc.
- **Retro-Viren**: gegen das Immunsystem (z.B. Viren-Scanner)
mögliches Angriffsziel: **Deaktivieren** des Viren-Scans
- **nächste Generation**: Handy Viren? PDA-Viren, was noch?
- Viren **benutzen Spam**, um sich schneller zu verbreiten

4.2.2 Wurm

- selbständig ablauffähiges Programm,
- nutzt die Infrastruktur eines Netzes, um **sich selbstständig zu verbreiten**
- Ausgangspunkte für Wurmangriff häufig: **BO-Angriff** auf Systemprozesse, die ständig rechenbereit sind oder in regelmäßigen Abständen aktiviert werden
- **Beispiele**: (Beschreibung siehe Buch)
 - 2000: **I Love You-Wurm**, Schaden > 8.76 Millionen \$
 - 2001 **Code Red Wurm**: **Pufferüberlauf** in MS-IIS
 - 2002: **Apache/mod_ss** bzw. **Linux Slapper**
 - 2003: **W32/Lovsan/MS Blaster-Wurm**
 - 2004 (u.a.): **MyDoom** (BO im Internet-Explorer)
Bagle Wurm: u.a. deaktivieren von Antivirenprogr.





4.2.3 Trojanisches Pferd (Backdoor)

- Selbständiges Programm,
- besitzt neben seiner spezifizierten Funktionalität eine verborgene, zusätzliche Funktionalität

Beispiel: Backdoor AOT (Sobig-Virus)

- Anzeigen eines jpeg-Bildes und **zusätzlich** :
 - Ausführung eines **Keyloggers**
 - **Verbindungsaufbau** und Transfer von Daten
 - **Durchsuchen** der Registry und Informationstransfer
 - **Eintrag in Registry** und automatischer Start

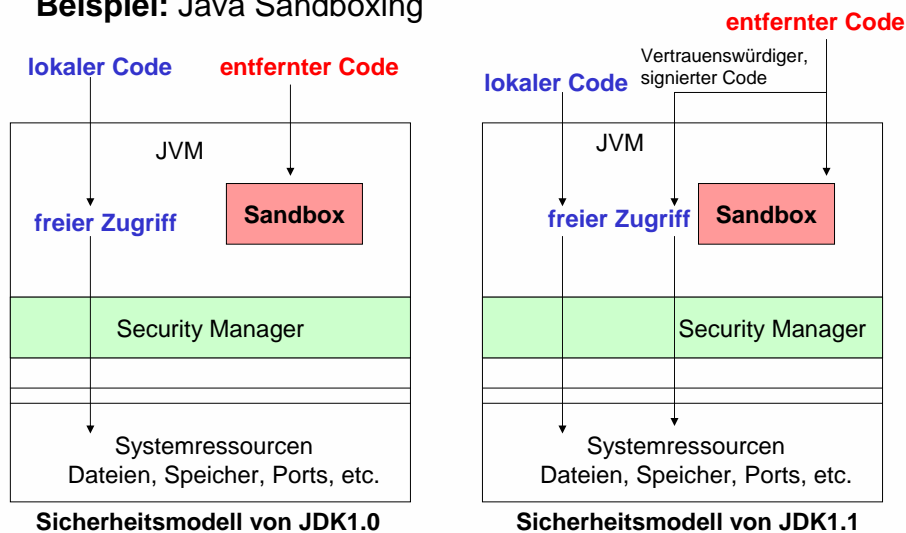
4.2.4 Gegenmaßnahmen

Klassifikation der Maßnahmen (siehe Folie 20, Kapitel 1)

- **Angriffsverhinderung** (prevention):
 - Maßnahmen zur **Verhinderung von BOs**: Programmierung
 - **Beschränktes Laden/Ausführen** von Binärcode-Code: z.B. nur signierter Code, **reicht das?**
 - Maßnahmen sind noch ungenügend: was wünschenswert?
- **Angriffserkennung** (häufigste Maßnahmen heute, reicht nicht!)
 - **Intrusion Detection** (IDS): Erkennen von Signaturen (Vorsicht hier anderer Signatur-Begriff), Beispiele für Signaturen? **Qualität der Erkennung?**
 - **Monitoring**: Veränderungsüberwachung, Protokollieren von Schreibzugriffen etc.

- **Viren-Scanner**: Suchen in Datenbank nach Signaturen, Regeln, welche Dateien (z.B. .exe) zu untersuchen
Heuristiken (wozu?), Qualität der Erkennung?
- **Schaden begrenzen** (Mitigation)
Bem.: Maßnahmen, um Schaden vertrauenswürdig, selbst-organisiert, automatisch zu beheben fehlen
 - Ausführung in isolierten Umgebungen („Quarantäne“), **Sandboxing**-Konzept (vgl. Folie 36), **virtuelle Maschinen**
- **Re-Organisation** (wenige Maßnahmen, häufig manuell)
 - **Patchen**, Patchmanagement:
großes Problem in Unternehmen
 - Konfigurieren: nur die unbedingt erforderlichen Dienste zugänglich machen, **Blockieren von Ports!**

Beispiel: Java Sandboxing



4.2.5 Mobiler Code, aktiver Inhalt (aktiver Content)

- auf einem **entfernten**, potentiell **nicht vertrauenswürdigen** Rechner generiert und
- auf lokalem Rechner **ausgeführt** (häufig interpretiert)

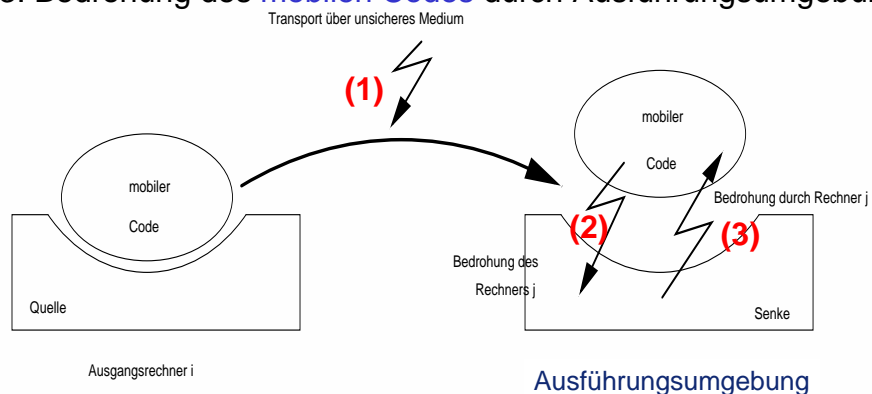
Sprachen: Java (Applets), JavaScript, ActiveX Control, Visual Basic Script, ...

Aktiver Content z.B. in HTML Seiten:

- spezielles **Tag** gibt an, ob es sich um ein Java Applet (Tag <Applet >), um JavaScript-Code (Tag < Script Language JavaScript >), um ein ActiveX-Control (Tag < Object >) handelt

Sicherheitsprobleme

1. Bedrohungen des Codes auf dem **Transportweg**
2. Bedrohung der **Ausführungsumgebung** durch den Code
3. Bedrohung des **mobilen Codes** durch Ausführungsumgebung



Gegenmaßnahmen u.a.

ad (1) **Verschlüsselter Transport**, inkl. Integrität, Authentizität

ad (2) Schutz der Ausführungs-Umgebung:

- **Signieren** von Code, Zertifikate,
- **Bsp**: signierte Applets (JAR) in Java JDK1.0, JDK1.1
Problem : keine Aussage über Funktionalität des Codes
- Zugriffsbeschränkungen durch **Sandboxing**
Problem : meist Alles oder nichts, kaum Differenzierung

ad (3) Schutz des Codes **vor Umgebung**:

- kaum Ansätze
Problem: beliebige Manipulationsmöglichkeiten durch direkte Zugriffe auf Daten/Code
- **notwendig**: Nachweise, dass Gastrechner vertrauenswürdig ist (ggf. Attestation-Konzept von TPM!)

Fazit: Probleme durch Malicious Software **werden steigen**

- Daten enthalten häufig **interpretativ ausführbaren Code**: z.B. Makros bei Word, Excel, Kommandos in pdf,- ps-Dateien
- rapide ansteigender **Austausch** von ausführbarem mobilem Code und auch mobilen Daten(-objekten):
 - u.a. **Peer-to-Peer-Netze** (P2P): Tauschbörsen etc.
 - **Grid-Computing**
- gleichzeitig: rasant **ansteigende Zahl an Rechnern** (Ubiquom): Mobiltelefone mit ausführbarem Code, PDAs, Laptops,
 - manuelles Patchmanagement ist illusorisch
 - Jedermann betreibt bereits ein eigenes Netz, weiss er/sie was er da tut? Weiss er/sie, wofür er ggf. haften muss?

Kleiner Ausblick auf aktuelle rechtliche Diskussionen

Theorie:

- Provider **haften für eigene Inhalte** (Web-Seite, etc.), aber auch für Inhalte **verlinkter Seiten**, wenn er sich erkennbar den Inhalt zu eigen macht (Teledienstegesetz (TDG))
- Hersteller haften für Schäden durch fehlerhafte Computerprogramme (**Produkthaftung**), gilt auch für in Umlauf gebrachte Open Source Software
- Haftung des Unternehmens, Mitarbeiters, etc. falls aufgrund **unsachgemäßer Konfigurierung Schadsoftware** verbreitet wird
- Haftung bei **Verletzung des Urheberrechts** (klassisch: Kopie eines Falkplan-Auszugs als Anreisehinweis auf Webseite ohne entsprechende Genehmigung)

Praxis?

Kapitel 5 Security Engineering

Ziele

- Systematische **Integration** von Sicherheitsaspekten **bei der Entwicklung** von IT-Systemen
- Etablierung von IT-Sicherheit **in bestehenden** IT-Systemen
- Beschreibt (Vorgehens-) **Modelle, Methoden** und Maßnahmen, um sichere IT-Systeme zu entwickeln **und** zu betreiben.
- **Aber:** Noch keine etablierte Disziplin wie Software-Engineering
- Vorgehensprozesse sind **angelehnt** an Methoden aus dem Software-Engineering,
- aber auch an Methoden aus dem Bereich Fehlertoleranz

Vorbemerkung

Wie gehen wir mit Sicherheit im normalen Alltag um?

- wir machen eine **Risiko-Abwägung**
- wir schätzen ab, wie hoch ein Schaden sein könnte, wenn ein Sicherheitsproblem auftritt (z.B. Brand) und leiten daraus ab, wie viel **Aufwand für Schutzmaßnahmen** vertretbar ist
- wir führen für gefährdete Bereiche **spezielle Schutzmaßnahmen** ein: Brandschutzmauern, Airbag, Reisepass, ...
- wir führen **Kontrollen** durch:
Rauchmelder, Aufprall-Sensoren, Grenzkontrollen, ...



Bei IT Sicherheit geht man genau so vor:

- Analyse der möglichen Bedrohungen, des Risikos,
- festlegen geeigneter (angemessener) Sicherheitsmaßnahmen
- festlegen angemessener Kontrollen

Schutzmaßnahmen sollten angemessen sein



... und Kontrollen müssen wirksam sein!



Systematisches, methodisches Vorgehen ist unabdingbar!

5.1 Leitlinien und methodische Hilfestellungen

- **British Standard 7799** „Code of Practice for Information Security Management“ bzw.
- **ISO/IEC 17799** „Information technology — Code of practice for information security management“
- **RFC 2196** „Site Security Handbook “
- **BSI** (Bundesamt für Sicherheit in der Informationstechnologie) IT-Sicherheitshandbuch & **Grundschutzhandbuch (GSHB)**
 - Informationen: www.bsi.de,
 - CD mit Grundschutzhandbuch (kostenlos), > 2000 Seiten!
 - Zusammengefasste Darstellung u.a. im Buch IT-Sicherheit
 - **GSTool**: Hilfestellung bei der Anwendung des GSHB
<http://www.bsi.de/gstool/index.html>

British Standard (BS) 7799

- Ziel: Aufbau eines IT-Sicherheitsmanagements und seiner Verankerung in der Organisation,
- **Management-orientiert**, nicht technisch
- keine detaillierten Umsetzungshinweise, sondern übergreifende Anforderungen

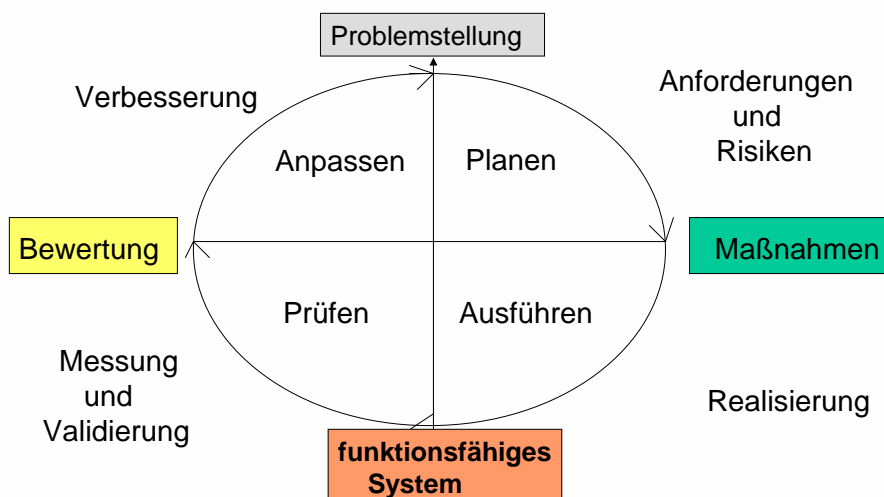
BS 7799 ist Basis für **ISO/IEC 17799**

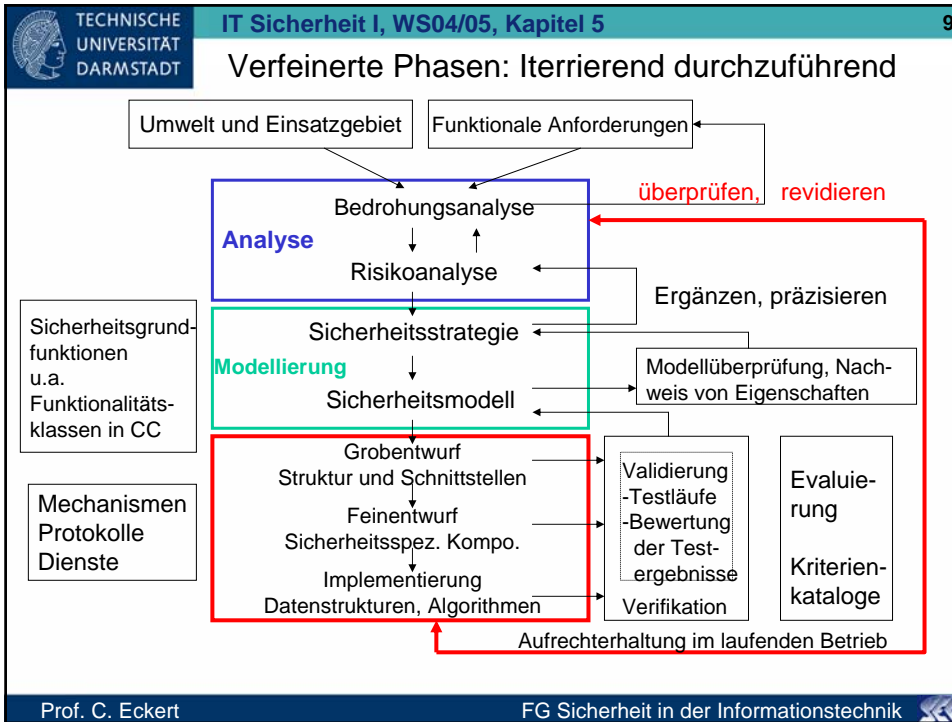
- **Best Practice Verfahren** und –methoden,
- **keine** Empfehlung für **konkrete** Sicherheitslösungen
- **keine** Hilfestellung zur **Bewertung** existierender S-Maßnahmen
- **Themenbereiche**: Security policy, Security organization, Assets classification and control, Personnel security, Physical and environmental security, Computer and network management, System access control, Systems development and maintenance, Business Continuity planning, Compliance.

BSI-Grundschutzhandbuch

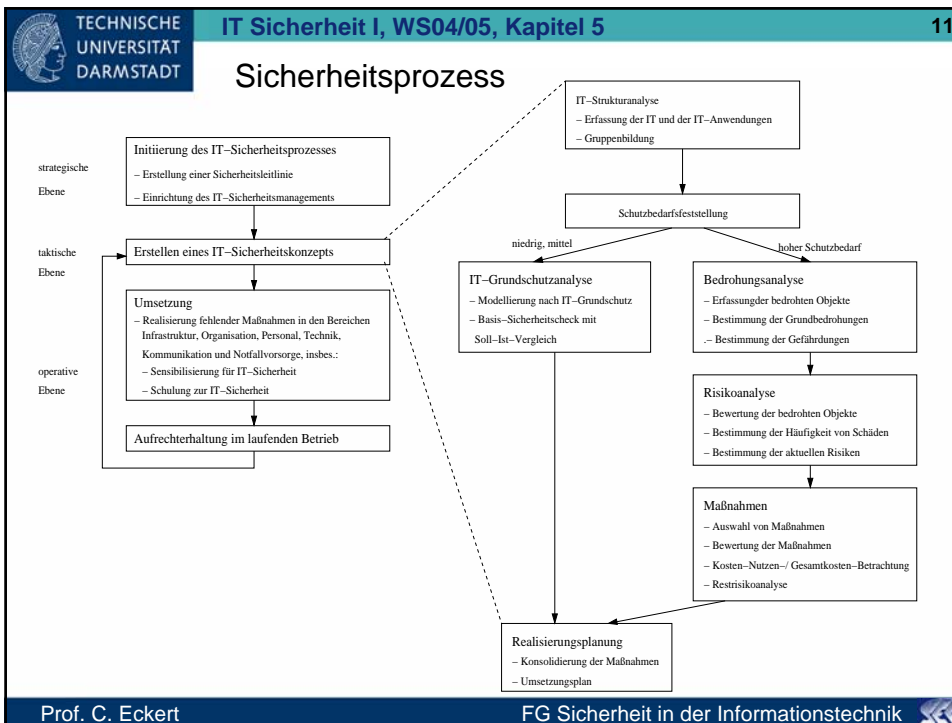
- sehr weit im Einsatz in Unternehmen mit niedrigem, bzw. geringem Schutzbedarf
- > 50 Bausteine beschreiben verschiedene Aspekte der IT-Sicherheit (Server, PC, Firewall, E-Mail, WLAN, ...)
- organisatorische, personelle, infrastrukturelle und technische Standardsicherheitsmaßnahmen
- Gefährdungs- und Maßnahmenkataloge
- **Zertifizierung:** IT-Grundschutzzertifikat (verschiedene Stufen)
 - Umsetzung durch lizenzierten Auditor bestätigt (höchste St.)
- **Weitere Unterstützung durch GSTOOL:** u.a. Hilfen bei Strukturanalyse, Schutzbedarfsfeststellung, Berichterstellung, Revisionsunterstützung, Basissicherheitscheck

5.2. Phasen der Systemkonstruktion (high-level)





- TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 5 10
- ### Verfeinerte Schritte (orientiert am BSI-GSHB): Überblick
1. Strukturanalyse und Pflichtenheft-Erstellung
 2. Ermittlung des Schutzbedarfs
 3. Bedrohungsanalyse
 4. Risikoanalyse
 5. Erstellen einer Sicherheitspolicy
 6. Modellierung des Systems
 7. Entwurf einer Systemarchitektur
 8. Feinentwurf und Implementierung
 9. Validierung und Evaluierung des Systems
 10. Wartung Überprüfung im laufenden Betrieb
- Zu klären:** Welche Methoden sind in welchem Schritt verwendbar?! Schritte im Folgenden genauer
- Prof. C. Eckert FG Sicherheit in der Informationstechnik

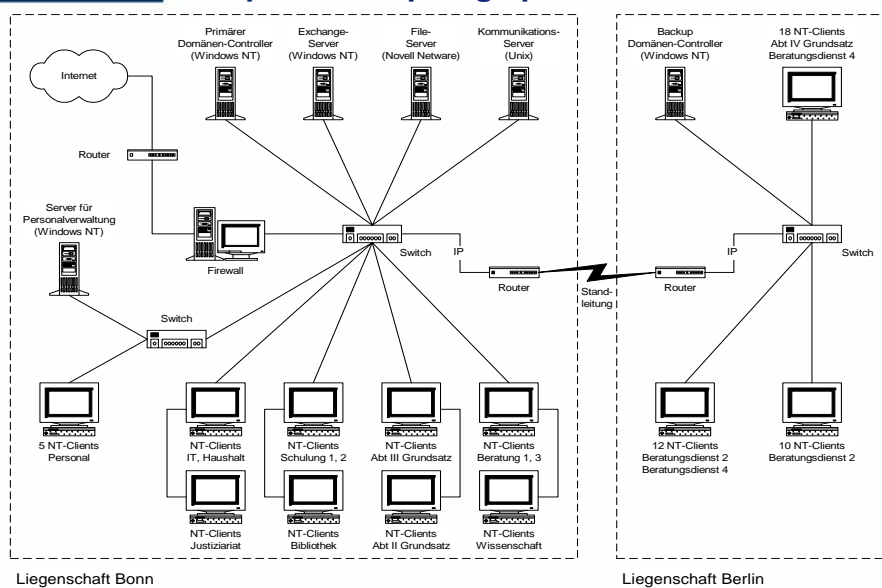


- TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 5 12
- ### 5.3 Schritte des Sicherheitsprozesses im Einzelnen
- #### 5.3.1 Strukturanalyse: Spezifikation des IT-Systems
- Beschreibung der vorhandenen bzw. einzusetzenden Systemkomponenten u. –dienste => Systemfunktionalität
 - Beschreibung der **Systemanforderungen** => Pflichtenheft
 - Beschreibung der **Einsatzumgebung**
 - Erstellung eines **Netztopologieplans**
 - grafische **Übersicht aller Teilkomponenten** (z.B. PCs, Server, DBs, Hubs) und deren Verwendungszweck
 - **Vorhandene Dienste** u. Verbindungen (LAN, WLAN, ...)
 - Ergänzend: **in Tabellenform**: technischer Details (z.B. Hardwareplattform, MAC-Adresse, ...)
- Prof. C. Eckert FG Sicherheit in der Informationstechnik

Beispiel: „Bundesamt für Organisation und Verwaltung“ (BOV) [GSHB]

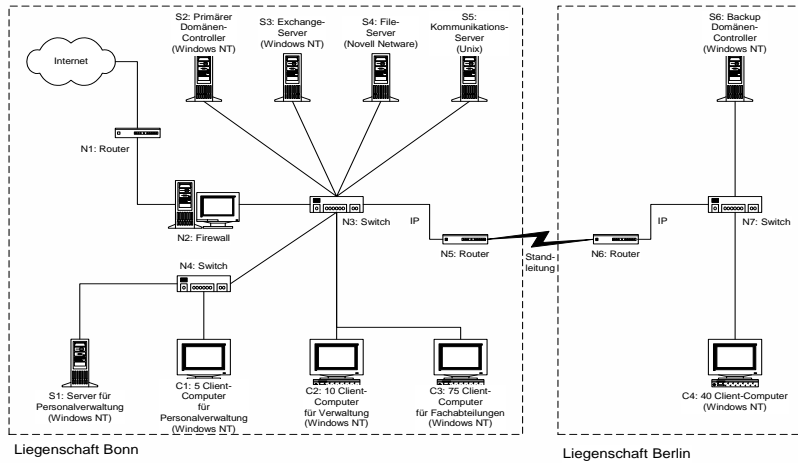
- imaginäre Bundesoberbehörde,
- zwei Standorte: Hauptstelle Bonn und Außenstelle Berlin
- 150 Mitarbeitern, von denen 130 an vernetzten Bildschirmarbeitsplätzen (90 in Bonn und 40 in Berlin) mit Internet-Zugang arbeiten
- angemietete 2 Megabit-Standleitung zwischen Bonn und Berlin
- alle zu Grunde liegenden Normen, Vorschriften, Formulare, Textbausteine und relevanten Arbeitsergebnisse sind in einer zentralen Datenbank vorgehalten
- Entwürfe werden ausschließlich elektronisch erstellt, weitergeleitet und unterschrieben.
- IT-Referat in Bonn zur Realisierung und Betreuung aller benötigten Funktionalitäten

Beispiel Netztopologieplan für BOV [GSHB]



Bereinigter Netztopologieplan: Gruppenbildung

- gleicher Typ, gleich oder nahezu gleich konfiguriert,
- (nahezu) gleiche Netzeinbindung (z. B. am gleichen Switch)
- gleiche Anwendungen, ...



Beispiel BOV, tabellarische Erfassung

Nr.	Beschreibung	Plattform	Anzahl	Aufstellungsort	Status	Benutzer
S1	Server für Personalverwaltung	Windows NT-Server	1	Bonn, R 1.01	in Betrieb	Personalreferat
S2	Primärer Domänen-Controller	Windows NT-Server	1	Bonn, R 3.10	in Betrieb	alle IT-Anwender
S4	File-Server für Arbeitsergebnisse und Grundlagendokumente	Novell 4.x-Server	1	Bonn, R 3.10	in Betrieb	alle IT-Anwender außer Personalreferat
S5	Kommunikationsserver für Intranet	Unix-Server	1	Bonn, R 3.10	in Betrieb	alle IT-Anwender
C1	Gruppe von Clients der Personaldatenverarbeitung	Windows NT-Workstation	5	Bonn, R 1.02 - R 1.06	in Betrieb	Personalreferat
C2	Gruppe von Clients in der Verwaltungsabteilung	Windows NT-Workstation	10	Bonn, R 1.07 - R 1.16	in Betrieb	Verwaltungsabteilung
C3	Gruppe von Clients in den Fachabteilungen II und III	Windows NT-Workstation	75	Bonn, R 2.01 - R 2.75	in Betrieb	Fachabteilung I und II
N5	Router zur Berlin-Anbindung	Router	1	Bonn, R 3.09	in Betrieb	alle IT-Anwender
N6	Router zur Bonn-Anbindung	Router	1	Berlin, E.03	in Betrieb	alle IT-Anwender
N7	Switch	Switch	1	Berlin, E.03	in Betrieb	alle IT-Anwender
T1	TK-Anlage für Bonn	ISDN-TK-Anlage	1	Bonn, B.02	in Betrieb	alle Mitarbeiter in der Hauptstelle Bonn
T2	TK-Anlage für Berlin	ISDN-TK-Anlage	1	Berlin, E.03	in Betrieb	alle Mitarbeiter in der Außenstelle Berlin
N5	Router zur Berlin-Anbindung	Router	1	Bonn, R 3.09	in Betrieb	alle IT-Anwender
...

TECHNISCHE UNIVERSITÄT DARMSTADT		IT Sicherheit I, WS04/05, Kapitel 5		17						
Beispiel BOV: Erfassung der IT-Anwendungen										
Server										
Legende: Sj X Ai bedeutet „Anwendung Ai ist mit dem IT-System Sj verknüpft“										
Beschreibung der IT-Anwendungen			IT-Systeme							
Nr.	IT-Anwendung/Informationen	Pers.-bez. Daten	S 1	S2	S3	S4	S5	S6	S7	
A1	Personaldatenverarbeitung	X	X							
A2	Beihilfeabwicklung	X	X							
A3	Reisekostenabrechnung	X	X							
A4	Benutzerauthentifikation	X		X				X		
A5	Systemmanagement			X						
A6	Exchange (E-Mail, Terminkalender)	X			X					
A7	zentrale Dokumentenverwaltung					X				
A8	Printservice für Bonner Standort					X				
A9	BOV-Intranet						X			
A10	Datenbank der Grundlagendokumente						X			
A11	Printservice für Berliner Standort							X		
A12	Faxservice								X	
A13	Office-Anwendungen (Textverarbeitung, Tabellenkalkulation)									
A14	Internetzugang									
A15	Präsentationsdurchführung									
A16	Filterfunktionalität									
...	...									

Prof. C. Eckert

FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT		IT Sicherheit I, WS04/05, Kapitel 5		18					
Beispiel BOV: Erfassung der IT-Anwendungen									
Clients									
Beschreibung der IT-Anwendungen			IT-Systeme						
Nr.	IT-Anwendung/Informationen	Pers.-bez. Daten	C 1	C2	C3	C4	C5	C6	
A1	Personaldatenverarbeitung	X	X						
A2	Beihilfeabwicklung	X	X						
A3	Reisekostenabrechnung	X	X						
A4	Benutzerauthentifikation	X		X				X	
A5	Systemmanagement			X					
A6	Exchange (E-Mail, Terminkalender)	X	X	X	X	X			
A7	zentrale Dokumentenverwaltung				X	X			
A8	Printservice für Bonner Standort			X	X	X			
A9	BOV-Intranet		X	X	X	X	X		
A10	Datenbank der Grundlagendokumente				X	X	X		
A11	Printservice für Berliner Standort					X		X	
A12	Faxservice		X	X	X	X			
A13	Office-Anwendungen (Textverarbeitung, Tabellenkalkulation)		X	X	X	X	X	X	
A14	Internetzugang		X	X	X	X			
A15	Präsentationsdurchführung						X	X	
A16	Filterfunktionalität								
...	...								

Legende: Sj X Ai bedeutet „Anwendung Ai ist mit dem IT-System Sj verknüpft“

Prof. C. Eckert

FG Sicherheit in der Informationstechnik

Beispiel BOV: Erfassung der IT-Anwendungen

Netzkopplungselemente									
Beschreibung der IT-Anwendungen			IT-Systeme						
Nr.	IT-Anwendung/Informationen	Pers.-bez. Daten	N2	N2	N3	N4	N5	N6	N7
A1	Personaldatenverarbeitung	X	X			X			
A2	Beihilfeabwicklung	X	X			X			
A3	Reisekostenabrechnung	X	X			X			
A4	Benutzerauthentifikation	X		X	X	X	X	X	X
A5	Systemmanagement			X	X	X	X	X	X
..	..								

Telekommunikationskomponenten									
Beschreibung der IT-Anwendungen			IT-Systeme						
Nr.	IT-Anwendung/Informationen	Pers.-bez. Daten	T1	T2	T3	T4			
A1	Personaldatenverarbeitung	X	X			X			
A2	Beihilfeabwicklung	X	X			X			
A3	Reisekostenabrechnung	X	X			X			
A4	Benutzerauthentifikation	X		X	X	X			
A5	Systemmanagement			X	X	X			
..	..								

Legende: Sj X Ai bedeutet „Anwendung Ai ist mit dem IT-System Sj verknüpft“

5.3.2 Schutzbedarfsermittlung

Ziel:

- Feststellung des Schutzbedarfs des IT-Systems
- typischerweise im Hinblick auf Vertraulichkeit, Integrität u. Verfügbarkeit

Vorgehen:

- Schutzbedarfsfeststellung **anhand von Schadensszenarien**, z.B. orientiert an Grundschriftbuch des BSI
- meist **nicht quantitative** sondern **qualitative Aussagen**
- Qualitative Aussagen orientiert an **Schutzbedarfskategorien**: niedriger bis mittlerer, hoher, sehr hoher Bedarf

Schutzbedarfskategorien:

niedrig bis mittel	Die Schadensauswirkungen sind begrenzt und überschaubar
hoch	Die Schadensauswirkungen können beträchtlich sein
sehr hoch	Die Schadensauswirkungen können ein existentiell bedrohliches, katastrophales Ausmaß annehmen

Schadensszenarien: z.B. 6 Szenarien nach BSI GSFB:

- (1) Verstoß gegen **Gesetze/Vorschriften/Verträge**
Beispiele?
- (2) Beeinträchtigung der **persönlichen Unversehrtheit**:
relevant z.B. für medizinische Überwachungsrechner,
Diagnosesysteme, Flugkontrollrechner, Verkehrsleitsysteme

Schadensszenarien (cont.)

- (3) Beeinträchtigung des **informationellen Selbstbestimmungsrechts**

Hintergrund: **Volkszählungsurteil 1983**

Das Grundrecht gewährleistet insoweit die Befugnis des Einzelnen, grundsätzlich selbst über die Preisgabe und Verwendung seiner persönlichen Daten zu bestimmen. Einschränkungen dieses Rechts auf "informationelle Selbstbestimmung" sind nur im überwiegenden Allgemeininteresse zulässig,,

- Erhebung personenbez. Daten ohne Rechtsgrundlage/Einwilligung
- unbefugte Kenntnisnahme bei deren Verarbeitung oder Übermittlung
- unbefugte Weitergabe/Nutzung personenbezogener Daten zu einem anderen, als dem bei der Erhebung zulässigen Zweck
- Verfälschung von personenbezogenen Daten

(4) Beeinträchtigung der **Aufgabenerfüllung**: z.B.

- Fristversäumnisse durch verzögerte Bearbeitung,
- verspätete Lieferung wg verzögerter Bearbeitung,
- fehlerhafte Produktion aufgrund falscher Steuerungsdaten oder
- unzureichende Qualitätssicherung durch Ausfall eines Testsystems

(5) **negative Auswirkungen**: z.B.

- Ansehensverlust einer Behörde bzw. eines Unternehmens,
- Vertrauensverlust gegenüber einer Behörde/Unternehmen,
- verlorenes Vertrauen in die Arbeitsqualität eines Unternehmens
- Einbuße der Konkurrenzfähigkeit

(6) **finanzielle Auswirkungen**: z.B.

- unerlaubte Weitergabe von Forschungsergebnissen,
- Einsichtnahme in Marketingstrategiepapiere, Umsatzzahlen ...
- Zusammenbruch des Zahlungsverkehrs einer Bank

Beispiel (generisch): Ermittlung des Schutzbedarfs hoch bis sehr hoch (vgl. Folien 25,26)

Beispiel: Individuellen Schutzbedarfskategorien für BOV
„niedrig bis mittel“

Der finanzieller Schaden ist kleiner als 25.000,- €

„hoch“

Der finanzieller Schaden: zwischen 25.000,- und 2.500.000,-€

„sehr hoch“

Der finanzieller Schaden ist größer als 2.500.000,- €

Beispiel: Szenarienorientierte Ermittlung eines **hohen-sehr hohen Bedarfs**

Was, wäre, wenn Fragen bezogen auf Kategorien und Schutzziele

(gilt natürlich auch für die Ermittlung eines niedrigen oder mittleren Bedarfs)

Szenario	Schaden und Folgen
(1)	Fundamentaler Verstoß gegen Gesetze und Vorschriften. Vertragsverletzungen, deren Haftungsschäden ruinös sind.
(2)	Mögliche gravierende Beeinträchtigungen der persönlichen Unversehrtheit. Gefahr für Leib und Leben.
(3)	Eine bedeutende Beeinträchtigung des informationellen Selbstbestimmungsrechts des Einzelnen scheint möglich. Möglicher Missbrauch personenbezogener Daten würde für den Betroffenen den gesellschaftlichen und wirtschaftlichen Ruin bedeuten.

Szenario	Schäden u. Folgen
(4)	Beeinträchtigung wird von allen Betroffenen als nicht tolerabel eingeschätzt. Die maximale tolerierbare Ausfallzeit ist kleiner als eine Stunde.
(5)	Eine landesweite Ansehens- und Vertrauensbeeinträchtigung ggf. sogar existenzgefährdender Art ist denkbar.
(6)	Finanzieller Schaden ist für die Institution existenzbedrohend.

5.3.3 Bedrohungsanalyse

Ziele:

- Ermitteln des **Sicherheits-Ist-Zustand**
 - Aufdecken von Schwachstellen, pot. Bedrohungen
 - Nutzung publizierter Schwachstellen (CERT, Hersteller, ...)
- Ermittlung von **Gefährdungsfaktoren** (organisatorische, technische, benutzerbedingte), u.a. mit
 - Penetrationstests

Verwendete Methoden:

- Analyse mit **Bedrohungsbaum/Angriffsbaum**
 - Grafisch oder textuell
 - Nutzung publizierter Attack-Patterns für Standard-Szenarien
- **Bedrohungsmatrix**

Bedrohungsmatrix: Erfassen der Bedrohungen in einer Matrix

Zeilen: (Wichtigste) Problembereiche

- Bedrohungen durch externe Angriffe
- Bedrohungen durch interne Angriffe
- Bedrohungen der Verfügbarkeit, Integrität, Vertraulichkeit
- ...

Spalten: Potentielle Auslöser

- interner Benutzer
- Administrator
- mobiler Code
- ...

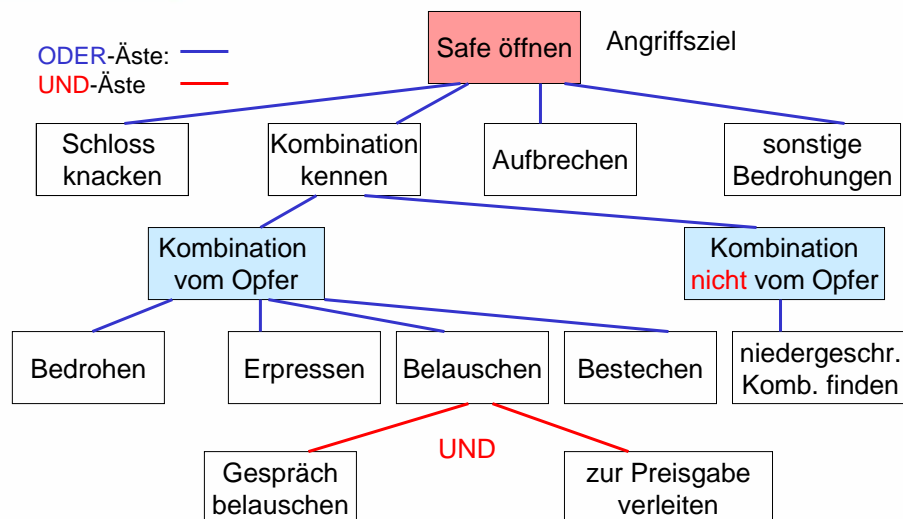
Strukturierte Analyse mit Bedrohungsbaum (attack tree)

- Anlehnung an Fehlerbäume, Bereich der Fehlertoleranz
- **Pro Angriffsziel** ein Baum: (Bsp siehe Folie 30)
 - **Wurzel** beschreibt ein **Angriffsziel**, z.B. Safe knacken
 - **Blatt**: **einzelner Angriffsschritt**, z.B. Schloss knacken
 - **Pfad** von Blatt zur Wurzel: **Angriff** zum Erreichen des Ziels
z.B. durch Erpressung an die Kombination kommen
- Beschreibung von Situationen, in denen
 - mehrere Angriffsschritte **zusammen** notwendig: **UND-Äste**
 - **alternative** Angriffsschritte: **ODER-Äste** (Teilbäume)

Zur systematischen Erstellung sinnvoll:

- Festlegen von Unterzielen, Teilzielen
- Teilziele definieren wieder verwendbare Teilbäume

Beispiel: Bedrohungsbaum



Alternative zur grafischen Darstellung eines Attack-Trees:

Textuelle, kompaktere Darstellung eines Bedrohungsbaums:

- alle direkten Nachfolger werden auf einer Ebene dargestellt
- Ebene wird als **ODER**- bzw. **UND**-Ebene annotiert
- Jede Ebene wird durch ein textuelles Einrücken nachgebildet.

Beispiel: Bedrohungsanalyse für eine verschlüsselte Nachricht

- Szenario: Alice (A) und Bob (B) tauschen verschlüsselte Daten aus,
- Nachricht M ist mit gemeinsamem Schlüssel K_AB verschlüsselt
- **Ziel:** Angreifer will M im Klartext ‚sehen‘

Ausschnitt aus einem textuellen Angriffsbaum:

ODER Subziel 1: Dechiffrieren der verschlüsselten Nachricht

ODER Knacken des Kryptotextes
Kryptoanalyse

Subziel 2: Bestimmung des Schlüssels K_AB

ODER Subziel 2.1: Zugriff auf Speicher von A und
auf dort abgelegten K_AB

Subziel 2.2: Zugriff auf Speicher von B und
auf dort abgelegten K_AB

Subziel 2.3: Verleite A zur Preisgabe von K_AB

Subziel 2.4: Verleite B zur Preisgabe von K_AB

...

Bem: u.a. ist es **einfacher, ein Endgerät anzugreifen**, als aufwändig zu versuchen, einen **Schlüssel zu knacken!**

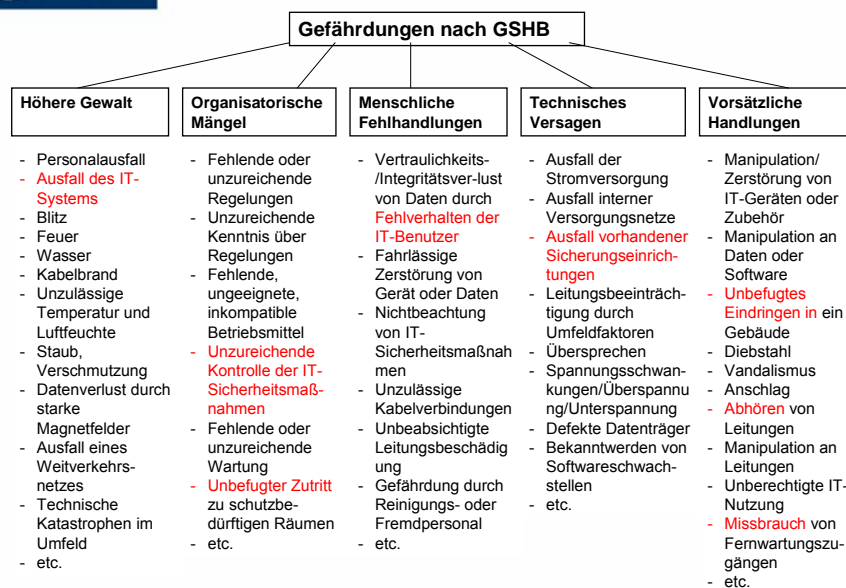
Weitere Methoden zur Identifikation von Bedrohungen

Kollektionsmethoden	Suchmethoden	
	Analytische Methoden	Kreativitätsmethoden
Checklisten Risiko-Identifikations-Matrix (RIM) SWOT-Analyse / Self-Assessment Interview, Befragung	Fragenkataloge Fehlermöglichkeits- und Einflussanalyse (FMEA) HAZOP Baumanalyse Morphologische Verfahren	Brainstorming Brainwriting Delphi-Methode ...

Vorwiegend geeignet zur Identifikation **bestehender** und **offensichtlicher** Risiken

Vorwiegend geeignet zur Identifikation zukünftiger und bisher **unbekannter** Risikopotentiale

Identifikation von Bedrohungen mit Checklisten



Beispiel: HAZOP (Hazard and Operability Studies) Methode zur Risikoidentifikation

- Stammt aus der chemischen Industrie zur Untersuchung von Funktionssicherheit
- „Abweichendes Verhalten“
- Team & **Guidewords**
- Benutzt, um strukturiert Fehler-Bäume aufzustellen

Guidewords

	Datenrate	Bandbreite
<i>Kein</i>	Keine Daten Übertragen	
<i>Mehr</i>	Datenrate zu hoch	Höhere Auslastung als erwartet
<i>Weniger</i>	Datenrate zu Gering	Niedrigere Ausl.
<i>Ebenfalls</i>	Zusätzlicher Kommunikationskanal geöffnet	

SecHAZOP: Erweiterung der HAZOP Methode für IT-Sicherheit (Diplomarbeit)

5.3.4 Risikoanalyse

Ziele: **Bewertung** der Bedrohungen

- Abschätzung des **potentiellen Schadens**
- Abschätzung der **Wahrscheinlichkeit des Eintretens**

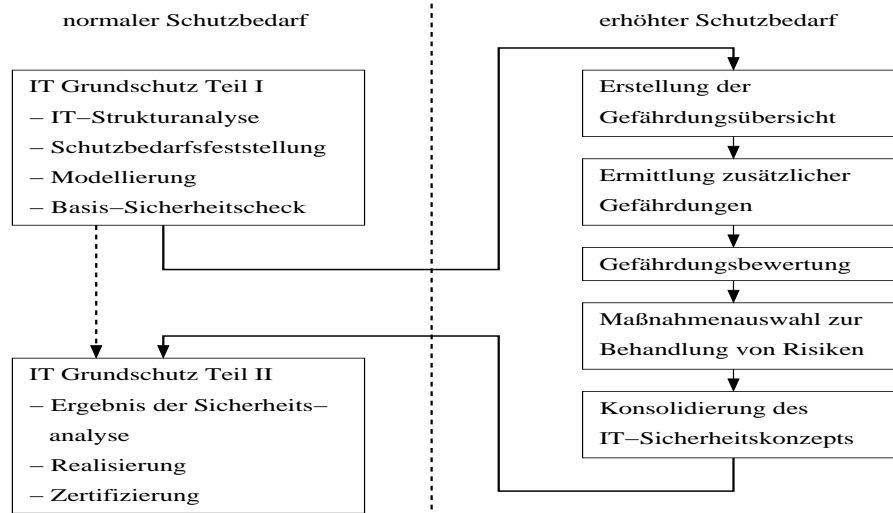
Typische Methoden:

- Kosten-Nutzen Analyse
- quantitative oder qualitative Bewertung der Bedrohungen
- Risikoberechnung, u.a. basierend auf **Angreifer-Modellen** und
- **Penetrationstests**

Bem: Risikoanalyse ist schwierig und aufwändig,

- BSI empfiehlt sie nur bei **hohem und sehr hohem** Bedarf
- Methodik unter: <http://www.bsi.de/gshb/risikoanalyse/risiko.pdf>

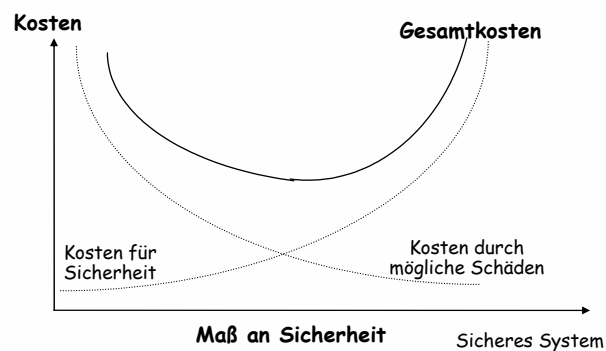
Ergänzende (falls Grundschutz Nicht ausreichend) Risikoanalyse (BSI-GSHB))



Kosten-Nutzen-Analyse:

Bewertung des Risikos

- Was ist wertvoll und **muss geschützt** werden?
- Wie **hoch** sind die zu erwartenden **Schäden**?
- Wie **hoch** sind die **Kosten** für Gegenmaßnahmen?



Ziel: Identifizierte Bedrohungen bewerten!

Risiko nach DIN-VDE Norm 31000-2:

- Quantitatives Risiko $R = S * E$ mit:
 - **Schadenshöhe** (Schadensausmaß) S,
 - **Eintrittswahrscheinlichkeit** E

Schadenshöhe S:

- **Primäre Schäden**: Produktivitätsausfall, Wiederbeschaffungs-Personalkosten, Wiederherstellungskosten, ...
- **Sekundäre Schäden** (**schwer zu quantifizieren**): Imageverlust, Vertrauensverlust bei Kunden,

Beispiele:

S = 1.000.000 €; E = 0,01: R = 10.000 €

S = 1.000.000.000 €; E = 0,001: R = 1.000.000 €

S = 30.000 €; E = 0,5: R = 15.000 €

Eintrittswahrscheinlichkeit E ($0 \leq E \leq 1$): **Wie ermitteln?!**

- Eigene Erfahrungen (z.B. aus Auditprozessen)
- Öffentliche Statistiken (CERT, ...) (was heißt CERT?)
- Einschätzung des Nutzens für den Angreifer u. des Aufwandes
- Durchführung von **Penetrationstestings**, um Aufwand für erfolgreichen Angriff abzuschätzen

Grundlage/Ausgangspunkt: **Erstellen von Angreifermodellen:**

- **Angreifertyp** (Hacker, Script-Kiddie, Wirtschaftsspion, ...),
- **Budget** (Unternehmen, Regierung, Privatperson, ...)
- **Kenntnisse** (keine, Insider-Wissen, Expertenwissen, ...),
- **Ziele** (Gewinn, Schaden, Rache, ...)

Risikoberechnung:

Problem: u.a.

- Erfassung der Abhängigkeiten zwischen einzelnen Risiken

Lösungsansatz:

- Attributierung des Bedrohungs-Baumes mit S- und E-Werten

Attributierung:

- Gegeben sei ein Baum T mit der Menge K seiner Knoten.
- Sei A eine endliche Menge von Attributen (z.B. Aufwand, Kosten)
- Eine Abbildung $f: K \longrightarrow \text{POT}(A)$ nennen wir eine **Attributierung** von T .

Beispiel für eine einfache Attributierung:

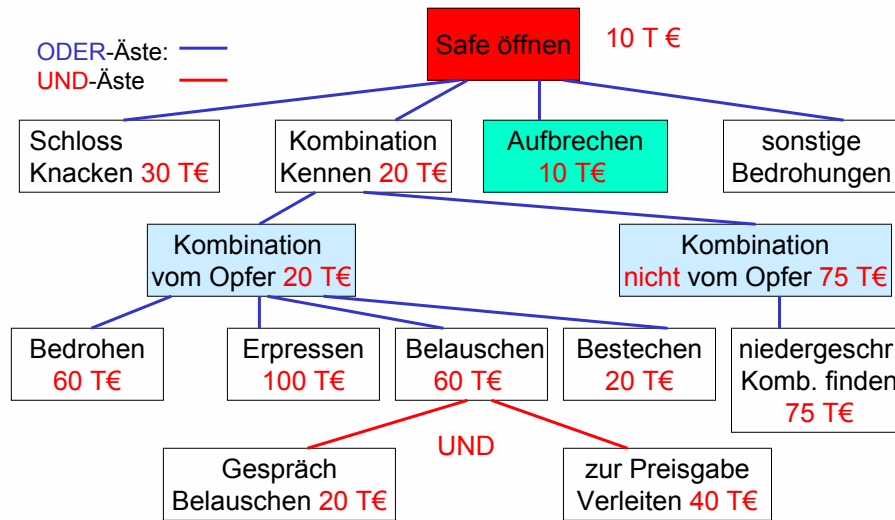
- Gegeben $A = \{U, M\}$, $U =$ unmöglich und $M =$ möglich
- Die Attributierungsabbildung f sei wie folgt definiert:
Für jeden **ODER**-Knoten $k \in K$ gilt:
 - (1) $f(k) = M$ gdw. es gibt $k' \in \text{suc}(k)$: $f(k') = M$ und
 - (2) $f(k) = U$ gdw. für alle $k' \in \text{suc}(k)$: $f(k') = U$, $\text{suc}(k) =$ Menge aller Nachfolger-Knoten von k im Baum

Für jeden **UND**-Knoten $k \in K$ gilt:

- (1) $f(k) = M$ gdw für alle $k' \in \text{suc}(k)$: $f(k') = M$ und
- (2) $f(k) = U$ gdw es gibt $k' \in \text{suc}(k)$: $f(k') = U$

- **Mögliche Angriffe** sind alle diejenigen Pfade von Blättern zur Wurzel, die mit **M attribuiert** sind

Beispiel: Attributierung des Baums von Folie 30



Quantitative Erfassung des Risikos: meist sehr schwierig

Alternative:

Qualitatives Sicherheitsrisiko R

- abhängig von folgenden Größen:
- **Schadenshöhe S:**
 - Kategorisierung auf Skala z.B. von 1 bis 5:
1 = unbedeutend, ..., 5 = existenzbedrohend
- **Eintrittswahrscheinlichkeit E:**
 - Kategorisierung auf Skala von z.B. 1 bis 5:
1 = sehr selten, ..., 5 = sehr wahrscheinlich

Penetrationstests als Methode zur Risikoanalyse

- **Simulation des Verhaltens** eines vorsätzlichen Angreifers
 - Innentätersicht oder Außentätersicht
- **Schwachstellen** und potentielle Schäden ermitteln
- Durchführung häufig durch Externe, z.B. durch **Tiger Teams**

Verschiedene Vorgehensweisen: Ansätze:

- **Blackbox**: keine oder nur geringe Kenntnisse über das System vorhanden (typischerweise **Außentäter**)
- **Whitebox**: detaillierte Kenntnisse über interne Strukturen, Anwendungen, Dienste etc. (**Innentäter**)

Typische Vorgehensweise bei einem Penetrationstest

- (Internet) **Recherche**: Ermittlung von frei zugängliche Informationen über den Untersuchungsgegenstand, **Bsp.?**
- Ermittlung **offener Ports** (Portscanner etc.): Rückschluss auf daran gekoppelte Anwendungen/Dienste, **Bsp?**
- **Fingerprinting**: Ermittlung von Informationen über das verwendete Betriebssystem (u.a. Version, Browser, Hardware)
- Nutzung von **Expertenwissen in Schwachstellendatenbanken**: Identifikation von Schwachstellen des Zielsystems
- **Angriffe** auf das Zielsystem, Ausnutzen der Schwachstellen: **Ziel** u.a.
 - **unberechtigten Zugriff** auf das Zielsystem erlangen,
 - **Vorbereitung** weiterer Angriffe

Typische Angriffe im Rahmen von Penetrationstests u.a.

- Angriffe durch Erraten von Passwörtern oder durch Wörterbuchattacken
- Angriffe durch Aufzeichnen des Netzverkehrs
- Angriffe durch Manipulieren des Netzverkehrs und
- Angriffe durch Einspielen gefälschter Datenpakete
- Angriffe durch Ausnutzen bekannter Software-Schwachstellen:
 - Makro-Sprachen,
 - Betriebssystemfehler, BOs,
 - Remote-Dienste, etc.

Analyse-Tools für Windows/Unix: (siehe Buch) u.a.

Name	Funktion	Plattform	URL	Was noch bekannt?
Nmap	Portscanner	beide	http://www.insecure/nmap	
Super Scan	Portscanner	Windows	http://www.computech.ch	
Nessus	Schwachstellen-Scanner	beide	http://www.nessus.org	
SARA	Freeware von Saint	Unix	http://www-arc.com/sara	
Whisker	Lücken in CGI-Skripts	Unix	sourceforge.net/projects/whisker	
Kismet	Auffinden/Abhören von WLANs	Unix	http://www.kismetwireless.net	
Firewalk	Testen von Firewallregeln	Unix	http://www.packetfactory.net	
Dsniff	Abhören auch geswitchter N.	Unix	http://www.monkey.org	
Snort	IDS, Sniffer	beide	http://www.snort.org	

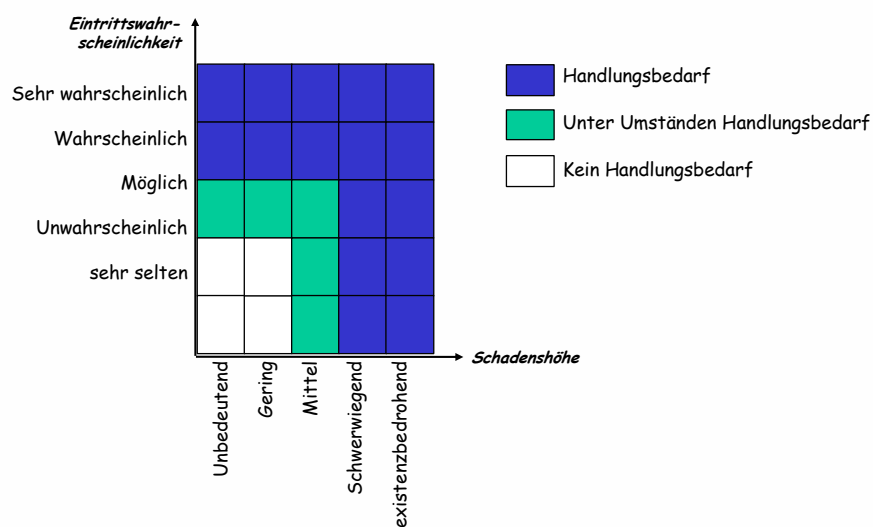
Bem.:

Durchführung von Penetrationstests ist **nicht unproblematisch!**

- Tests müssen vertraglich mit AG abgesichert sein
- Eindringen in ein System durch das Testteam darf **produktiven Betrieb nicht nachhaltig stören**,
- durch Manipulation dürfen **keine irreparablen Schäden** entstehen
- Beachtung **datenschutzrechtlicher Belange**: Absprache mit dem Betreiber des Zielsystems und Datenschutzbeauftragten
- **Zugangskontrolldiensteschutzgesetz (ZKDSG)** beachten:
 - z.B. Umgehen eines passwortgeschützten WWW-Server:
 - Verstoß gegen das ZKDSG
 - weitere Infos u.a. unter <http://www.bsi.de>

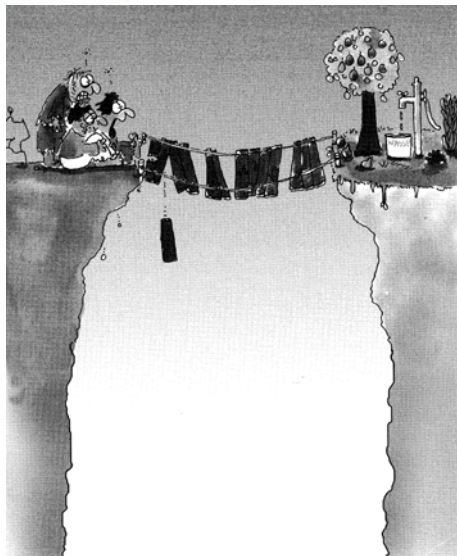


Risikoanalyse zeigt Handlungs-Bedarf auf



Fazit Risikoanalyse

- Risikoanalyse ist ein aufwändiges Verfahren,
- Analyse verlangt Expertenwissen:
 - über schützenswerte Güter, Unternehmensprozesse.
 - über Angreifer, über Sicherheitstechnologien, ...
- Quantitative Risikoanalyse: in der Praxis kaum möglich, aber oft vom Management gewünscht: RoSI-Werte
- **RoSI** Return on Security Invest, gewünscht:
 - $\text{RoSI} = \text{Recovery-Kosten} - \text{ALE (Annual Loss Expentancy)}$
 - $\text{ALE} = \text{Recovery-Kosten} - \text{Ersparnis} + \text{Investition}$
- Aber: *„Nicht alles was zählt kann gezählt werden, und nicht alles was gezählt werden kann zählt.“* Albert Einstein
- In der Regel **qualitative** Sicherheitsanalyse:
 - aber gg. stark von subjektiven Einflüssen abhängig.
 - Durchführung **durch Team**, nicht einzelne Personen!



Leben ist Risiko.
Was wir tun, ist riskant.
Was wir nicht tun, ist es auch.

5.3.5 Erstellen einer Sicherheitspolicy

- Bedrohungs- und Risikoanalyse beschreibt **Ist-Zustand**
- Schutzbedarfsermittlung beschreibt **Soll-Zustand**
- Falls **Ist** \neq **Soll**, dann sind Maßnahmen zur Behebung der Defizite durchzuführen
- Festlegen einer **Sicherheitsstrategie**: informelle oder formale Definition geeigneter Regeln und Maßnahmen
- Präzisierung durch Modellbildung (Schritt 6)

Vorab: Erstellen einer **IT-Sicherheitsleitlinie**

- Angaben zum anzustrebenden Sicherheitsniveau.
- legt **verbindliche Prinzipien** für die IT-Sicherheit des Unternehmens fest

Beispiel für eine Sicherheitsleitlinie (Auszug)

Die für das Unternehmen relevanten Gesetze und Vorschriften sowie vertragliche und aufsichtsrechtliche Verpflichtungen müssen eingehalten werden.

Jeder Mitarbeiter soll im Rahmen seines Umgangs mit IT (als Nutzer Berater, Geschäftspartner) die erforderliche Integrität und Vertraulichkeit von Informationen und (wenn erforderlich) Verbindlichkeit und Beweisbarkeit von Geschäftskommunikation gewährleisten und die Richtlinien des Unternehmens einhalten.

Erkannte Fehler sind den Zuständigen umgehend zu melden, damit schnellst-möglich Abhilfemaßnahmen eingeleitet werden können.

Erstellen einer Sicherheitspolicy (vgl. Kapitel 2, Folien 21 ff)

Festlegen:

- der **Schutzziele** und einer Menge von technischen und organisatorischen **Regeln**,
- von **Verfahren** und **Verhaltensrichtlinien**
- von Maßnahmen zur Gewährleistung der Schutzziele, um das in der Analyse ermittelte, angestrebte Sicherheitsniveau zu erzielen
- Festlegen von **Verantwortlichkeiten** und **Rollen**

Beispiel: Auszug aus einer Policy für ein Krankenhausszenario

- Vgl.: <http://info.imsd.uni-mainz.de/AGDatenschutz/Empfehlungen/Zugriff.html>

Beispiel: Zugriffsrechte bei einem typischen Szenario

Aufnahme (textuelle Beschreibung des Workflows)

Die reguläre Aufnahme eines Patienten erfolgt **durch einen dazu berechtigten Verwaltungsmitarbeiter**.

Bei der Aufnahme wird der Patient, in der Regel durch einen separaten Vordruck, auf seine Rechte bezüglich des Datenschutzes hingewiesen und darüber informiert, **welche Daten von wem verarbeitet werden**, und stimmt mit seiner Unterschrift der Speicherung und Verarbeitung seiner Daten zu.

Er wird durch den aufnehmenden Verwaltungsmitarbeiter einer Fachabteilung zugewiesen. **Dadurch werden die Zugriffsrechte für diese Fachabteilung freigegeben**.

Handelt es sich um eine Wiederaufnahme, d. h. wurde der Patient schon einmal in diesem Krankenhaus behandelt, **darf bei der Aufnahme auf bereits vorhandene Stammdaten zugegriffen** werden.

Bei Identifizierungsproblemen ist eine minimierte Auswahl an Fällen und identifizierenden Daten anzubieten, auf **keinen Fall aber eine vollständige Patientenliste** zur Auswahl.

Mehrere **Abstraktionsebenen** der Sicherheitspolitik:

- von abstrakt formulierte Sicherheitspolitik (z.B. Informell **textuelle** Sicherheitspolitik) (siehe Beispiel)
- bis zu konkret formulierte Sicherheitspolitik (z.B. **formal spezifizierte** Sicherheitspolitik).

Regeln: Ereignis, Bedingung & Aktion, z.B.

- **Ereignis:** Benutzerzugriff
- **Bedingung:** Kontextinformation
- **Aktion:** Erlaubte Aktion

Probleme: textuelle Policies:

- ggf unpräzise, missverständlich, inkonsistent
- Vergleich/Abgleich von Policies ist sehr schwierig (z.B. kooperatives Arbeiten, Austausch von Dokumenten)

Formalisierte Policy: **Policy-Sprachen, z.B. XACML, SAML, ...**

- Sprachkonzepte, um
 - ▶ die Zugriffskontrolle eines Subjekts
 - ▶ die Flusskontrolle einer Information (bzw. eines Objektes) zu regeln
- Eine Policy-Sprache bietet idR folgende Konzepte:
 - ▶ Positive Verpflichtung } Wird ständig (oder periodisch) durchgesetzt oder überwacht
 - ▶ Negative Verpflichtung }
 - ▶ Positive Autorisierung } Wird bei einem Ereignis durchgesetzt oder überwacht
 - ▶ Negative Autorisierung }
- Durch die Konzepte können folgende Konflikte in Policies auftreten:
 - ▶ Positive Verpflichtung <> Negative Verpflichtung
 - ▶ Positive Autorisierung <> Negative Autorisierung
 - ▶ Positive Verpflichtung <> Negative Autorisierung
 - ▶ Positive Autorisierung <> Negative Verpflichtung
- Notwendig: Regeln bzw. Maßnahmen zur Konfliktlösung von Sicherheitspolitiken

Beispiel einer Sicherheitspolitiksprache: XACML

- XACML wurde von OASIS entwickelt
- Repräsentiert in XML Syntax
- Regelt die Autorisierung von Aktionen des Subjektes
- Hat 3 Konstrukten: Rules, Policy, PolicySet
- Rules:
 - ▶ Rule Target: Definiert: Subjekt, Objekt und die Aktion
 - ▶ Effect: *PERMIT* oder *DENY*
 - ▶ Condition: Kontext-Bedingungen bool'schen Ausdrücke (für Zugriffe)
- Policy
 - ▶ Policy Target: Definiert Subjekt, Objekt und die Aktion
 - ▶ Rule-combining algorithm-identifier: Evaluierungsregeln
Bsp: Deny-Overrides
 - ▶ Set of Rules: eine Menge von Rule-Konstrukten
 - ▶ Obligation

Beispiel: XACML

```
<Policy PolicyId="SamplePolicy"
  RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-
  algorithm:permit-overrides">

  <!-- This Policy only applies to requests on the SampleServer -->
  <Target>
    <Subjects>
      <AnySubject/>
    </Subjects>
    <Resources>
      <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-
      equal">
        <AttributeValue
          DataType="http://www.w3.org/2001/XMLSchema#string">SampleServer</AttributeVa
          lue>
          <ResourceAttributeDesignator
            DataType="http://www.w3.org/2001/XMLSchema#string"
            AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"/>
        </ResourceMatch>
      </Resources>
    <Actions>
      <AnyAction/>
    </Actions>
  </Target>
```

Beispiel: XACML

```
<!-- Rule to see if we should allow the Subject to login -->
<Rule RuleId="LoginRule" Effect="Permit">

  <!-- Only use this Rule if the action is login -->
  <Target>
    <Subjects>
      <AnySubject/>
    </Subjects>
    <Resources>
      <AnyResource/>
    </Resources>
    <Actions>
      <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-
equal">
        <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">login</AttributeValue>
        <ActionAttributeDesignator
DataType="http://www.w3.org/2001/XMLSchema#string"
AttributeId="ServerAction"/>
      </ActionMatch>
    </Actions>
  </Target>
```

Beispiel: XACML

```
<!-- Only allow logins from 9am to 5pm -->
<Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:and">
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-greater-than-or-equal"
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-one-and-only">
    <EnvironmentAttributeSelector DataType="http://www.w3.org/2001/XMLSchema#time"
AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time"/>
  </Apply>
  <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#time">09:00:00</AttributeValue>
  </Apply>
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-less-than-or-equal"
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-one-and-only">
    <EnvironmentAttributeSelector DataType="http://www.w3.org/2001/XMLSchema#time"
AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time"/>
  </Apply>
  <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#time">17:00:00</AttributeValue>
  </Apply>
  </Condition>
</Rule>

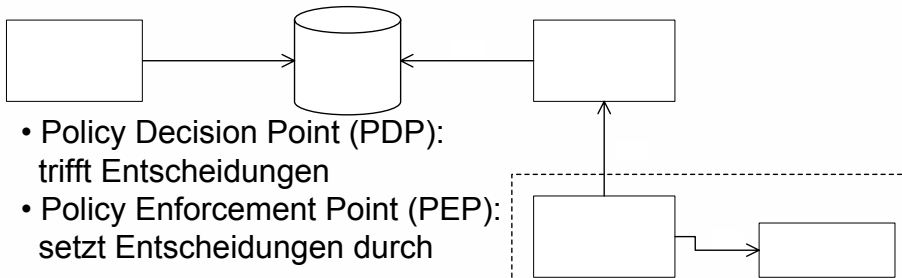
<!-- We could include other Rules for different actions here -->

<!-- A final, "fall-through" Rule that always Denies -->
<Rule RuleId="FinalRule" Effect="Deny"/>

</Policy>
```

Systematische Erstellung von Policies: Policy-Engineering

- Noch **kaum Methoden, keine Tools** dafür
- am Lehrstuhl: Projekt: SicaRI, 2 laufende Dissertationen, u.a.
 - Entwickeln von **Policy-Patterns** (a la Design-Patterns)
 - Methoden zur **Konfliktauflösung** (Meta-Policies)
 - Entwicklung von Policy-gesteuerten Systemen (Framework)
 - Policy Repository: speichert Regeln



- Policy Decision Point (PDP): trifft Entscheidungen
- Policy Enforcement Point (PEP): setzt Entscheidungen durch

5.3.6 Modellierung

Ziel:

- formale Beschreibung der Sicherheitseigenschaften
- verschiedene **Sicherheitsmodelle** im Einsatz:
 - Subjekt/Objekt-zentriert (einfache Zugriffskontrolle)
 - Rollen-basiert oder auch Informationsfluß-basiert
- Sicherheitsmodelle werden in Kapitel 6 ausführlicher behandelt

Hier: einfacher Ansatz zur **Modellierung basierend auf GSHB**

- **Modellierung nach GSHB**: Nachbilden des IT-Systems mit Hilfe von **Bausteinen**, die im Grundschutzhandbuch vorgegeben sind.

Grundschutzmodell

- **beschreibt Bausteine**, um den Grundschutz zu gewährleisten

Beispiele für solche Bausteine:

- Laptops, Unix-PC, Windows-PC, Telearbeit, Server, Modem,
- Verkabelung, Viren-Schutzkonzept, Notfallversorgung, ...

Für jeden Baustein:

- Beschreibung **typischer Gefährdungen** für den Baustein und
- Empfehlungen für **anzuwendende Maßnahmen**
- Angabe von Abwehrmaßnahmen bezogen auf: Infrastruktur, Personal, Organisation, Hard-Software, Kommunikation, Notfallvorsorge

Beispiel: Baustein 7.2 Modem (aus BSI-Grundschutzhandbuch)

Beschreibung

Über ein Modem wird eine Dateneneinrichtung, z. B. ein PC, über das öffentliche Telefonnetz mit anderen Dateneneinrichtungen verbunden, um Informationen austauschen zu können. Ein Modem wandelt die digitalen Signale aus der Dateneneinrichtung in analoge elektrische Signale um, die über das Telefonnetz übertragen werden können. Damit zwei IT-Systeme über Modem kommunizieren können, muss auf den IT-Systemen die entsprechende Kommunikationssoftware installiert sein. Unterschieden werden externe, interne und PCMCIA-Modems. Ein externes Modem ist ein eigenständiges Gerät mit eigener Stromversorgung, das üblicherweise über eine serielle Schnittstelle mit dem IT-System verbunden wird. Als internes Modem werden Steckkarten mit Modem-Funktionalität, die über keine eigene Stromversorgung verfügen, bezeichnet. Ein PCMCIA-Modem ist eine scheckkartengroße Einsteckkarte, die über eine PCMCIA-Schnittstelle üblicherweise in Laptops eingesetzt wird.

Gefährdungslage

In diesem Kapitel werden für den IT-Grundschutz beim Einsatz eines Modems folgende Gefährdungen angenommen:

Gefährdungslage des Bausteins

Höhere Gewalt:

- G 1.2 Ausfall des IT-Systems

Menschliche Fehlhandlungen:

- G 3.2 Fahrlässige Zerstörung von Gerät oder Daten
- G 3.3 Nichtbeachtung von IT-Sicherheitsmaßnahmen
- G 3.5 Unbeabsichtigte Leitungsbeschädigung

Technisches Versagen:

- G 4.6 Spannungsschwankungen/Überspannung/Unterspannung

Vorsätzliche Handlungen:

- G 5.2 Manipulation an Daten oder Software
- G 5.7 Abhören von Leitungen
- G 5.8 Manipulation an Leitungen
- G 5.9 Unberechtigte IT-Nutzung
- G 5.10 Missbrauch von Fernwartungszugängen
- G 5.12 Abhören von Telefongesprächen und Datenübertragungen
- G 5.18 Systematisches Ausprobieren von Passwörtern
- G 5.23 Computer-Viren
- G 5.25 Maskerade
- G 5.39 Eindringen in Rechnersysteme über Kommunikationskarten
- G 5.43 Makro-Viren

Beispiel einer Gefährdungsbeschreibung laut Handbuch:

G5.9 Unberechtigte IT-Nutzung

Ohne Mechanismen zur Identifikation und Authentisierung von Benutzern ist die Kontrolle über unberechtigte IT-Nutzung praktisch nicht möglich. Selbst bei IT-Systemen mit einer Identifikations- und Authentisierungsfunktion in Form von Benutzer-ID- und Passwort-Prüfung ist eine unberechtigte Nutzung denkbar, wenn Passwort und zugehörige Benutzer-ID ausgespäht werden. Um das geheim gehaltene Passwort zu erraten, können Unbefugte innerhalb der Login-Funktion ein mögliches Passwort eingeben. Die Reaktion des IT-Systems gibt anschließend Aufschluss darüber, ob das Passwort korrekt war oder nicht. Auf diese Weise können Passwörter durch Ausprobieren erraten werden. Viel Erfolg versprechender ist jedoch die Attacke, ein sinnvolles Wort als Passwort anzunehmen und alle Benutzereinträge durchzuprobieren. Bei entsprechend großer Benutzeranzahl wird damit oft eine gültige Kombination gefunden. Falls die Identifikations- und Authentisierungsfunktion missbräuchlich nutzbar ist, so können sogar automatisch Versuche gestartet werden, indem ein Programm erstellt wird, das systematisch alle möglichen Passwörter testet.

Beispiel:

1988 nutzte der Internet-Wurm eine Schwachstelle der betroffenen Unix-Betriebssysteme aus, um gültige Passwörter zu finden, obwohl die gültigen Passwörter verschlüsselt gespeichert waren.

Maßnahmenempfehlungen für den Baustein:

Infrastruktur:

- M 1.25 (3) Überspannungsschutz (optional)
- M 1.38 (1) Geeignete Aufstellung eines Modems

Organisation:

- M 2.25 (2) Dokumentation der Systemkonfiguration
- M 2.42 (2) Festlegung der möglichen Kommunikationspartner
- M 2.46 (2) Geeignetes Schlüsselmanagement (optional)
- M 2.59 (1) Auswahl eines geeigneten Modems in der Beschaffung
- M 2.60 (1) Sichere Administration eines Modems
- M 2.61 (2) Regelung des Modem-Einsatzes
- M 2.204 (1) Verhinderung ungesicherter Netzzugänge

Personal:

- M 3.17 (1) Einweisung des Personals in die Modem-Benutzung

Hardware/Software:

- M 4.7 (1) Änderung voreingestellter Passwörter
- M 4.30 (2) Nutzung der in Anwendungsprogrammen angebotenen Sicherheitsfunktionen
- M 4.33 (1) Einsatz eines Viren-Suchprogramms bei Datenträgeraustausch und Datenübertragung
- M 4.34 (2) Einsatz von Verschlüsselung, Checksummen oder Digitalen Signaturen (optional)

- M 4.44 (2) Prüfung eingehender Dateien auf Makro-Viren
- ### Kommunikation:
- M 5.30 (1) Aktivierung einer vorhandenen Callback-Option
 - M 5.31 (1) Geeignete Modem-Konfiguration
 - M 5.32 (1) Sicherer Einsatz von Kommunikationssoftware
 - M 5.33 (1) Absicherung der per Modem durchgeführten Fernwartung
 - M 5.44 (2) Einseitiger Verbindungsaufbau (optional)

Beispiel einer Maßnahmenbeschreibung im Handbuch: M 4.7

M 4.7 Änderung voreingestellter Passwörter

Verantwortlich für Initiierung: TK-Anlagen-Verantwortlicher, IT-Sicherheitsmanagement, Leiter IT

Verantwortlich für Umsetzung: Administrator

Viele IT-Systeme, TK-Anlagen und Netzkoppelemente (bspw. ISDN-Router, Sprach-Daten-Multiplexer etc.) besitzen nach der Auslieferung durch den Hersteller noch voreingestellte Standardpasswörter. Diese sollten als erstes durch individuelle Passwörter ersetzt werden. Hierbei sind die einschlägigen Regeln für Passwörter zu beachten (vgl. M 2.11 *Regelung des Passwortgebrauchs*).

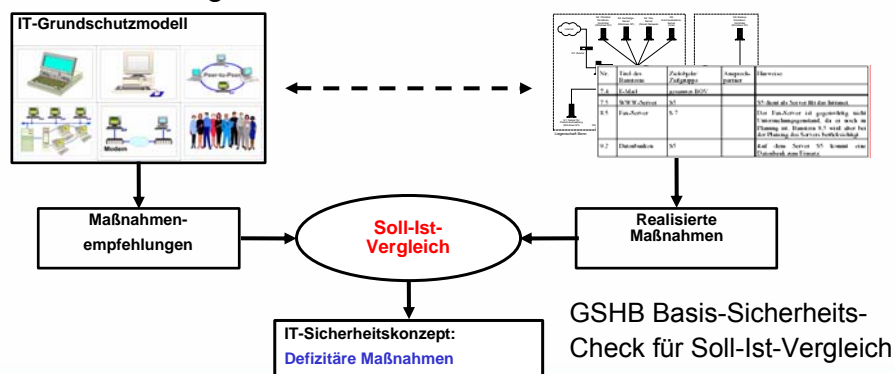
Achtung: Bei einigen TK-Anlagen werden vorgenommene Änderungen der Konfiguration nur im RAM abgelegt. Dies gilt auch für Passwortänderungen. Daher ist nach einer solchen Operation stets eine Datensicherung vorzunehmen und eine neue Sicherungskopie zu erstellen. Unterbleibt dies, so ist nach einem "Restart" der Anlage wieder das Standardpasswort gültig. Weiterhin sollte überprüft werden, ob nach Einrichten eines neuen Passworts das Standardpasswort tatsächlich seine Gültigkeit verloren hat und nicht weiterhin für den Systemzugang genutzt werden kann.

Ergänzende Kontrollfragen:

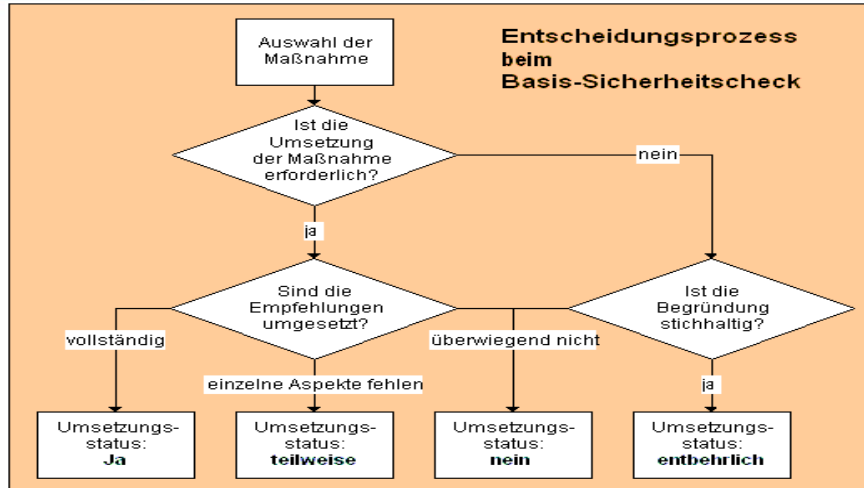
- Ist die Anlage noch mit einem Standardpasswort versehen?
- Wurden die Sicherungskopien nach der Vergabe und Speicherung des individuellen Passworts angelegt?
- Ist der Systemzugang mit dem Standardpasswort nach der Eingabe eines neuen Passworts weiterhin möglich?
- Werden die einschlägigen Regeln zum "Passwort-Handling" beachtet?

Ergebnis der GSHB-Modellierung

- Abbildung auf Bausteine, das liefert geeignete
- Basis für die nächsten Schritte, nämlich
 - Erstellung einer Sicherheits-Architektur und
 - Umsetzung der Maßnahmen



Formulare im GSHB als Hilfsmittel

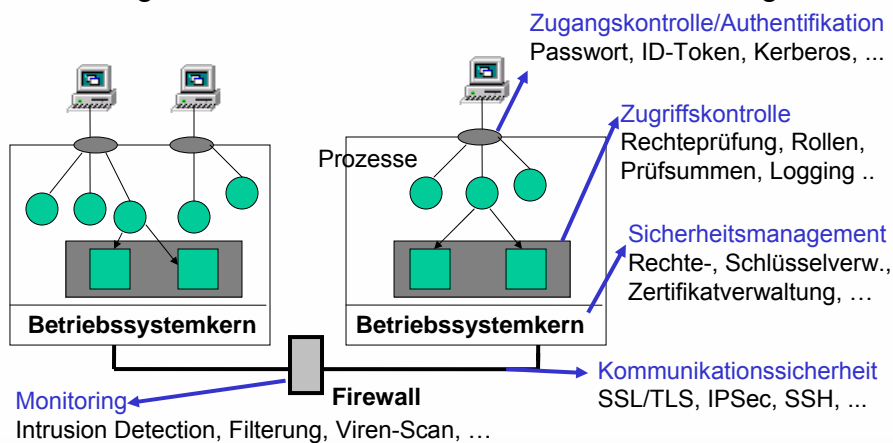


Erkannte Defizite durch Sicherheitskonzept beheben

- Architektur-Design, Auswahl von Maßnahmen, Komponenten, ..

5.3.7- 8 Design und Implementierung

- Mechanismen, Dienste, Protokolle zur Umsetzung der Sicherheitspolicy und des Modells
- Verwaltung sicherheitsrelevanter Informationen, Management



Allgemeine Prinzipien für Konstruktion/Betrieb sichere Systeme

- **Erlaubnis-Prinzip** (*fail-safe defaults*)
Zugriffe sind verboten, falls nicht explizit erlaubt
- **Vollständigkeits-Prinzip** (*complete mediation*)
Jeder Zugriff ist zu kontrollieren
- Prinzip der **minimalen Rechte** (*need-to-know*)
Subjekt erhält nur die benötigten Rechte
- Prinzip des **offenen Entwurfs** (*open design*)
Geheimhaltung darf nicht Voraussetzung für Sicherheit sein
- **Benutzerakzeptanz** (*economy of mechanism*)
einfach zu nutzende Mechanismen, Verfahren

Beispiele für die Prinzipien in heutigen Systemen?

Frage: was ist bekannt an Grundfunktionen?

Design der Sicherheits-Architektur:

Implementierung benötigter **Sicherheitsgrundfunktionen**

- **Identifikation und Authentifikation:**
Welche Subjekte/Objekte, wann authentifizieren;
Welche Aktionen bei Maskierungsangriffen durchführen
- **Rechteverwaltung:**
Welche Subjekte, welche Rechte, in welchem Kontext
wer darf, wann, welche Rechte ändern, vergeben
- **Rechteprüfung:**
Wann, bei welchen Aktionen, welche Kontrollen durchführen,
welche Ausnahmen sind vorgesehen;
welche Aktionen bei unautorisierten Zugriffsversuchen
z.B. Alarm auf Operatorkonsole ausgeben, Log-Eintrag, ...

- **Beweissicherung:**

Welche Ereignisse, wie protokollieren;

Wer darf wann, wie auf welche protokollierte Daten zugreifen
Maßnahmen zur Computer Forensik, **Bedeutung?**

- **Wiederaufbereitung:**

Welche gemeinsam genutzten Objekte, wie wiederaufbereiten

- **Gewährleistung der Funktionalität:**

Welche Funktionalität mit welcher Priorität anbieten,
unter welchen Randbedingungen ist Funktionalität verzichtbar

Für Systemarchitektur:

- Nutzung vorhandener Betriebssystemdienste, falls möglich
- ggf. Härtung vorhandener Betriebssysteme (was heißt das?)

5.3.9 Evaluierung

- bekannte Test- und Evaluierungsmethoden aus dem Software-Engineering: Komponententest, Integrationstest, ...
- aber auch spezielle Sicherheitszertifizierung möglich
 - internationale Bewertungskriterien, Bsp. Common Criteria
 - BSI-Grundschutz-Zertifikat

5.3.10 Wartung und Prüfung im laufenden Betrieb

- zZ noch sehr viel Patch-Management
- einfache Analyse-Tools: IDS, Viren-Scanner, Portscanner, ...
- Umfassende methodische Ansätze und Werkzeuge fehlen!
- Beispiel eines komplexeren Analysetools: **eSI**

Beispiel: Überprüfung im laufenden Betrieb

eSI: Elektronischer Sicherheitsinspektor: entwickelt am FhI-SIT

- kontinuierliche Prüfung der Wirksamkeit umgesetzter Schutz-Maßnahmen, automatisch!
- Basis: elektronische Checklisten/Regelwerke (Policies)
- eSI ist im SIT selber im Einsatz, Lizenzen in Unternehmen
- Ziel/Arbeit kontinuierliche Weiterentwicklung: **Diplomarbeiten!**

Ansatz für eSI: automatisierte Analyse von Maßnahmen:

Zu klärende Fragestellungen:

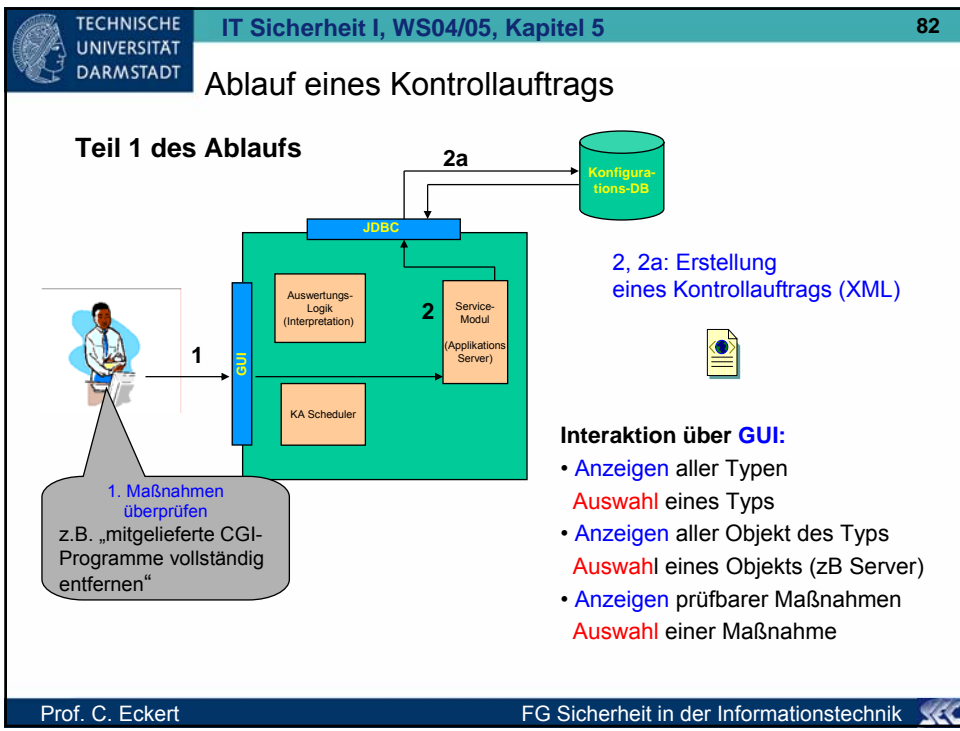
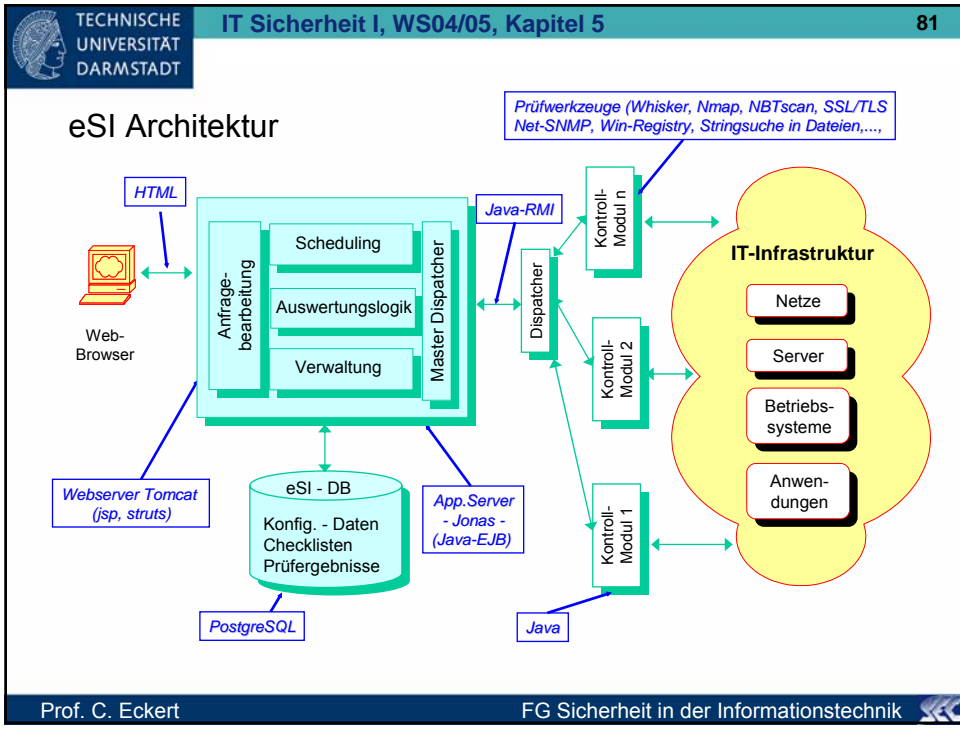
- Welche Maßnahmen sind technisch prüfbar?
- Welche Prüfmethoden, Tools, Skripte sind notwendig?
- Welche Ergebnisse werden jeweils erwartet, wie ist auf Abweichungen zu reagieren (Abweichungen erkennen!)

eSI Maßnahmenüberprüfungen:

- Grundschutzmaßnahmen des GSHB
- SANS Top 20 Schwachstellen
- individuelle Maßnahmen (z.B. mobile Endgeräte)

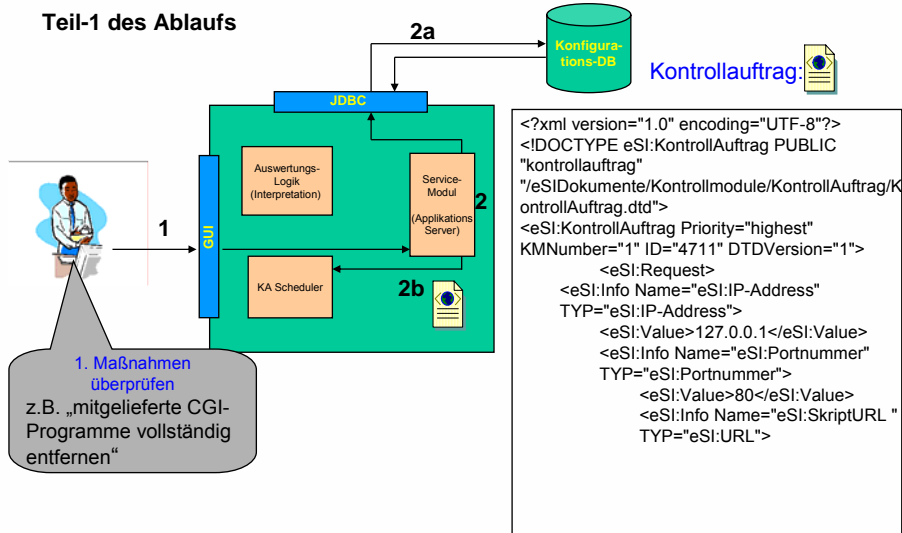
Einige Maßnahmen-Beispiele

- | | |
|--|-------------------------------|
| ▪ „Abschalten von DNS“ | GSHB M4.96 V07/99 |
| ▪ „Ein Dienst pro Server“ | GSHB M4.97 V07/99 |
| ▪ „Standardscripte“ <=> MnNr.5 | SANS G7 V2.504 |
| ▪ „Port 111 (RPC) blockieren“ | SANS U1 V3.21 |
| ▪ „Problematische Parameter bei Samba“ | GSHB M5.82 V10/00 |
| ▪ „regelm. Aktualisierung d. Virensuchprogr.“ | GSHB M4.3 V07/99 |
| ▪ „transienter Betrieb des Virensuchprogramms“ | GSHB M4.3 V07/99 |
| ▪ „residenter Betrieb des Virensuchprogramms“ | GSHB M4.3 V07/99 |
| ▪ „Virens Scannerkonfiguration nach Vorgabe“ | Spezial-SKE M0.006 |
| ▪ „»Verzeichnisinhalt auflisten« deaktivieren“ | GSHB M2.174 V10/03 |
| ▪ „Symbolische Links deaktivieren“ | GSHB M2.174 V10/03 |
| ▪ „IP-Forwarding deaktivieren“ | GSHB M4.95 V07/99 |
| ▪ „Keinen CGI-Support für Web-Server die das nicht brauchen“ | SANS G7 V2.504 konfigurieren, |

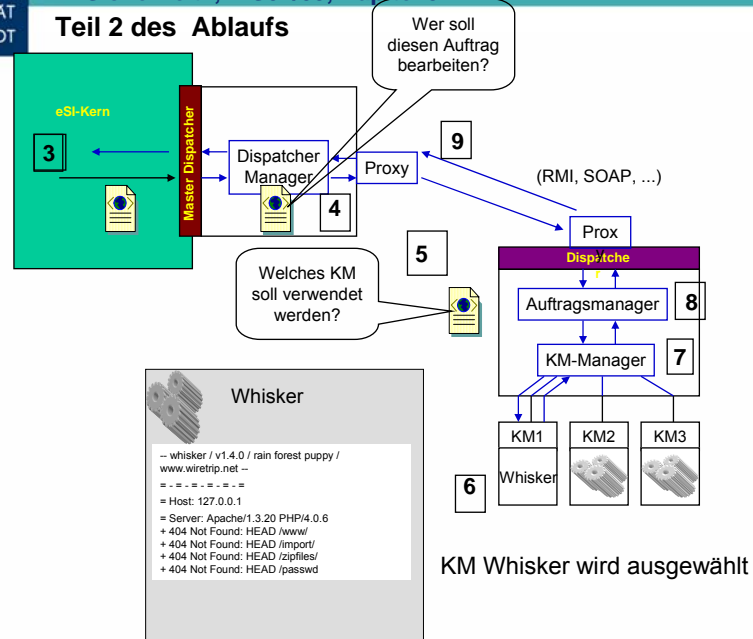


Ablauf eines Kontrollauftrags

Teil-1 des Ablaufs



Teil 2 des Ablaufs



TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 5** 85

Teil 2 des Ablaufs

Kontrollauftrag

- Nummer des KM: 1
- Gewünschte Informationen: State, httpResponse, Path
- Von KM benötigte Informationen: zsO_IP=127.0.0.1
- Erfasste Informationen: ++ 404 Not Found: HEAD /www/
- Aufgetretene Fehler: keine
- Dauer: 12:31-12:45

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT Sicherheit I, WS04/05, Kapitel 5** 86

Teil 3 des Ablaufs

Für Beispiel: CGI-Scripte:
Ja, wenn keine URL gefunden,
Nein, sonst, mit Angabe der URLs

Ausgabe des Prüfergebnisses

Maßnahmen überprüfen

Maßnahme umgesetzt?

Prof. C. Eckert FG Sicherheit in der Informationstechnik

Kapitel 6 Sicherheitsmodelle

Ziele

- **Abstraktion** von realen Gegebenheiten, Vereinfachung
- **Formalisierte Beschreibung**, Nachweis von Eigenschaften
- Rahmenfestlegung **für Implementierung**

6.1 Zugriffsmatrix-Modell (ZM) (Access Matrix Model)

Komponenten einer ZM

- (Dynamische) Menge von **Objekten** O_t
- (Dynamische) Menge von **Subjekten** S_t mit: $S_t \subseteq O_t$
- Menge von **Rechten** R
- **Zugriffsmatrix** $M_t : S_t \times O_t \rightarrow 2^R$, beschreibt einen Schutz-Zustand

Beispiel Schutzzustand: $M_t(s2, o2) = \{r1, r2\}$,

Subjekte	Objekte			
	o1	o2	o3	s1 ...
s1				
s2		{r1, r2}		
s3				

Beispiel

- Rechte-Vergabe an Prozesse: z.B. send $\in M_t(\text{pid}, 21)$

Modellierung von **Sicherheitseigenschaften** mit ZM:

- Spezifikation von **zulässigen** u. **unzulässigen** Zuständen
- Nachweise: u. a. nur zulässige Zustände **erreichbar**

Beispiel: Filtertabelle eines einfachen Firewalls

- Szenario: Paketfilter (Auszug) für ein universitäres Umfeld
- Rechtevergabe abgeleitet aus einer CERT Empfehlung
- *-Eintrag: beliebige Adresse bzw. beliebiger Port

Subjekt (Sendeadr., port)	Objekt (Zieladr., Zielport)				
	*, 53 (DNS)	intern, 123 NTP	*, 69 TFTP	*, 512 rlogin	*, *
(*, *)	Allow		Deny	Deny	
(extern, 123)		Allow			
PC-Cluster, *					Deny

Unentscheidbarkeit des Safety – Problems

- **Frage:** gibt es Algorithmus, um Erreichbarkeit von Zuständen zu entscheiden?

Präzisierung:

- gegeben Zustand z , in dem ein Subjekt s das Recht r an Objekt o **nicht** besitzt,
- gibt es Nachfolgezustand z' , in dem das Subjekt s das Recht r an Objekt o besitzt?

Zur Beantwortung der Frage muss das **dynamische Verhalten** des modellierten Systems beschrieben werden!

Systemverhalten beschrieben über

Folge von Konfigurationen $K_t = (M_t, O_t, S_t)$

$$K_{S_0} \stackrel{*}{\models} K_t \stackrel{\alpha_j(X_1, \dots, X_k)}{\models} K_{t+1} \stackrel{\alpha_j(X_1, \dots, k)}{\models} \dots \stackrel{\alpha_l(X_1, \dots, X_k)}{\models} K_{t+n}$$

- K_{S_0} beschreibt die Anfangskonfiguration;
also die Menge der zum Systemstart vergebenen Rechte
- $\alpha_j(X_1, \dots, X_k)$ beschreibt Ausführung einer Matrixänderungsoperation, **Beispiel dafür?**
- Nachfolgekonfiguration beschreibt neuen Schutzzustand

Erreichbarkeitsfrage: nunmehr präzisiert:

Safety-Problem: Gegeben $K_{S_0} \stackrel{*}{\models} K_t$ mit $r \hat{I} M_t(s, o)$

Problem: $\exists n$ mit $K_t \stackrel{n}{\models} K_{t+n} : r \hat{I} M_{t+n}(s, o)$?

Frage: Gibt es Algorithmus, der Safety-Problem **entscheidet**, für beliebiges Regelwerk, Subjekt s , Recht r und für ein Objekt o ?

Antwort: **Nein!** Safety-Problem der ZM ist **unentscheidbar**,
Papier von Harrison, Ruzzo, Ullman (1976)!

Bew.: zurückführen auf das **Halteproblem bei TM**:

- Modellierung so, dass TM anhält, genau dann wenn $r \hat{I} M_{t_c}(s, o)$
- Zustand der TM entspricht der Matrix M_t
- Zustandsübergang: Operation zur Matrixänderung

Fazit:

Es gibt **kein universelles** Entscheidungsverfahren!

Aber:

- (1) Es gibt Algorithmen für spezielle Klassen von Regeln, z.B. **Mono-operationale Systeme** sind entscheidbar, nur Regeln
 - if $r_1 \in M_t(s_1, o_1), \dots, r_n \in M_t(s_n, o_n)$ then **op**
 - anstatt then **op₁, ..., op_m**
 - Owner: **keine** speziellen Rechte zur Rechteweitergabe
- (2) Algorithmus existiert, falls **Menge der Subjekte beschränkt** ist!
 - Resultat hat großen Einfluss auf Entwicklung von Konzepten für sichere Systeme, eine Konsequenz:
 - **Einführung** von **Rollen-Konzepten** (siehe 6.2)
 - Ziel: Beschränkung der Subjekt-Menge

Fazit: (cont.)

- (3) Für gegebenes Regelwerk:
 - ist eine **individuelle Analyse durchführbar!**

Problem:

- **Aufwand** zur Durchführung der Analyse ist **sehr hoch!**
- Es sind kaum Tools zur Unterstützung vorhanden!
- Konsequenz: ein **a posteriori Nachweis ist schwierig!**
- Notwendig ist deshalb eine **konstruktive Lösung** mit dem Ziel, Sicherheitseigenschaften per konstruktionem zu gewährleisten
- Erreichbar u.a. durch die **Festlegen von Invarianten**, deren Einhaltung die Sicherheitseigenschaften garantieren

Beispiel hierfür siehe u.a. in 6.2

6.1.2 Nachweis von Eigenschaften im ZM

Frage: Welche Probleme könnten untersucht werden?

Antworten:

In den Folien 9 – 14 werden 3 mögliche Probleme beschrieben, die mit ZM-Modellen untersucht werden können

(1) **Soll-Ist**-Vergleiche:

- d.h. prüfen, ob Schutzziele durch Modellierung korrekt erfasst sind
- **wenn nicht**, dann kann auch die Modell-Implementierung **nur fehlerhaft** sein!

Beispiel: siehe nächste Folie

Beispiel:

- **Soll**-Anforderungen:
 - **Joe** darf **nicht das Recht** lese_Kontostand an Konto_Bill erhalten
- **Ist**-Modellierung:
 - auch Prozeduren sind Objekte bzw. Subjekte,
 - Prozedur: Drucke_Auszug besitzt eigene Rechte: u.a. das Rechte: lese_Kontostand

Beispiel: Auszug aus dem Schutzzustand:

	...	Konto_Bill	Drucke_Auszug
Joe			execute
Drucke_Auszug		lese_Kontostand	
...			

Konsequenz der Modellierung aus dem Beispiel:

- mit Joe's execute-Recht an Prozedur Drucke_Auszug und
- mit lese_Kontostand-Recht der Prozedur Drucke_Auszug gilt:
 - Joe erhält **implizit das Recht** lese_Kontostand an dem Objekt Konto_Bill, damit ein **Widerspruch zur Soll-Vorgabe**

Allgemein: **implizite Rechtevergaben** ergeben sich durch die Bildung der transitiven Hülle der Matrix

$$\exists r_1, r_2 \hat{I} R : \exists K_t : r_1 \hat{I} M_t(S, X_i) \wedge r_2 \hat{I} M_t(X_i, X_j),$$

Dann gilt auch: $r_2 \hat{I} M_t(S, X_j)$

D.h. durch Bilden der **transitiven Hülle** der Rechte kann man den Ist-Zustand (explizite und implizite Rechte) ermitteln

(2) **Konsistenz** von Schutzzuständen

D.h. Erkennen von **widersprüchlichen Rechtvergaben!**

Annahme:

Auch **Negative Rechte** können explizit vergeben werden:

$$\neg r \hat{I} M_t(S, X)$$

Z.B. negatives Recht: **Deny** unter Windows 2000/XP

Erkennen von inkonsistenten Schutzzuständen

- Gegeben seien z.B. folgende zwei Schutzzustände:
 1. $\exists r_1 \hat{I} R : \exists K_t : r_1 \hat{I} M_t(S, X_1) \wedge \neg r_2 \hat{I} M_t(S, X_2)$
 2. $r_1 \hat{I} M_t(S, X_1) \wedge r_2 \hat{I} M_t(X_1, X_2) \wedge \neg r_2 \hat{I} M_t(S, X_2)$
- S erhält implizit über X_1 r_2 an X_2 , aber gleichzeitig gilt: $\neg r_2 \hat{I} M_t(S, X_2)$, d.h **inkonsistente Rechtvergabe!**

(3) Modellierung von Zugriffsrechten

Entspricht Modellierung der gewünschten Rechtevergabe?

Beispiel:

Modellierung von Schreib-Rechten für Verzeichnisse

- **Semantik** (analog zu Unix) des Schreib-Rechts (w) sei:
 - mit **w-Recht für Directory D**
 - besitzt ein Subjekt **gleichzeitig** auch folgende 2 Rechte
 - (1) das Recht zum **Löschen** beliebiger Dateien $f \in D$ und
 - (2) das Recht zum **Hinzufügen** (Schreiben) von $f \notin D$ zu D
- Gewünschte Rechtevergabe sei wie folgt informell festgelegt:
 - Gegeben Directory D mit Datei f
 - Für D gelte: Alle Benutzer erhalten **w-Recht** an D ;

Beispiel: cont.

Gewünschte Rechtevergabe (cont.)

- Für f gelte:
- Nur Owner darf schreiben (modifizieren), also:
 - Rechtevergabe für D : **rwX -wx -w-** und
 - Rechtevergabe für f : **rw- r-- ---**

Durch die Semantik des w-Rechts für Verzeichnisse gilt:

- **w-Recht** an D :
 - erlaubt das Löschen von f in D für alle Benutzer!
 - erlaubt das Einfügen von $f \notin D$ anstelle von f in D !
- Fazit **Soll \neq Ist!** Die Modellierung ist zu grob!
- **wie könnte man das verbessern?**

Fazit: Zugriffsmatrix-Modell

Positiv: u.a.

- sehr einfach und intuitiv nutzbar,
- relativ flexibel, feingranulare Subjekte/Objekte und Rechte
- einfach zu implementieren, z.B. Rechtesten

Negativ: u.a.

- Fehlende Typisierungskonzepte (aber Gruppenbildung)
- keine Rechtevergabe an Klassen mit Rechte-Vererbung
- Skaliert sehr schlecht: in der Praxis häufig
 - hoch dynamische Menge von Subjekten
 - aufwändige Rechtevergaben, bzw. Rücknahmen
- wenig geeignet für größere Unternehmen, Web-Services ...

6.2 Rollen-basierte Modelle (R.S. Sandhu et. al. 1996)

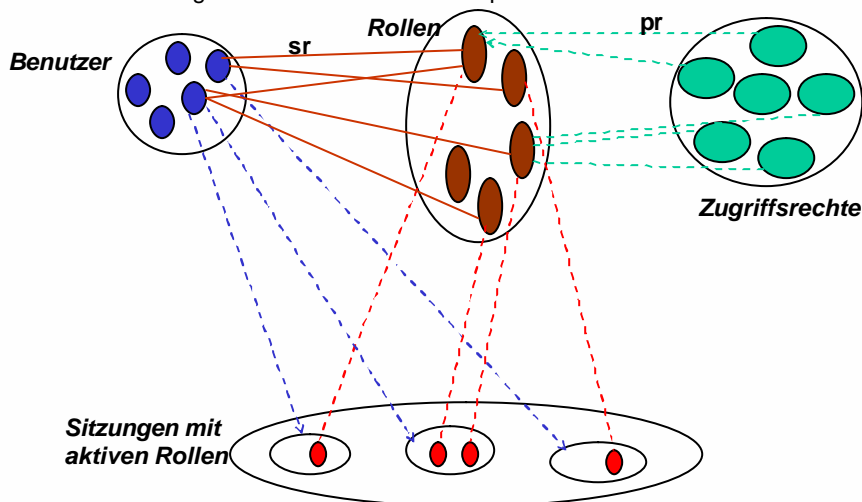
RBAC-Modell (Role-based Access Control)

- **Aufgaben-orientierte** Rechtevergabe durch Rollen
- **Rolle:** beschreibt bestimmte Aufgabe mit damit verbundenen
 - Verantwortlichkeiten und Berechtigungen
- **Nachbilden** von Organisationsstrukturen
 - gut geeignet für größere Unternehmen,
 - für dynamische Strukturen
- erfüllen der Prinzipien: **need-to-know**, **separation-of-duty**
- **weit verbreitet:** u.a.
 - Web-Services,
 - ERP (Enterprise Resource Planning)-Systeme (u.a.SAP),
 - CMS (Content-Management Systeme), ...

6.2.1 RBAC-Modell hier nur einfachste Ausprägung

- Menge von **Subjekten** = Benutzer;
- Menge von Rollen **Role**, Rolle $r \in \hat{I}$ Role
- Menge von **Zugriffsrechten** P (permission) für Objekte
- Zwei Abbildungen:
 - (1) Benutzer - Rollenzuordnung $sr : S \rightarrow 2^{Role}$
 - (2) Rechte - Rollenzuordnung $pr : Role \rightarrow 2^P$
- **Sitzung** $session \subseteq S \times 2^{Role}$, $(s, RL) \in session$, dann ist RL die Menge der aktiven Rolle des Benutzers s , $RL \subseteq sr(s)$
- $Ri \in session(s)$, falls $(s, RL) \in session \wedge Ri \in RL$
D.h. s agiert in Rolle Ri , falls s Mitglied in der Rolle Ri ist u. diese Rolle in einer Sitzung aktiviert hat

Zusammenhang zwischen den RBAC-Komponenten



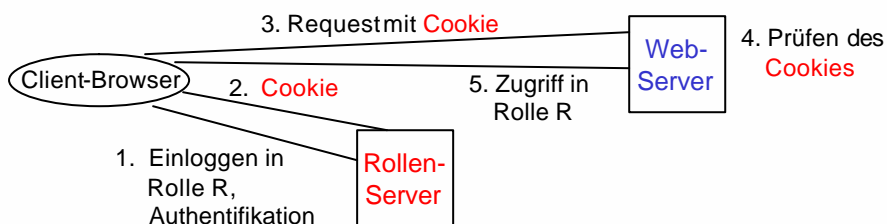
Beispiel: **Rollen** und deren Berechtigungen im Krankenhausszenario:

Behandelnde Ärzte (auch AiP, PJ, zeitweise zugeordnete Ärzte):

- ganze **Patientenakte** im Behandlungszusammenhang (außer besonders sensible Daten), (lesend, schreibend)
- **abteilungsinterne Daten** aller Aufenthalte
- **Pflegekräfte:**
 - **Zugriff auf Krankenakte**; Umfang durch Abteilungsleiter festgelegt
- **Sonstige Mitarbeiter** der Fachabteilung **analog** (z. B. Arztsekretariat)
- **Famulanten**, Studenten, Auszubildende:
 - erforderlicher Umfang durch verantwortlich Lehrenden festgelegt (im Rahmen seiner eigenen Befugnisse).
- **Verwaltungsmitarbeiter:**
 - **Stammdaten**, (lesend, schreibend)
 - **abrechnungsrelevante Daten** (u. U. auch besonders sensible!).

Beispiel: Web-Szenario:

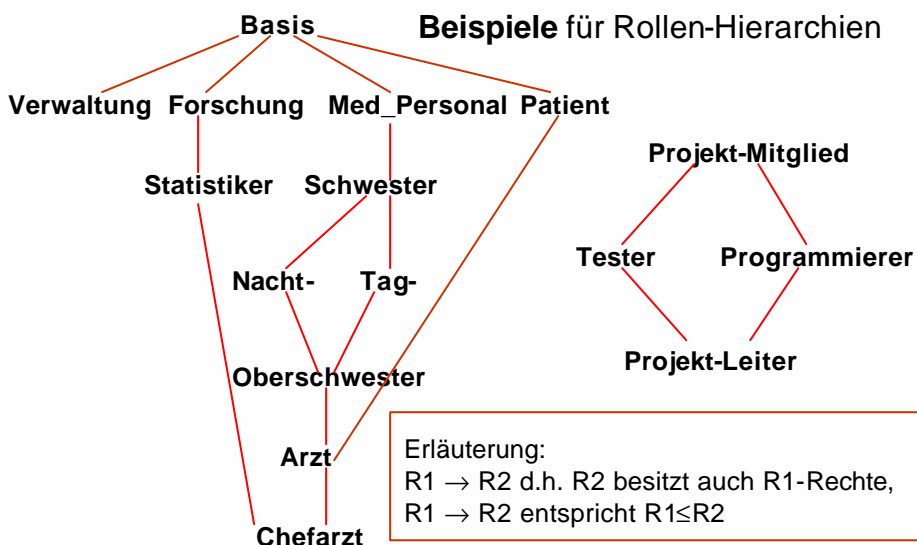
- **Web-Server** mit RBAC-Zugriffskontrolle
- z.B. spezieller **Rollen-Server**, bei dem sich Benutzer anmeldet, um eine Rolle zu aktivieren,
- Rollen-Server stellt darüber Bescheinigung aus (z.B. **Cookie**), das der Benutzer berechtigt ist, in der Rolle zu agieren
- Benutzer weist die Bescheinigung dem Web-Server vor und erhält die der Rolle zugeordneten Rechte



6.2.2 Rollenhierarchien

Ziel: Vereinfachung von Verwaltungsaufgaben,
Nachbilden hierarchischer Organisationsstrukturen

- Definition einer **partiellen Ordnung** \leq auf Rollen:
 $R_i, R_j \in Role$: falls $R_j \leq R_i$, dann besitzt R_i alle Rechte von R_j und ggf. noch zusätzliche Rechte
- **Beispiel:** Software-Entwickler \leq Projekt-Leiter
Rechte des Entwicklers: w,r,x, auf Projekt-Dateien
Rechte des Leiters: w,r,x, auf Projekt-Dateien **und**
w,r,x auf Projekt-Budget-Dateien, etc.
- **Vererbung** der Rollenmitgliedschaft: falls $R_j \leq R_i$, dann gilt: $\forall s \in S : R_i \in sr(s) \Rightarrow R_j \in sr(s)$
- Rechtevererbung ist aber nicht unproblematisch! **Wieso?**



Eigenschaften, die ein **RBAC-System garantiert** (Invarianten)

Bem.: die Einhaltung ist zu kontrollieren!

- Ein Subjekt darf nur in solchen Rollen **aktiv** sein, in denen es

Mitglied ist: $\forall s \in S : R_j \in session(s) \Rightarrow R_j \in sr(s)$

- Ein Subjekt besitzt nur die **Rechte seiner aktiven Rollen**:

Gegeben Funktion $exec : S \times P \rightarrow Boolean$, mit

$exec(s, p) = true \Leftrightarrow s$ ist zu p berechtigt

$\forall s \in S : exec(s, p) \Rightarrow \exists R_i \in Role :$

$R_i \in session(s) \wedge p \in pr(R_i)$

- In einem **hierarchischen** rollenbasierten Modell:

Gegeben sei die partielle Ordnung \leq über Rollen:

$\forall s \in S : exec(s, p) \Rightarrow \exists R_i \in Role, R_i \in session(s) \wedge$

$(p \in pr(R_i) \vee \exists R_j \wedge R_j \leq R_i \wedge p \in pr(R_j))$

6.2.3 Statische u. dynamische Beschränkungen

Ziel: Regeln, die die Rollenmitgliedschaft beschränken

(1) **Statische Aufgabentrennung** (separation of duty):

- Wechselseitiger Ausschluss von Rollenmitgliedschaften:

Festlegen einer Relation $SSD \subseteq Role \times Role$, mit

$(R_i, R_j) \in SSD \Leftrightarrow R_i$ und R_j sind **wechselseitig ausgeschlossen**

- Sei $Member(R_i) = \{s \mid s \in S \wedge R_i \in sr(s)\}$

Beschränkungsregel: $\forall R_i, R_j \in Role \forall s \in S :$

$(s \in Member(R_i) \wedge s \in Member(R_j)) \Rightarrow (R_i, R_j) \notin SSD$

Beispiel: $R1 = \text{Kassenprüfer_von_Filiale_A};$

$R2 = \text{Kassierer_in_Filiale_A};$

Beschränkung: $(R1, R2) \in SSD$

(2) Dynamische Aufgabentrennung:

Wechselseitiger Ausschluss von Rollenaktivitäten

- Festlegen einer Relation $DSD \subseteq Role \times Role$, mit
 $(R_i, R_j) \in DSD \Leftrightarrow R_i$ und R_j sind **dynamisch w.A.**
- Beschränkungsregel: Für jedes Subjekt s sei:
 $Active(s) = \{ R_i \mid \exists RL \subseteq Role \wedge$
 $(s, RL) \in session \wedge R_i \in RL \}$,
dann muss gelten;
 $(s \in Member(R_i) \wedge s \in Member(R_j))$
 $\wedge \{ R_i, R_j \} \subseteq Active(s) \Rightarrow (R_i, R_j) \notin DSD$

Beispiel: $R3$ = Kundenbetreuer; $R4$ = Konto_Besitzer;
Beschränkung: $(R3, R4) \in DSD$

Fazit: RBAC-Modell

- Rollenkonzepte sind **sehr flexibel** verwendbar, skalieren gut
- Rollen werden zunehmend eingesetzt, u.a. Web-Services
- Modellierung **zusätzlicher Zugriffsbeschränkungen** durch Relationen auf Rollen möglich (u.a. Kontext-Abhängigkeiten)
- **Direktes Nachbilden** bekannter Organisations- und Rechtstrukturen in Unternehmen
- intuitive und relativ einfache Abbildung der Rollen auf Geschäftsprozesse (Workflows)
- Änderungen **von pr selten**; dagegen aber u.U. Änderung der **Rollenmitgliedschaften sr häufig**;
- Konsequenz: **einfache** und **effiziente Rechte-Verwaltung**

Fazit: RBAC-Modell (cont.)

- Rechtevergabe erfolgt über pr für relativ statische Menge von Rollen,
- d.h. beschränkte Menge von Rollen: vgl. Safety Problem

Grenzen?

- **keine kontextabhängigen** Rechte:
Beispiele für Kontextw, die man ggf gerne modellieren möchte?
- **keine Kontrolle von Informationsflüssen**
z.B. kann Information durch Trojaner weitergereicht werden
- Umsetzung des RBAC Modells in heutigen Betriebssystemen:
Ideen? Probleme?

6.3 Bell LaPadula Modell (Bell, LaPadula 1973) (BLP)

Probleme bei 6.1 und 6.2:

- **keine Kontrolle von Informationsflüssen**

Lösung: Multi-level Security (MLS), **Labeling-Konzepte!**

Ausprägung: BLP-Modell: erstes formalisiertes Modell,

Modell (hier nur Auszug der wichtigsten Aspekte)

- **Zugriffsrechte**

$R = \{ \text{read - only, append, execute, read - write, control} \}$

- Menge von **Sicherheitsklassen** SC , $X \hat{I} SC$ mit $X = (A, B)$,
 A ist **Sicherheitsmarke** (labels), z.B. *vertraulich, geheim*
 B Menge von **Kategorien** (compartments), z.B. *Arzt*

- $\forall s \hat{I} S$: **Clearance**: $SC(s) \hat{I} SC$; Maximale $SC(s)_{MAX}$ und aktuelle $SC(s)_{AKT}$
- $\forall o \hat{I} O$: **Classification** $SC(o) \hat{I} SC$
- **Partielle Ordnung auf SC**: (SC, \leq) , für $X, Y \hat{I} SC$ gilt:
 $X = (A, B), Y = (A^c, B^c) \quad X \leq Y \hat{I} A \leq A^c \wedge B \hat{I} B^c$

Mandatorische (systembestimmte) Regeln

- MAC-Policy (mandatory access control)
- **Regeln**: no read up, no write down, tranquility

Simple-security-Property (no read up-Regel):

Für $z \hat{I} \{ \text{read} - \text{only}, \text{execute} \}$: $z \hat{I} M_t(s, o) \wedge SC(s) \geq SC(o)$

***-Property** (no write down-Regel):

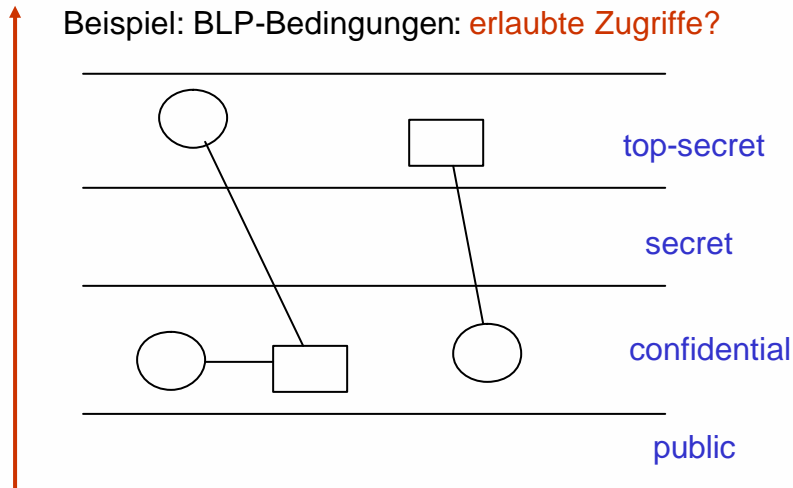
- append $\hat{I} M_t(s, o) \wedge SC(s) \leq SC(o)$; und
- read – write $\hat{I} M_t(s, o) \wedge SC(s) = SC(o)$

Ggf. **Strong Tranquility** Regel: während der Systemlaufzeit keine Änderung der Subjekt-Clearance oder der Objekt-Classification

Intuitive Interpretation der Regeln:

- Festlegen einer **partiellen Ordnung** über Sicherheitsmarken
- Informationsflüsse sind nur **von unten nach oben** (entlang der Ordnung) zulässig

Beispiel: BLP-Bedingungen: **erlaubte Zugriffe?**



Zulässige Informationsflüsse

○ Objekt

□ Subjekt

Zugriffsoperation	Beschränkung: Subjekt S / Objekt O
read (file)	$SC(S) \geq SC(O)$
exec (file)	$SC(S) \geq SC(O)$
write (file)	$SC(S) = SC(O)$
append (file)	$SC(S) = SC(O)$
read (directory) search (directory)	z.B. no read up $SC(S) \geq SC(O)$ $SC(S) \geq SC(O)$
create (directory)	$SC(S) = SC(O)$
read (signal/ipc)	$SC(S) \geq SC(O)$
kill (signal/ipc)	$SC(S) = SC(O)$

Beispiel:

BLP-Policy für Unix/MLS

Erweiterungen gegenüber Unix

- des **Login**:
Angabe von SC(s)
- der **Prozessbeschreibung**:
Prozesskontrollblock mit SC(s)
- der **Dateibeschreibung**:
i-node mit SC(file)

Grenzen des BLP-Modells

- Problem: Information/Objekte werden **sukzessive immer höher**, zu hoch (!), eingestuft **warum?**
- **Lösung:** Einführung von **Trusted Subjects** (Multi-level Subjekte) (u.a. Systemprozesse)
- Bem.: Subjekte sind **per definitionem vertrauenswürdig**, d.h. sie unterliegen nicht den BLP-Regeln, dürfen Sicherheitseinstufungen ändern (zurückstufen), **Problem?**
- Problem des **Blinden Schreibens**
 - ? s darf o modifizieren, aber anschließend (wegen no read up) nicht lesen!
 - ? Problematisch in Bezug auf Integrität!

Fazit: BLP-Modell

- formales, **einfach zu implementierendes** Modell
- Gut geeignet zum **Nachbilden hierarchischer Informationsflüsse**: z.B. Schutz personenbezogener Daten (u.a. medizinische Daten, Prüfungswesen)
- Viele **Betriebssysteme bieten BLP-Erweiterungen**: u.a. Sun Trusted Solaris 7, Trusted HP-Unix, Linux-Derivate
- **aber:** Beschränkte Ausdrucksfähigkeit
 - Keine Integrität (vgl. u.a. blindes Schreiben),
 - Keine Modellierung von Covert Channels über die Informationen dann doch unberechtigt fließen können
- Dennoch: **Labeling-Ansatz geeignet** zur Beschränkung von Informationsflüssen, in Varianten im Einsatz

Weitere Sicherheitsmodelle: u.a.

- **Biba-Modell**: Integritätslevel, Idee wie bei BLP
- **Verbandsmodell**: reines Informationsflussmodell
nur Sicherheitsmarken und Festlegung der Flussrelation,
d.h. der erlaubten/verbotenen Informationsflüsse
- **Non-Interference-Modelle**: **Idee**: Effekte von Aktionen
dürfen nur für Berechtigte sichtbar sein, Ausblenden der
Effekte ändert nichts an den Ergebnissen, die die Nicht-
Berechtigte berechnen
- **Chinese-Wall Modell**: u.a. Finanzbereich, **Idee**: Historie
der Zugriffe beschränkt die Zulässigkeit weiterer Zugriffe
- **Modelle für Authentifikation, kryptogr. Protokolle**, etc.

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 1

Kapitel 7 Grundlagen kryptographischer Techniken

Ziel: Einführung der wichtigsten Grundlagen zu Krypto-Verfahren zur Gewährleistung der: Vertraulichkeit, Integrität, Authentizität, Verbindlichkeit

7.1. Einführung

- **Kryptographie:**
Lehre von den Methoden zur Ver- und Entschlüsselung
- **Kryptoanalyse:**
Wissenschaft von Methoden zur Entschlüsselung
- **Kryptologie:** Kryptographie und -analyse, eng verzahnt
- **Steganographie:** (griech. stegano=geheim ; graphien=schreiben)
Methoden zum Verbergen der Existenz einer Nachricht, **Beisp.?**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 2

7.1.1 Klassen kryptographischer Verfahren

Definition: Kryptographisches Verfahren:
Ein **kryptographisches Verfahren** ist gegeben durch
 (M, C, EK, DK, E, D)

1. Menge von Klartextnachrichten M , über dem Alphabet A_1
2. Menge von Kryptonachrichten C , über dem Alphabet A_2
3. der Menge von Verschlüsselungs-Schlüsseln **EK**,
4. der Menge von Entschlüsselungs-Schlüsseln **DK**,
und der Abbildung: $f: EK \rightarrow DK$
mit: $d = f(e)$, $e \in EK$, $d \in DK$
5. dem **injektiven** Verschlüsselungsverfahren $E: A_1^* \times EK \rightarrow A_2^*$
6. dem Entschlüsselungsverfahren $D: A_2^* \times DK \rightarrow A_1^*$, mit
 $\forall M \in A_1^* D(E(M, e), d) = M$ mit $e \in EK$, $d \in DK$, $f(e) = d$

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 3

Einfaches Beispiel: Caesar-Chiffre

- $A_1 = A_2 = \{A, B, C, \dots, Z\}$
- **EK**: zyklische Verschiebung der Buchstaben um n Positionen **nach rechts**
- **DK**: zyklische Verschiebung der Buchstaben um n Positionen **nach links**
- $e=d=n$, d.h. n ist der Schlüssel

Beispiel:
Schlüssel: $n = 7$

a	b	c	d	...	x	y	z
H	I	J	K	...	E	F	G

- ▶ Klartext $M \in A_1^*$: DER BALL IST RUND
- ▶ Chiffretext $C \in A_1^*$: KLY IHSS PZA YBUK

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 4

Zwei Klassen kryptographischer Verfahren:
Symmetrische (Secret-Key) und **Asymmetrische** (Public Key)

Angreifer: Inform. $C \xrightarrow{M'}$ $M' \stackrel{?}{=} M$

Sender: Nachrichtenraum $M \xrightarrow{E}$ Schlüsselraum EK $e \rightarrow C = E(M, e)$

Empfänger: Schlüsselraum DK $d \rightarrow M = D(C, d)$

(a) **Symmetrisch** : $e = d$ **geheim!** z.B. DES, AES
 (b) **Asymmetrisch** : $e \neq d$ z.B. RSA, ElGamal

öffentlich \leftrightarrow geheim

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 5

Anforderungen an kryptographische Verfahren:

- Sicherheit darf **nicht von Geheimhaltung** der Ver- und Entschlüsselungsfunktionen abhängen!
Häufiger Verstoß dagegen: **Security by Obscurity**
z.B. beim Verschlüsselungsverfahren A5 (im GSM verwendet)
- geheimer Schlüssel darf mit der Kenntnis über die verwendeten Verfahren **nicht praktikabel** berechenbar sein!
- Stärke des Verfahren sollte **nur von der Güte** des geheimen Schlüssels abhängen! **Kerckhoffs-Prinzip**
- **Bem.:** Berechnungsaufwand zum Schlüsselknacken ist abhängig
 - von der aktuellen Rechner-Technologie (CPU),
 - von kooperativen Nutzungen (Internet, Grid-Computing, etc.)
 - neue Rechner-Architekturen, z.B. Quantencomputer?

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 6

- **Konsequenz:**
 - Verfahren muss gut konzipiert und Schlüsselraum EK muss **sehr groß** sein,
 - **Ausprobieren** aller Schlüssel (brute force) soll nicht mit praktikablem Aufwand möglich sein (exhaustive Search)
- **Beispiel:**
 - 56-Bit Schlüssel (u.a. DES): Schlüsselraum = 2^{56}
 - 1998 Deep-Crack-Supercomputer: Kosten ca 250.000 \$
 - Knacken eines DES-Schlüssels **in 56 Stunden!**
- **Anforderung:** (u.a. von RegTP)
 - symmetrische Verfahren: Schlüssel ≥ 128 Bit
 - asymmetrische Verfahren: Schlüssel ≥ 2048 Bit
- **Bekannte Verfahren** und Schlüssellängen?

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 7

7.1.2 Klassifikation von Angriffen auf Krypto-Verfahren

Basis:

- die Ver- und Entschlüsselungsverfahren **sein bekannt**,
- die Angriffe unterscheiden sich in dem **Umfang** der Informationen, die dem Angreifer vorliegen
- **Ziele:** bestimmten Text entschlüsseln, oder aber auch verwendeten geheimen Schlüssel knacken, etc.
- Verfahren sollten den Angriffen widerstehen!

(1) **Ciphertext-only-attack:** Informationen des Angreifers:

- **Kryptotext** liegt ihm zur Analyse vor,
- Kenntnis über **statistische Eigenschaften** des Alphabets u.a. Häufigkeit des Auftretens von Buchstaben
- **Ziel:** meist nur Kryptotext entschlüsseln

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 8

Frage: Abwehr von Angriffen (2) und (3), wie?

(2) **Known-plaintext-attack:** Informationen des Angreifers:

- **Klartext/Kryptotext** – Paare liegen vor (z.B. durch Sniffen) z.B. Standardbriefanfänge/-enden; Präambeln (header) bei Programmen, Challenge/Response-Paare
- Ziel: meist Knacken des verwendeten Schlüssels

(3) **Chosen-plaintext-attack:** Informationen des Angreifers:

- wählbarer **Klartext** und
- gezielte **Berechnung von Klartext/Kryptotext** – Paaren z.B. Passwort-Cracking-Attacke (wie funktioniert das?)

(4) **Chosen-ciphertext-attack:** Informationen des Angreifers:

- wählbare **Kryptotexte** und
- Berechnung von Klartext/Kryptotext – Paaren z.B. Anwendung bei asymmetrischen Verfahren

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 9

7.2 Symmetrische Verfahren

- Ver- und Entschlüsselungs-Schlüssel **sind gleich** $e = d$, oder leicht auseinander ableitbar, $d=f(e)$, f einfach berechenbar
- Nutzung eines gemeinsamen, **geheimen** Schlüssels (secret Key)
- **Problem:** Sicherer Austausch des gemeinsamen Schlüssels!
- Bekannte Repräsentanten symmetrischer Verfahren
 - **DES** (Data Encryption Standard): noch immer weit verbreitet
 - **AES** (aktueller Krypto-Standard), RC4, A5, IDEA, ...
- Klassen symmetrischer Verfahren: **Block-** und **Stromchiffren**

7.2.1 Blockchiffre

- zu verschlüsselnder Klartext wird in Blöcke **fester Länge** aufgeteilt (z.B. 64 oder 128 Bit) (siehe Folie 9)
- **blockweises** Verschlüsseln, für jeden Block **gleicher Schlüssel**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 10

Block-Chiffre: Klartextblöcke M_1, \dots, M_r , Blocklänge = n (häufig 64 Bit)

The diagram illustrates the block cipher process. At the top, a long bar represents the 'Klartext M' (plaintext M). This is split into blocks 'M1', 'M2', ..., 'Mr'. A note says 'Aufteilen in Blöcke z.B. n = 64'. Each block Mi is processed by a 'Verschlüsselung' (encryption) step using a 'Schlüssel K' (key K). The resulting ciphertext blocks are 'C1', 'C2', ..., 'Cr', labeled as 'Kryptotextblöcke'. Each ciphertext block Ci is then processed by an 'Entschlüsselung' (decryption) step using a 'Schlüssel K'' (key K'). The final output is the recovered plaintext blocks 'M1', 'M2', ..., 'Mr', which are concatenated back into the 'Klartext M'.

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 11

Beispiel für Blockchiffren: Triple-DES

- Basis: Blockchiffre DES (Data Encryption Standard),
 - ▶ Blocklänge: 64 Bit,
 - ▶ Schlüssellänge 56 Bit: zu kurz!
- Idee: Mehrfachanwendung des DES: Modi
 - ▶ DES-EEE₃ drei unterschiedliche Schlüssel für Verschlüsselung
 - ▶ DES-EDE₃ drei unterschiedliche Schlüssel; verschlüsselt, entschlüsselt und erneut verschlüsselt
 - ▶ DES-EEE₂ erste und dritte Operation verwendet den gleichen Schlüssel
- Effektive Schlüssellänge: $3 \cdot 56 \text{ Bit} = 168 \text{ Bit}$

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 12

- Anwendungsbereich für Blockchiffren:
 - Voraussetzung: vor Beginn der Verschlüsselung liegt der **Klartext vollständig** vor
 - **Beispiele:** Verschlüsselung von **Dateien** auf der Festplatte, Verschlüsseln von **E-Mails**, von Dateien **beim FTP**, ...
 - **ungeeignet für stromverarbeitende** Anwendungen, z.B. Sprachverschlüsselung, Tastatur-Eingaben, warum?

Betriebsmodi von Blockchiffren

ECB – Electronic Code Book:

- Blockweise verschlüsseln (siehe Folie 9)
- **Gleiche** Klartextblöcke ergeben **gleiche** Chiffretextblöcke
- **Probleme:** u.a. Einspielen von Blöcken; Vertauschen der Reihenfolge, Blockentnahme ist möglich

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 13

CBC – Cipher Block Chaining:

- Verknüpfen mit vorherigem Chiffretextblock
- Startwert = **Initialisierungsvektor IV**
- Gleiche Klartextblöcke ergeben **ungleiche** Chiffretextblöcke
- Einspielungen sind erkennbar: Fehler bei Entschlüsselung

Verschlüsselung

Entschlüsselung

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 14

OFB – Output Feedback und **CFB – Cipher Feedback-Modus**

- Nutzen der Blockchiffre als Stromchiffre (siehe Folie 13)

7.2.2 Stromchiffre

- Verschlüsselung mittels (pseudo)zufälligem **Schlüsselstrom K**
- Schlüssel hat die **gleiche Länge** wie Nachricht M
- **Verschlüsselung:** M **XOR** K
- **Problem:** Effiziente Erzeugung von Schlüsselströmen K
- **Lösung:** z.B. Nutzen von **Blockchiffren als Pseudozufallsgenerator**
 - d.h. Blockchiffre **zum Generieren** der Schlüsselbits verwendet
 - Einsatz von Blockchiffren im **OFB oder CFB-Modus** (Folie 13)

Idee: Sender u. Empfänger kennen **geheimen Initialwert**, erzeugen damit **gleiche Folge** von Schlüsselbits

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 15

Stromchiffre unter Nutzung einer Blockchiffre im OFB-Modus

Verschlüsselung

Entschlüsselung

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 16

7.3 Asymmetrische (Public-Key) Verfahren (Diffie, Hellman 1976)

- Ein Schlüsselpaar (K_E, K_D) pro Kommunikationspartner A
- Schlüsselpaar: ein **geheimer** und ein **öffentlicher** Schlüssel

Basis: **Einweg-Funktion** (engl. one-way) $f: X \rightarrow Y$

Eigenschaften von Einweg-Funktionen:

- (1) $\forall x \in X$ gilt: $f(x)$ ist **effizient** berechenbar; und
- (2) für fast alle $y \in Y$ gilt, dass es **nicht effizient** möglich ist, das Urbild zu y zu berechnen, also $x \in X$ zu berechnen, mit $y = f(x)$

- Bis heute nicht bewiesen, ob überhaupt Einwegfunktionen existieren
- Bewiesen ist: sie existieren gdw. **P \neq NP**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 17

- Gute Kandidaten:
 - (1) **Faktorisierung**: gegeben $n=pq$, berechne p,q
 - (2) **Diskreter Logarithmus**: gegeben p Primzahl, $g \leq p$ und y bestimme k so, dass $y = g^k \pmod p$

Einweg-Funktionen **mit Falltür** (engl. trapdoor one-way):

- mit Zusatzinformation sind Urbilder effizient berechenbar
- **Bsp.**: gegeben $n = pq$, und Funktion f mit $f(x) = x^2 \pmod n$,
 - Invertierung von f schwierig ohne Kenntnis von p,q ,
 - Kenntnis von p,q ist ‚**Falltür**‘, damit **Invertierung effizient** berechenbar

Beispiele für asymmetrische Verfahren:

- RSA („Quasi-Standard“), ElGamal-Verfahren

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 18

Allgemeine Eigenschaften asymmetrischer Verfahren:

- Die Schlüsselpaare (K_E, K_D) müssen folgende Eigenschaft erfüllen: $\forall M \in A_1^*: D(E(M, K_E), K_D) = M$,
 K_E sei der **öffentliche**, K_D der **geheime** Schlüssel
solche Schlüsselpaare müssen **leicht zu erzeugen** sein
- Ver- und Entschlüsselungen (E und D) sind **effizient durchführbar**
- K_D ist aus K_E **nicht** mit **vertretbarem Aufwand berechenbar** \Rightarrow Einsatz von Einweg-Funktionen mit Falltür
- Optionale Eigenschaft: ermöglicht Erzeugung **digitaler Signaturen**
 $\forall M \in A_1^*: E(D(M, K_D), K_E) = M$
 $D(E(M, K_E), K_D) = M$

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 19

Beispiel: RSA (1978), Rivest, Shamir, Adleman
Detaillierte Beschreibung siehe Buch

1. wähle Primzahlen p, q
Modul $n = pq$ (d.h. Basis: Primfaktorzerlegung)

2. wähle d so, dass $ggT((p-1)(q-1), d) = 1$,

3. wähle e mit $ed = 1 \pmod{((p-1)(q-1))}$ (typischer Wert: $e = 65537 (= 2^{16} + 1)$)

• (e, n) öffentlicher Schlüssel; (d, n) geheimer Schlüssel

• **Verschlüsselung** E: $E(M) = M^e \pmod{n = C}$

• **Entschlüsselung** D: $D(C) = C^d \pmod{n = M}$

• **Einsatzbereiche:** Verschlüsseln, Signieren, Schlüsselaustausch

Ronald Rivest Adi Shamir Leonard Adleman

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 20

Bem.: Public-Key Verfahren wesentlich ineffizienter (Faktor 1000) als symmetrische Verfahren (z.B. XOR, Shift)

In der Praxis (wo?) deshalb häufig in **hybriden Verfahren** eingesetzt



- **Public Key** Verfahren: Austausch des geheimen Schlüssels K ,
- **symmetrisches** Verfahren zur effizienten Daten-Verschlüsselung



Ablauf: z.B. vertrauliche Kommunikation zwischen A und B

- A: Erzeuge Schlüssel K_{AB} für symmetrisches Kryptoverfahren
 $E(K_{AB}, K_E^B) = \text{Crypt_Key}$, Verschl. mit Public Key von B
- B: $D(\text{Crypt_Key}, K_D^B) = K_{AB}$, Entschl. mit Private Key von B
- A: $E(M, K_{AB}) = \text{Crypt_Text}$, verschlüsselte Nachricht M
- B: $D(\text{Crypt_Text}, K_{AB}) = M$ entschlüsselte Nachricht M

Alles ok, oder noch Probleme?

Prof. C. Eckert FG Sicherheit in der Informationstechnik

 TECHNISCHE UNIVERSITÄT DARMSTADT	IT-Sicherheit, WS04/05, Kapitel 7	21
<h3>7.4 Schlüsselvereinbarung: Diffie/Hellman – Verfahren (DH)</h3> <p>Ziel: Vereinbarung eines gemeinsamen, geheimen Schlüssels, ohne diesen auszutauschen!</p> <p>Aber: keine Verschlüsselung, keine Authentifizierung der Partner!</p> <ul style="list-style-type: none"> ● Einsatz u.a. in SSL/TLS, Kerberos, IPsec-Protokollen ● basiert auf dem Problem des diskreten Logarithmus ● Bem: ElGamal-Verfahren basiert auf der gleichen Idee <p>Funktionsweise des DH-Verfahrens</p> <p>(1) Wähle große Primzahl q (allen Teilnehmern bekannt);</p> <p>(2) wähle allen bekannten Wert α, der primitive Wurzel von q in der zyklischen Gruppe Z_q^* der primen Reste modulo q ist. $\alpha \in Z_q^*$ und $\{1, \dots, q-1\} = \{\alpha^1, \dots, \alpha^{q-1}\}$</p>		
Prof. C. Eckert		FG Sicherheit in der Informationstechnik 

 TECHNISCHE UNIVERSITÄT DARMSTADT	IT-Sicherheit, WS04/05, Kapitel 7	22
Schritte	Teilnehmer A	Teilnehmer B
Wähle geheimen Schlüssel	$X_A \leq q$	$X_B \leq q$
Berechne öffentlichen Schlüssel und tausche ihn aus	$Y_A = \alpha^{X_A} \text{ mod } q$	$Y_B = \alpha^{X_B} \text{ mod } q$
Berechne gemeinsamen Schlüssel K_{AB}	$K_{AB} = Y_B^{X_A} \text{ mod } q$ $= \alpha^{X_A X_B} \text{ mod } q$	$K_{AB} = Y_A^{X_B} \text{ mod } q$ $= \alpha^{X_B X_A} \text{ mod } q$
Angreifer: Welche Kenntnisse?	Aufwand K_{AB} zu berechnen?	
Angriffe auf DH-Verfahren möglich? Probleme?		
Prof. C. Eckert		FG Sicherheit in der Informationstechnik 

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 23

7.5 Hashfunktion und Message Authentication Code

7.5.1 Hashfunktion:

Allgemein (Mathe, Informatik): Hashfunktion H ist eine Abbildung

- $H: X^* \rightarrow X^n$ (Message Digest, One-way Hash), z.B. $n = 128$
- H ist nicht **injektiv**, deshalb sind prinzipiell **Kollisionen** möglich!

Ziel: Einsatz von Hashfunktionen für IT-Sicherheit:

- **Integrität von Daten** mit H überprüfen!
- **Integritätsprüfung** für Dokument M : Berechnen von $H(M)$ und prüfen, ob $H(M)$ der Originalhashwert ist
- dazu notwendig: **eindeutige** Hashwerte von Daten berechnen,
- **Idee:** bei Veränderung der Daten muss sich auch ein anderer Hashwert ergeben, **Problem sind die möglichen Kollisionen (s.o.)!**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 24

Anforderungen an H , um Integritätsprüfungen zu ermöglichen:

1. $\forall M \in X^*$ gilt: $H(M) = h$ ist **einfach** zu berechnen.
2. Gegeben h : das Bestimmen des Wertes $M \in X^*$, mit $M = H^{-1}(h)$ ist **nicht effizient** möglich, Einwegeneigenschaft von H
3. **Kollisionsresistenz**: das Finden von $M, M' \in X^*$, mit $H(M) = H(M')$ ist **nicht effizient** möglich
4. Gegeben sei $M \in X^*$, es ist **nicht effizient** möglich, ein $M' \in X^*$ zu finden, so dass gilt: $M \neq M'$ und $H(M) = H(M')$
Unterschied zw. Anforderung (3) und (4), was ist einfacher?

Beispiele: (Basis der Funktionen: Daten-Kompression, siehe Fol.25)

- **DES-CBC**: 64-Bit Hashwert, der letzte Block dient als Hash
- dedizierte Hashfunktionen: u.a. MD4, **MD5** mit 128 bit Hash
- oder **SHA-1** (Secure Hash Algorithmus) 160 bit Hash

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 25

Kryptographische Hashverfahren basieren auf Kompressionsfunktion

M = M_1 M_2 ... $M_n + \text{Padding}$

Initialwert \rightarrow G G ... G Hashwert

- Initialwert für Kompressionsfunktion ist notwendig,
- Initialwert ist häufig immer gleich (wie bei MD5) oder wird als Klartext übertragen
- das ist der Ausgangspunkt für Angriffe, um gezielt Kollisionen zu konstruieren, Abhilfe durch HMAC (siehe Folie 27 ff)

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 26

7.5.2 Message Authentication Code (MAC)

Ziel: Authentizität des Datenursprungs und Datenintegrität

Hashfunktion mit Schlüssel: MAC: $X^* \times EK \rightarrow X^n$

- geheimer (Pre-shared) Schlüssel K_{AB} zwischen Partnern A,B

Prinzipieller Ablauf am Beispiel eines Nachrichtenaustausches

Sender A:

- (1) berechnet $MAC(M, K_{AB}) = h$ und
- (2) sendet M und h an B

Empfänger B:

- (3) empfängt M', h und
- (4) prüft $MAC(M', K_{AB}) = h'$, $h \stackrel{?}{=} h'$

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 27

Beispiel: **Keyed MD5**: $M' = M \parallel \text{key}$; berechnen von MD5(M')

Bem. MD5, SHA-1 etc.: Schlüssel **nicht** zum Verschlüsseln!

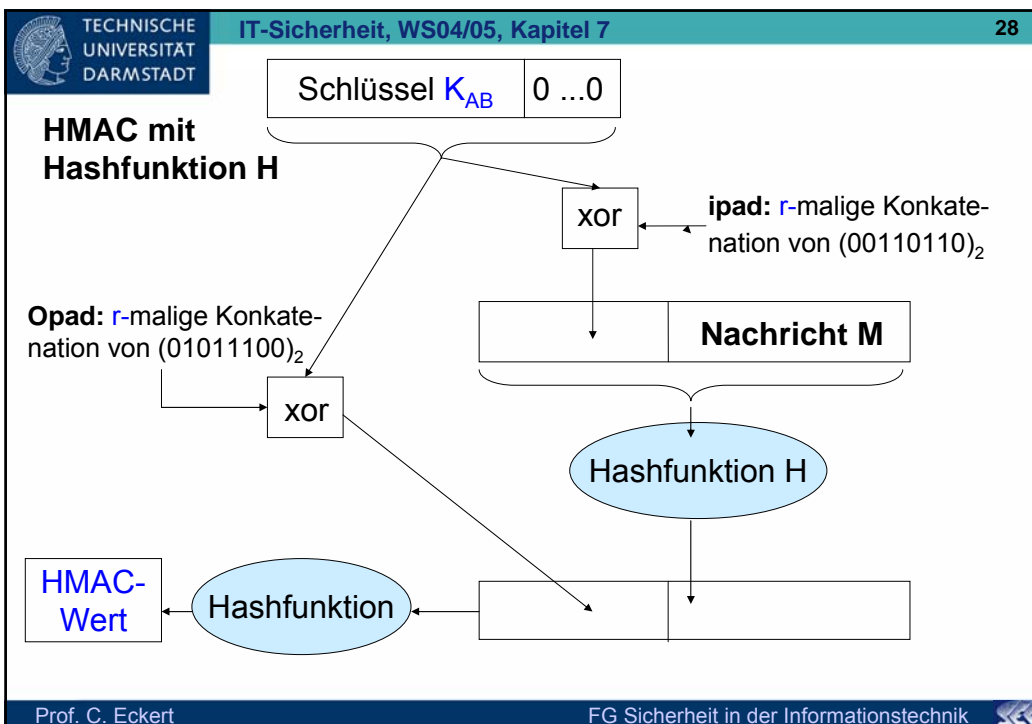
Problem bei u.a. MD5: Kollisionen effektiv erzeugbar,

- Angriffe seit 1996 bekannt
- **Lösung**: HMAC-MD5, bzw. allgemein **HMAC-Verfahren**

HMAC-Verfahren

- Basis: in vielen Protokollen/Anwendungen werden MD5 und/oder SHA-1 als Hash- bzw. MAC-Verfahren verwendet
- Völlig neues Verfahren wäre sehr aufwändig zu integrieren
- Idee: HMAC als ‚**Wrapper**‘ um existierende Verfahren,
- HMAC nutzt Schlüssel, **um Initialisierungsvektor zu variieren**
- HMAC-Verfahren sind **keine Verschlüsselungsverfahren**:
fallen nicht unter Krypto-Exportverbote,-Regulierungen

Prof. C. Eckert FG Sicherheit in der Informationstechnik



TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 29

Beispiel für Einsatz von Hashfunktionen:

- Speicherung **gehashter Passworte unter Unix**
- Hashing: Variante des DES-Verfahrens mit Salt-Wert
Salt: Wert 0, ..., 4095 2 Charakter, Effekte des Salts?
- Passwort dient als Hashschlüssel K
- Nachricht M ist die 64Bit Konstante (0 ... 0)
- $\text{crypt}((0, \dots, 0), \text{Passwort}, \text{Salt})$, Einweg-Funktion
Bem. Nicht mit sehr einfachen $\text{crypt}(1)$ -Chiffre zur Dateiverschlüsselung verwechseln!

Ablauf siehe nächste Folie

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 30

Hashen von Unix-Passworten

64 - Bit Eingabeblock , Schlüssel
0, ..., 0 Passwort

25 mal

DES

Auswahl einer Expansionsabbildung

Salt
fn

Kryptotext

Transformation

11 Bytes

Rfun4.4hYOU

Abgleich mit passwd

/etc/passwd oder shadow

Joe: fn Rfun4.4hYOU

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 31

7.6 Elektronische Signatur

Ziel: Nachweis der **Urheberschaft** eines Dokuments
ggf. juristische Gleichstellung zur handschriftlichen Unterschrift

Elektronische Signatur im Global and National Commerce Act:
The term "electronic signature" means an electronic sound, symbol, or process, attached to or logically associated with a contract or other record and executed or adopted by a person with the intent to sign the record.

Technische Erstellung elektronischer Signaturen:

- Public-Key-Verfahren, z.B. RSA,
- **Schlüsselpaar:** Verifikationsschlüssel (öffentlich),
Signaturschlüssel (privat)
- **Kryptographische Hashfunktion**, z.B. HMAC-SHA-1

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 32

Prinzip der Signaturbildung

Prinzip der Digitalen Signatur
Signaturbildung:

Dokument → Alice → Hash-Algorithmus → Fingerabdruck (Hashwert) → Signaturvorschrift (mit privater Schlüssel von Alice) → Signatur

Problem: u.a.

- **Sichere Speicherung** des Signierschlüssels: **wie, wo?**

Problem: u.a. **what you see is what you sign**

- Vertrauenswürdige Signierumgebungen?
- Unterschreiben als **willentlicher Akt?**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 33

Schaffung von Vertrauen:

- **Zertifizieren** (signieren) des öffentlichen Schlüssels

Prinzip der Digitalen Signatur
Schaffen von Vertrauen:

- Zertifizierung durch Zertifizierungsstelle, **Certification Authority (CA)**

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 34

Überprüfung der Signatur: erforderlich dafür sind:

- **öffentlicher Schlüssel** des Signierers
- **Zertifikat**, das die Gültigkeit dieses Schlüssels bestätigt

Prinzip der Digitalen Signatur
Signaturprüfung:

Problem: u.a.

- **Aussagekraft** des Zertifikats
- **Vertrauensmodelle:** Vertrauen in den Aussteller (CA)
- **Überprüfen** des Zertifikats!

Andere vertrauensbildende Maßnahmen?

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT-Sicherheit, WS04/05, Kapitel 7** 35

7.7 Zertifikate und PKI

Zertifikat: bindet öffentlichen Schlüssel an dessen Besitzer,

- digital signiert durch **vertrauenswürdige** Partei (CA)
- Zertifikate: idR nach **X.509.v3** Standard (RFC 2459)

SubjectName: C=DE, O=Fraunhofer, OU=SIT, OU=People, CN=Emil Mustermann

IssuerName: C=DE, O=Fraunhofer, CN=Zertifizierungsinstanz

SerialNumber: 1B

Validity - NotBefore: Wed Sep 18 09:12:25 2002 (020918071225Z)
NotAfter: Mon Jan 1 00:00:00 2007 (061231230000Z)

SubjectKey: Algorithm RSA (OID 1.2.840.113549.1.1.1), NULL
Public modulus (no. of bits = 1024):
0 CACAF080 64F76D97 EA2662FA 81FF10FF
Public exponent (no. of bits = 17):
0 010001

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT **IT-Sicherheit, WS04/05, Kapitel 7** 36

Beispiel: für ein Zertifikat und seine CA

Certificate Viewer: "Prof. Claudia Eckert's TU Darmstadt ID"

This certificate has been verified for the following uses:
SSL Client Certificate
SSL Server Certificate
Email Signer Certificate
Email Recipient Certificate

Issued To
Common Name (CN) Prof. Claudia Eckert
Organization (O) TU Darmstadt
Organizational Unit (OU) FB Informatik
Serial Number 04:EC

Issued By
Common Name (CN) RBG CA
Organization (O) TU Darmstadt
Organizational Unit (OU) FB Informatik

Validity
Issued On 20.10.2003
Expires On 19.10.2005

Fingerprints
SHA1 Fingerprint 5D:8E:AB:6A:53:FF:99:9F:47:3F:40:03:92:D5:1C:B7:BC:B2:B1:04
MD5 Fingerprint 6D:6A:9E:3D:91:3A:1E:D5:77:85:CF:B2:BF:ED:A8:EE

Certificate Viewer: "*.sec.informatik.tu-darmstadt.de"

This certificate has been verified for the following uses:
SSL Client Certificate
SSL Server Certificate

Issued To
Common Name (CN) *.sec.informatik.tu-darmstadt.de
Organization (O) TU Darmstadt, IT-Security group
Organizational Unit (OU) Webserver
Serial Number 05

Issued By
Common Name (CN) sec.informatik.tu-darmstadt.de
Organization (O) TU Darmstadt, IT-Security group
Organizational Unit (OU) Self-CA

Validity
Issued On 10.08.2003
Expires On 16.01.2006

Fingerprints
SHA1 Fingerprint F7:58:72:35:2A:D9:04:B0:01:89:F7:6D:88:C9:34:5B:64:A9:DD:82
MD5 Fingerprint 70:FE:90:62:90:18:84:B7:74:7F:04:78:75:AB:E2:F1

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 37

Public Key Infrastruktur (PKI)

“Public Key Infrastructure (PKI) provides the means to *bind public keys to their owners* and helps in the distribution of reliable public keys in large heterogeneous networks.“ NIST

The set of hardware, software, people, policies and procedures needed to *create, manage, store, distribute, and revoke Public Key Certificates* based on public-key cryptography. IETF PKIX W. group

Komponenten einer PKI (siehe auch Folie 38)

- **Certification Authority (CA):**
 - Stellt Zertifikate aus und signiert sie
 - Veröffentlicht aktuelle Zertifikate
 - Erstellt und veröffentlicht Listen von ungültigen Zertifikaten (CRLs), Certificate Revocation List
 - bietet Online Certificate Status Protocol (**OCSP**) für Clients

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 38

- **Registration Authority (RA):**
 - bürgt für die Verbindung zw. öffentlichem Schlüssel und Identitäten/Attributen der Zertifikatsinhaber
- **Verzeichnisdienst:** idR LDAP, Verteilung der Zertifikate und CRLs
- **Zeitstempeldienst:** signierte Zeitstempel (Gültigkeitsdauern, ...)
- **Personalisierung (PS)**
 - Übertragung von Schlüssel (secret key) und Zertifikaten

The diagram illustrates the components of a PKI system. At the top, a 'Trust Center' contains a 'Certification Authority (CA)'. Below the CA are three main services: a 'Registration Authority (RA)' (red box), a 'Verzeichnisdienst (DIR)' (green cylinder), and a 'Zeitstempeldienst' (green cylinder) with a clock icon. The RA, DIR, and Zeitstempeldienst are all connected to the CA. Below these services, an 'Endanwender' (stick figure) is connected to a 'PKI-Anwendung' (box). The PKI-Anwendung is also connected to a 'Personal Security Environment (PSE)' (box with a key icon). The Endanwender and PKI-Anwendung are connected to each other.

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 39

Ablauf für eine Zertifikat -Ausstellung

The diagram illustrates the process of certificate issuance. It involves three main entities: Benutzer A (User A), RA (Registration Authority), and CA (Certificate Authority). Benutzer A provides Credentials and a Passcode to the RA. The RA then sends the Passcode and Credentials to the CA. The CA issues a Zertifikat (Certificate) containing the key of Benutzer A, signed by the CA. The CA also stores the certificate in an Archiv (Archive).

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 40

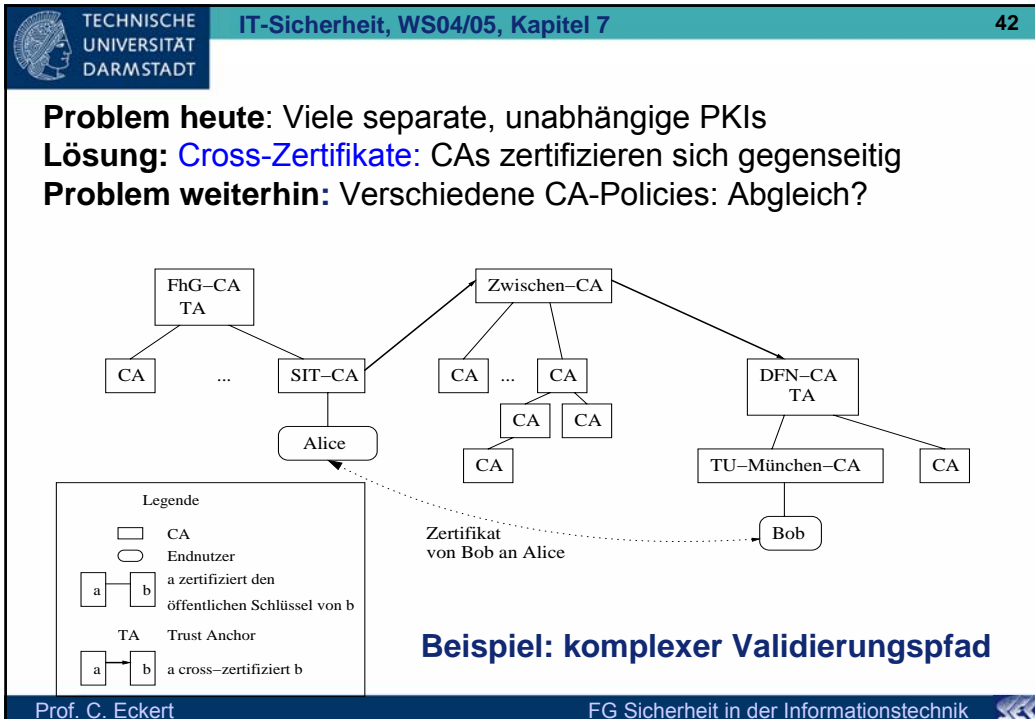
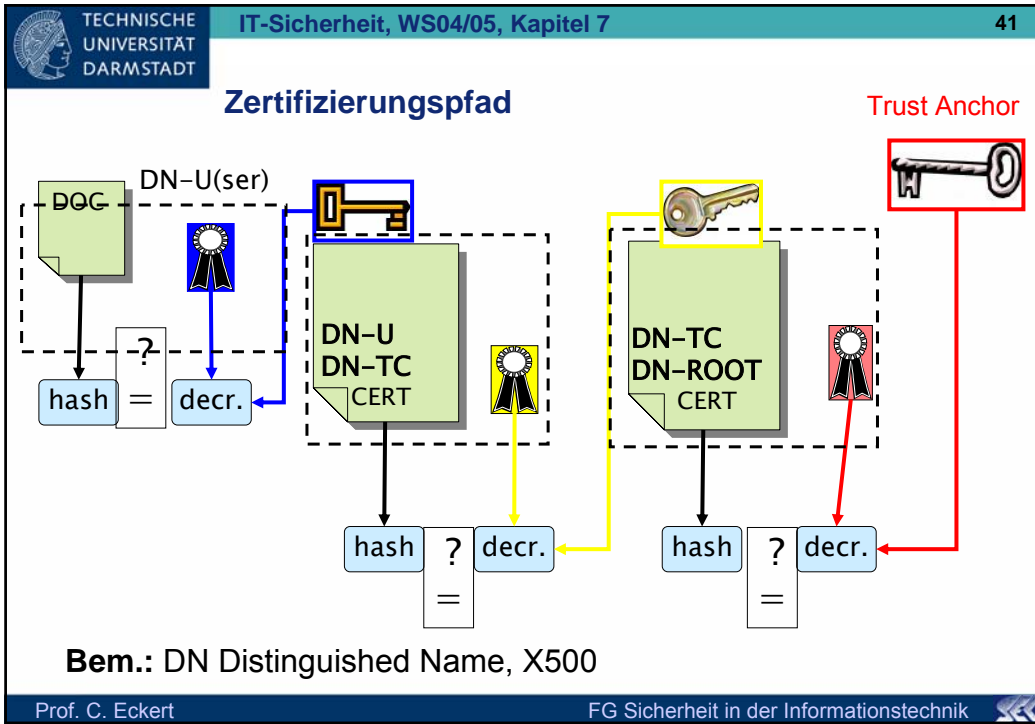
Häufig: Hierarchie von CAs

- Higher level CAs stellen Zertifikate für untergeordnete CAs aus
- jeder vertraut der top-level CA, der **Root-CA** (z.B. RegTP)
- Validierung eines Zertifikats: Aufbau eines **Zertifizierungspfades**

The diagram shows a hierarchy of CAs. At the top is the Zuständige Behörde (Responsible Authority), which is the RegTP. Below it are two Zertifizierungsstellen (Certification Authorities). Each Zertifizierungsstelle has three Teilnehmer (Participants) below it.

Hierarchie von CAs

Prof. C. Eckert FG Sicherheit in der Informationstechnik



TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 43

7.8 Deutsches Signaturgesetz (SigG) vom 22.5. 2001

- Gesetz über die Rahmenbedingungen für elektronische Signaturen
- **drei Signatur-Arten**: einfache, fortgeschrittene, qualifizierte

Signatur, Zertifikat, Zertifizierungsstelle (laut Gesetz)

- **Eine elektronische Signatur** ist ein:
„mit einem **privaten Signaturschlüssel** erzeugtes Siegel, das digitalen Daten assoziiert ist und das über den zugehörigen öffentlichen Schlüssel, der mit einem **Zertifikat** einer **Zertifizierungsstelle** versehen ist, den Inhaber des Signaturschlüssels und die **Unverfälschtheit der Daten** erkennen lässt.“
- **Zertifikat**: „eine mit einer **qualifizierten elektronischen** Signatur versehene, digitale Bescheinigung über die Zuordnung eines öffentlichen Signaturschlüssels zu einer natürlichen Person. ...“

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 44

- **Zertifizierungsstelle**: „eine natürliche oder juristische Person, die die Zuordnung von öffentlichen Signaturschlüsseln zu natürlichen Personen bescheinigt.“
 - der Betrieb einer Zertifizierungsstelle ist genehmigungsfrei
 - der Dienst ist der RegTP anzuzeigen (dabei Angabe eines Sicherheitskonzepts)
 - ein **Akkreditierter** Dienst erhält Gütezeichen der **RegTP**, d.h. Bestätigung: Anbieter hat Nachweis erbracht, dass seine technischen und administrativen Sicherheitsmaßnahmen den gesetzlichen Anforderungen entsprechen.
 - ein akkreditierter Anbieter ist berechtigt, qualifizierte elektronische Signaturen mit Anbieter-Akkreditierung zu erstellen.

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 45

Unterschiedliche Qualität von elektronischen Signaturen
Laut Deutschem Signaturgesetz:

- die **einfache** elektronische Signatur (unreguliert):
 - Keine Anforderungen an Zertifikate, Schlüsselerzeugung
 - ist **nicht** der Schriftform rechtlich gleichgestellt,
 - **Konsequenz** u.a. potentiell Geschädigter muss Schaden selber nachweisen, z.B. falsche Zertifikatangaben
 - Bei fortgeschrittener Signatur dagegen:
Haftung des Signaturanbieters für Richtigkeit und Vollständigkeit der Zertifikatangaben
- **Fortgeschrittene** elektronische Signaturen:
 - Ersetzt rechtlich **nicht** die eigenhändige Unterschrift
 - müssen Anforderungen gemäß § 2 II SigG erfüllen, d.h.

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 46

- **Anforderung** an Zertifikatsaussteller (CA):
 - muss **Genehmigung** gemäß §4 besitzen, dazu
 - Nachweis der Zuverlässigkeit (u.a. Einhaltung von Rechtsvorschriften),
 - Nachweis der Fachkunde,
 - Vorlage eines Sicherheitskonzepts für die CA
 - geprüfte Umsetzung des Sicherheitskonzepts
 - muss die Richtigkeit öffentlicher Schlüssel beglaubigen
- **Einsatz** fortgeschrittener Signaturen: u.a.
 - E-Business zwischen Geschäftspartnern mit vorhandenen Rahmenabkommen für die Zusammenarbeit

Qualifizierte elektronische Signatur:

- **rechtliche Gleichstellung eigenhändiger** Unterschrift
- unterliegt SigG, hohe Anforderungen, z.B. Schlüsselgenerierung

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 47

Fazit zur Signaturen, PKI

- wiss. Fragestellungen sind **im Prinzip schon lange gelöst**
- dem PKI-Hype vor ca 4 Jahren folgte die Ernüchterung:
 - das Etablieren einer Unternehmens-PKI ist **aufwändig**
 - das **Ausrollen** von Zertifikaten und Signaturkarten ist aufwändig, **Nutzen war gering**
 - **Sperrlistenverwaltung**, Trust-Center-Aufbau etc. schwierig
 - Unternehmensübergreifende PKI-Strukturen **gibt es kaum**

Einige Standards zur Thematik:

X.509: Standard für digitale Zertifikate
PKCS: De-Facto Standard, definiert von RSA, siehe Folie 48
PKIX: PKI-Gesamtstandard für das Internet
ISIS-MTT: deutscher PKI-Gesamtstandard, angelehnt an PKIX

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 48

Public Key Cryptography Standards (PKCS)

- PKCS #1: RSA Encryption Standard
- PKCS #3: Diffie-Hellman Key-Agreement Standard
- PKCS #5: Password-Based Encryption Standard
- PKCS #6: Extended-Certificate Syntax Standard
- PKCS #7: Cryptographic Message Syntax Standard
- PKCS #8: Private-Key Information Syntax Standard
- PKCS #9: Selected Attribute Types
- PKCS #10: Certification Request Syntax Standard
- PKCS #11: Cryptographic Token Interface Standard
- PKCS #12: Personal Information Exchange Syntax Standard
- PKCS #13: Elliptic Curve Cryptography Standard

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 49

Fazit zur Signaturen, PKI (cont.)

- **Zukunft:** neue Anwendungsfelder für Signaturen und PKI: u.a.
 - **Identitätsmanagement** und Provisioning
 - Eindämmen von SPAM mittels signierter Mails?
 - Einführung der **Gesundheitskarte** (2006)? Infrastruktur für
 - 80 Millionen Versicherte, 120.000 niedergel. Ärzte,
 - 55.000 Zahnärzte, 2.200 Krankenhäuser, 21.000 Apotheken
 - >280 gesetzliche Krankenkassen, 50 Private
 - **mobiles Arbeiten?**
 - Vertrauen schaffen zwischen mobilen Geräten, ...
 - **mobile Signatur** auf dem Handy: Hot Topic bei MNOs (Mobile Network Operators)

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT-Sicherheit, WS04/05, Kapitel 7 50

Fazit: Einsatzbereich kryptographischer Technologien

- **Vertraulichkeit:** Schlüsselaustausch (DH, RSA, ..)
Verschlüsselung (RSA, AES, DES, ...)
- **Verbindlichkeit:** Digitale Signatur, PKI, Zertifikate
(RSA, DSS (Digital Signature Standard), PKCS ...)
- **Integrität:** Hashfunktionen, MACs
(MD5, SHA-1 (Secure Hash Algorithm), DES-CBC, ...)
- **Authentizität:** Message Authentication Code (MAC), Signaturen
(HMAC-MD5, HMAC-SHA-1, ...)

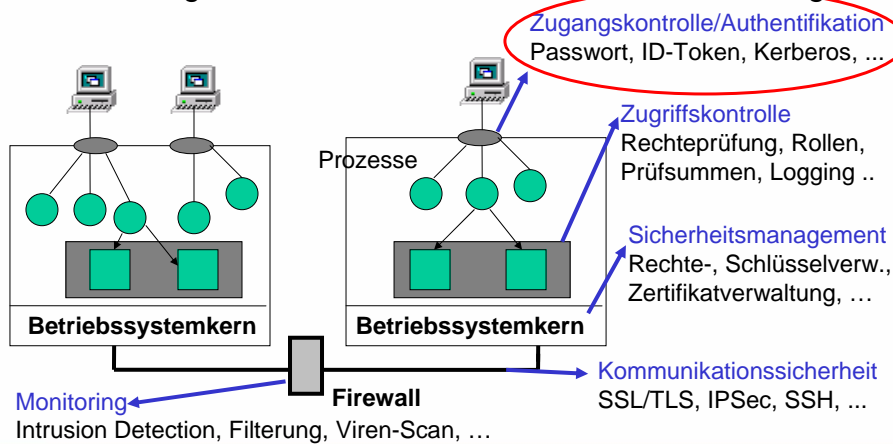
Basistechnologien werden **in Protokollen/Diensten kombiniert** z.B.

- Authentifizierte, vertrauliche und integere Kommunikation
mit SSL: Zertifikate, RSA (oder DH), MD5, SHA-1, Triple-DES

Prof. C. Eckert FG Sicherheit in der Informationstechnik

Sicherheitsdienste:

- Mechanismen, Dienste, Protokolle zur Umsetzung der Sicherheitspolicy und des Modells
- Verwaltung sicherheitsrelevanter Informationen, Management



Kapitel 8 Authentifikation

Ziel:

- eindeutige Identifikation und Nachweis der Identität
- Abwehr von Spoofing-Angriffen

Problem:

- Nicht nur Mensch-zu-Gerät Interaktion, sondern auch
- auch Gerät-zu-Gerät bzw. Gerät/Dienst-zu-Gerät/Dienst

Bem: zunehmende Vernetzung u. Miniaturisierung:

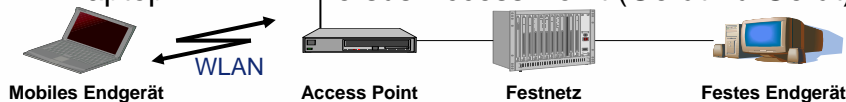
Geräte-zu-Geräte Kommunikation steigt rapide an!

Notwendig: Konzepte und Verfahren, um sowohl

- menschliche Individuen eindeutig zu identifizieren
- als auch Geräte (Web-Server, Laptop, Handy, ...) und
- Dienste (Dateisystem, Amazon, Bankportal,)

Formen der Authentifikation:

- **Einseitig:** u.a.
 - Benutzer versus PC (Mensch-zu-Gerät),
 - Handy vs Netz-Provider (GSM/GPRS) (Gerät-zu-Gerät)
 - WWW-Server (bei SSL) vs. Nutzer (Dienst-zu-Mensch)
 - Laptop im WLAN versus Access Point (Gerät-zu-Gerät)



- weitere Beispiele? **Mögliche Probleme?**

- **Wechselseitig:** u.a.
 - Handy versus Netz-Provider bei UMTS
 - Online Banking, falls Bank z.B. X.509 Zertifikate verwendet
 - weitere Beispiele? **Mögliche Probleme?**

Klassen von Authentifikations-Diensten

Authentifikation durch

- **Wissen:** z.B. Passworte, PINs, kryptogr. Schlüssel
- **Besitz:** z.B. Smartcard, USB-Token, SIM-Karte (Handy)
- **biometrische Merkmale:** z.B. Fingerabdruck
- **Mehrfaktor-Authentifikation:**

Kombination von Konzepten:
z.B. Handy, AOL-Zugang (USA), ...



Beispiel: 2-Faktor- Authentifikation beim Handy:

- (1) Authentifikation über PIN (**Wissen**) gegenüber SIM-Karte
 - (2) **und Besitz** der SIM-Karte (enthält geheimen Schlüssel K_{SIM})
- SIM-Karte authentifiziert sich gegenüber dem Netz mit K_{SIM}

8.1 Authentifikation durch Wissen

Allgemeine Vorgehensweise: Challenge-Response Verf.

- Authentifikation eines **Subjekts** gegenüber einer **Instanz**
 - **Subjekt** (Mensch, Gerät, Dienst, ...)
 - **Instanz** (Server, Gerät, Dienst, Mensch)

Idee: Authentizitätsnachweis (z.B. bei jedem Login)

- Subjekt gibt seine **Identität** an: z.B. Name, IMSI, MAC-Adr.
- Instanz sendet eine **Challenge** (idR Zufallszahl) zum Subjekt
- Subjekt **berechnet Response** (z.B. mittels Verschlüsselung)
- Instanz **prüft Response**, falls korrekt, dann hat Subjekt ein geheimes Wissen (z.B. Schlüssel) nachgewiesen

Passwort-Verfahren ist ein (schlechter) Spezialfall, **warum?**

8.1.1 Passwort-Verfahren

Vorab etablierte Basis:

- **Pre-Shared Secret:** vereinbartes Passwort zwischen System und Benutzer
- System **speichert/verwaltet Hashwerte** der Pre-Shared Secrets

Ablauf der Authentifikation:

- Angabe der **Identität:** Benutzer-Namen/Kennung
- **Challenge:** „Geben Sie Ihr Passwort ein“
- **Response:** Eingabe des Passwortes
- **Prüfung:**
 - System berechnet kryptogr. Hashwert über Passwort und
 - vergleicht Hashwert mit gespeichertem Hashwert

Probleme: u.a.(1) Verwaltung **gehashter Passworte** (der Pre-shared secrets)

- Problem 1.1: sichere Speicherung in Passwort-DB
mögliche Angriffe? Warum eigentlich Hashwerte?
- Problem 1.2: kontrollierter Zugriff auf DB notwendig
mögliche Angriffe?
- Problem 1.3: sicherer Transfer der Daten zum System
in welchen Kontexten ein Problem? Mögliche Angriffe?
- Problem 1.4: Authentizität des Systems u. der DB
mögliche Angriffe?

Welche Lösungen für Problembereiche 1.1-1.4 **bereits bekannt?**

Probleme (cont.): u.a.(2) Challenge ist **zu einfach, immer gleich**: **Konsequenz:**

- immer das gleiche Passwort als Response vorzuweisen
- Angreifer muss also **nur das Passwort** in Erfahrung bringen
Studien: ~ 22% der Passworte sind leicht zu knacken!

Angriffe auf Passworte:

- **Passwort-Cracking:** mittels frei verfügbaren **Crack-Tools**:
u.a. für Unix: John the Ripper (JtR), www.openwall.com/john
- **Social Engineering-Angriffe:** z.B. Telefonanrufe vom ‚Admin‘:
„Ich benötige Ihr Passwort für Admin-Aufgaben“
- **unverschlüsselte Passwort-Übertragung:** u.a.
im LAN vom Client zum Server, in Komm-Protokollen (**Bsp.?**)

Fazit: **Password-Replay** und **Spoofing** ist idR einfach

Lessons Learned:

- Notwendig ist eine **pragmatische, effiziente Methode**, so dass
- ein Subjekt bei **jeder Authentifizierung** die Kenntnis eines
 - **neuen, noch nicht verwendeten** Passworts nachweisen muss!

Lösungen: u.a.

- **Einmal Passwort** –Verfahren, OTP (One-time Passwort)
z.B. RSA SecureID



- **Ticket-basiert** mit Nachweis, dass man der Eigentümer des Tickets ist (z.B. Kerberos)

Allen diesen Lösungen ist gemeinsam:

- sie basieren auf einem Challenge-Response Verfahren

8.1.2 Challenge-Response-Verfahren (CR)

Allgemeine Methode, einsetzbar zur Authentifikation

- von Geräten, Services, Menschen
- verwenden von **Einmal-Passwörtern**
- zwei verschiedene Formen von CR-Verfahren:
Symmetrisches und **asymmetrisches** CR-Verfahren
- Symmetrisches Challenge Response: u.a. bei GSM/GPRS/UMTS, 802.11 a/b/g (WLAN), Kerberos, ...
- Asymmetrisches Challenge Response: u.a. bei SSL, Bem: **asymmetrisches CR in der Übung**, Vorteil, Nachteil?

Symmetrisches CR-Verfahren:

Basis: vorab **geheimer Schlüssel** K_{ID} (*pre-shared Secret*)
zwischen Subjekt und Instanz vereinbart

Protokollschritte des **symmetrischen CR**:

Ziel: Subjekt A authentifiziert sich gegenüber Instanz B

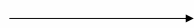
Subjekt A:

Instanz B

Schlüssel K_{ID} , Identifikation ID

Z.B. Login:

(1) ID



Schlüssel K_{ID} zu ID

Erzeugen von **RAND**
das ist die Challenge

(2) **RAND**



$E(\text{RAND}, K_{ID}) = C$

(3) **C**



$E(\text{RAND}, K_{ID}) = C'$

C (= Response)

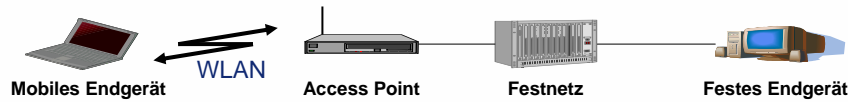
Test: $C' = C?$

Frage: kann man als Funktion E auch ein MAC-Verfahren verwenden?

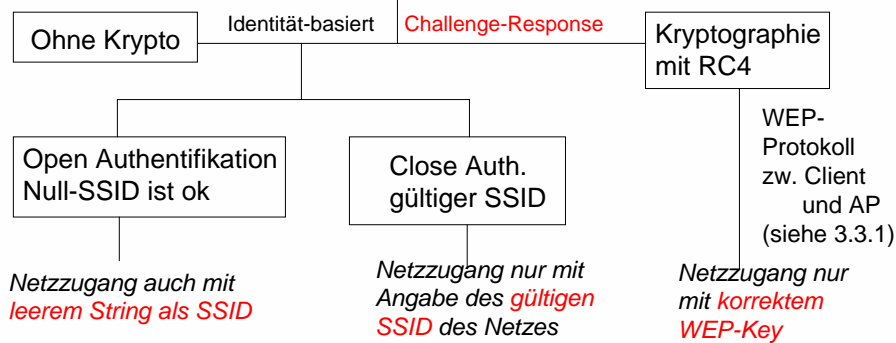
Bei Verwendung des CR-Verfahrens ist zu beachten:

- Falls **Klartext-Raum** für Challenges **zu klein** ist:
 - Angreifer legt Liste an, wartet auf wiederholte Challenge
 - ist das ein **Design- oder Implementierungsproblem?**
 - wie kann man das Problem lösen?
- **Man-in-the-Middle** Angriffe sind möglich (Designfehler?)
 - wie sieht ein möglicher Ablauf eines MitM-Angriffs aus?
 - was hat ein Angreifer von einem solchen Angriff?
 - wie könnte man das **abwehren (anderes Design?)**?
- **Known-Plaintext** Attacken:
 - Klartext **RAND**, Response **C** ist zugehöriger Kryptotext
 - durch Abhören: viele (**RAND,C**)-Paare abfangen,
daraus den benutzten Schlüssel K_{ID} berechnen, **Abwehr?**

8.1.3 Beispiel: Authentifikation in 802.11-Netzen (WLAN)



802.11 Authentifikationsvarianten



Beispiel WLAN (cont.)

Zugang zum WLAN: 3 Schritte:

- Identifikation, Authentifikation, Aufnahme in das WLAN
- (1) Identifikation des Access Points (AP)**
- Jeder AP besitzt eine **SSID** (Service Set Identifier),
- **AP sendet** in festen Intervallen Beacon-Frames mit SSID
- **Bem:** Broadcast der SSID kann durch **cloaked mode** unterbunden werden, aber das ist **nicht standard-konform**
 - **Kein nennenswerter** Schutz: SSID wird als Klartext in Management-Nachrichten des AP versandt!

Alternative:

- Client **erfragt** SSIDs mittels Probe-Nachrichten
- Merke:** SSID (oder ESSID) werden im Klartext übertragen

Beispiel WLAN (cont.)

Identifikation des mobilen Clients

- Vergabe von Netznamen auch an Clients (SSID, ESSID)
- nur Clients mit dem SSID des AP werden akzeptiert, aber
- falls in AP **any eingetragen** ist, wird **jeder** akzeptiert

Alternativ:

- Zugangskontrolle über MAC-Adresse, Qualität?

(2) Authentifikation des Clients?

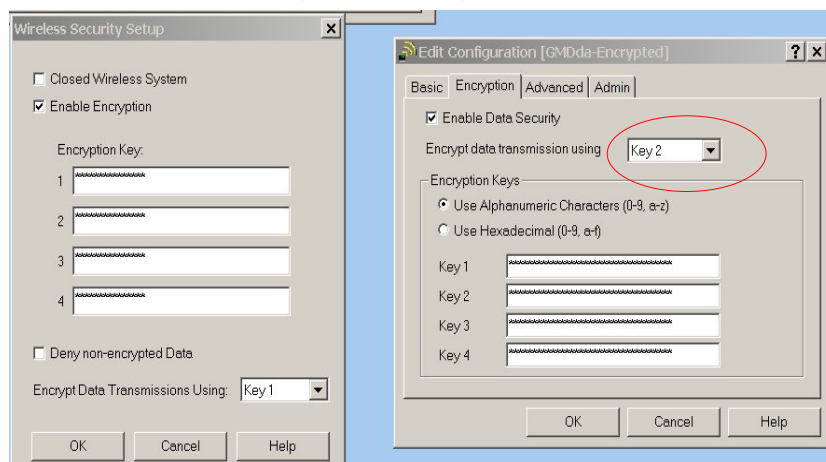
zwei Authentifikations-Schemata:

1. Open System Authentication: **keine Authentifikation**
Zugangskontrolle **allein** auf der Basis der **Identität!**
2. Shared Key Authentication:
symmetrisches Challenge-Response-Verfahren

Beispiel WLAN (cont.)

Vereinbaren des **Pre-Shared Secrets** unter WEP-Protokoll:

- **manuelles** Konfigurieren eines **gemeinsamen WEP-Schlüssels**
- **4** Schlüssel sind möglich, festlegen auf **einen!**



Beispiel WLAN (cont.)

WEP-Shared Key Authentifikation mit CR:

- Client meldet sich beim AP an mit **Request-Nachricht**, darin steht der **Index des zu verwendenden Schlüssels**
- **Pragmatik:** meist wird der erste Schlüssel gewählt
- Access Point sendet eine **128 Byte lange Zufallszahl RAND**
- Client berechnet Response: $C = \text{RAND} \text{ xor } \text{RC4}(K)$, (RC4 ist eine Stromchiffre, vgl. Kapitel 7)
- AP prüft Antwort C: $C \text{ xor } \text{RC4}(K) = \text{RAND}?$

Werden mit WEP-802.11 die **Angriffe von Folie 12 abgewehrt?**

- wenn **ja**, wodurch?
- wenn **nein**, **welche Angriffe** sind wie möglich?
- wenn nein: Armutszeugnis für Standardisierung, oder?

8.1. 4 Konkrete Einmal Passwort-Verfahren

OTP (One Time Password, RFC 2289)

- **Software-Lösung:** S/Key Verfahren
- **Hardware-basierte Lösung:** ID-Token

Software-Lösung:

- Ursprüngliches Konzept von L. Lamport: Lamport-Hash durch **S/Key-Verfahren** implementiert

S/Key-Verfahren (in verteilten Unix-Umgebungen)

- Beibehalten des ‚normalen‘ Passwort-Login für Benutzer
- Ersetzen des Unix-Logins bei entferntem Login

Parteien: Benutzer, lokaler Arbeitsplatz (PC), entfernter Server

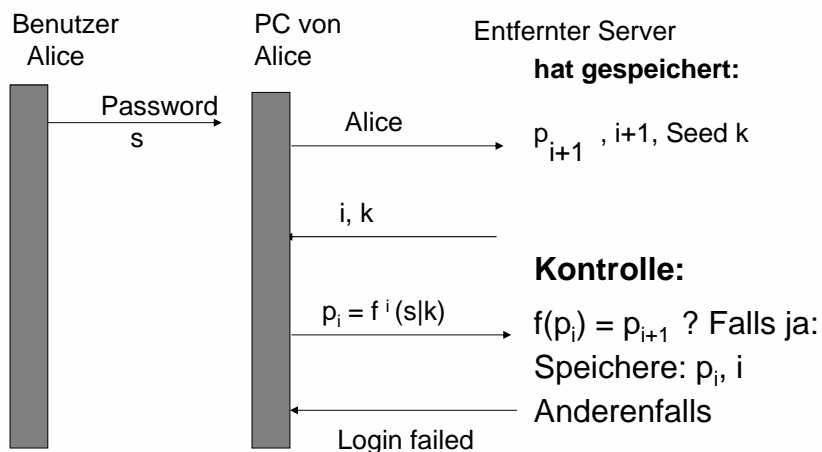
• **Benutzer:**

- besitzt geheimes Benutzer-Passwort s ,
- s ist Pre-Shared Secret zwischen Benutzer und PC,
- Server kennt s nicht

Vorgehen: vorbereitende Schritte

- PC berechnet aus s einmal benutzbare Passworte p_i
- dazu notwendig:
 - kryptographische Hashfunktion f (z.B. MD5)
 - Wahl einer Zahl N , Wahl eines Seed-Wertes k
 - $p_1 = f(s|k)$, $p_2 = f(p_1)$, ..., $p_N = f^N(s | k)$
- Übertragen des Startwertes an entfernten Server:
 - p_N , N , Seed-Wert k und Nutzer-Kennung


Protokoll: Authentifikation bei entferntem Server



Sicherheitsanalyse des Protokolls:

- was ist gut? Wozu Hashfunktion? Wozu seed?
- bestehen doch noch Angriffsmöglichkeiten, Ursachen dafür?

Hardware-basierte OTP-Verfahren: ID-Token

- **OTP**-Verfahren zur Authentifikation beim Server
Beispiel: Online-Banking
- Benutzer erhält ein Hardware-Token 
- **Token** besitzt eindeutige Nummer; Server kennt diese
- Token berechnet periodisch eine neue Zahl (Code),
z.B. alle 60 Sekunden, **Code ist das OTP**
- Tokencode hängt von einem Seed ab, ggf. der Zeit, der
Seriennummer des Tokens und ggf einer PIN
- Benutzer: Eingabe des OTP an einem Terminal
dazu: ablesen des **Passwortes vom Display des Tokens**
- **Server** muss: Seed, Zeit, Seriennummer, PIN kennen,
generiert auch Passwort und vergleicht beide

Beispiel RSA SecureID Token (sehr weit verbreitet)

Charakteristika: 2-Faktor-Authentifikation (Wissen und Besitz),

- **Zeit-synchronisiertes** Vorgehen
Synchronisation von Server (RSA ACE/Server) und Token
- Admin des Servers richtet Benutzer-Account ein, mit:
 - Token-Nummer und 64-Bit Seed **s**
 - Seed **s** wird auch auf Token gespeichert
 - Token wird an Benutzer ausgegeben

Erzeugen von OTPs:

- **alle 60 Sekunden** generieren Token u. Server neues Passwort
AES-Hashwert: Tokencode = AES(TokenId | s | Zeit)

RSA-Secure ID gibt es in verschiedenen Formfaktoren

- **Key Fob:** zur Authentifikation muss Benutzer PIN und One-time Tokencode übertragen



- **SecureID Card** (Kreditkartenformat)



- **PIN-Pad Karte**

- Eingabe der PIN über 10 Zeichen-Tastatur,
- Karte berechnet Hash von PIN und Tokencode
- Benutzer muss Hashwert beim Login angeben



Authentifikation durch Besitz

Hardware-Token: u.a. Smartcards, USB-Token

SIM-Kartenleser
mit USB



8.2. Smartcard

- Sichere Identifikation
- Sicherer Datenspeicher
- Sichere Ausführungsumgebung
- **Bsp:** Finanzbereich: GeldKarte, Kreditkarte, ec-Karte
Bürger: Telefonkarte, SIM, Dienstausweis,
Gesundheit: eGesundheitskarte (eGK) (**ab 1.1. 2006**), HPC



8.2.1 Hintergrund

Smartcard-“Player“:

- Chip-Produzenten (Siemens, Motorola, Thompson,...)
- Kartenhersteller (Gemplus, ORGA, Giesecke & Devrient, Schlumberger,...)
- Kartenherausgeber (Krankenkassen, Banken, Telcos,...)
- Karteninhaber (Kunden)

Lebenszyklus einer Smartcard

- **Fertigungsphase** (Kartenhersteller): Vom Wafer zur Karte, Test der Karte, Einbringen d. Personalisierungsschlüssels
- **Personalisierung**
Einbringen von personalisierten Daten: PIN, PUK, usw.
- **Nutzung der Karte ...**
- **„Invalidation Phase“** Sperrung der Karte im Hintergrundsystem

Typen von Karten

- **Speicherkarten**

- Preiswert (< 1 EURO)
- Stellt geringe Anforderung an Kartenleser ("CAD")
- Reines "Transportmedium" für Daten
- Speicherplatz 100 Byte - 8 KB
- Relativ niedriger Sicherheitslevel
- **Typische Anwendungen:**
 - "Prepaid-Karten" (Telefonkarte, usw.)
 - Krankenversicherungskarte
 - "Loyalty-Cards" (LH Miles & More, usw.)
- **Bem:** diese Klasse von Karten ist **noch keineswegs ‚smart‘**

Typen von Karten (cont.)

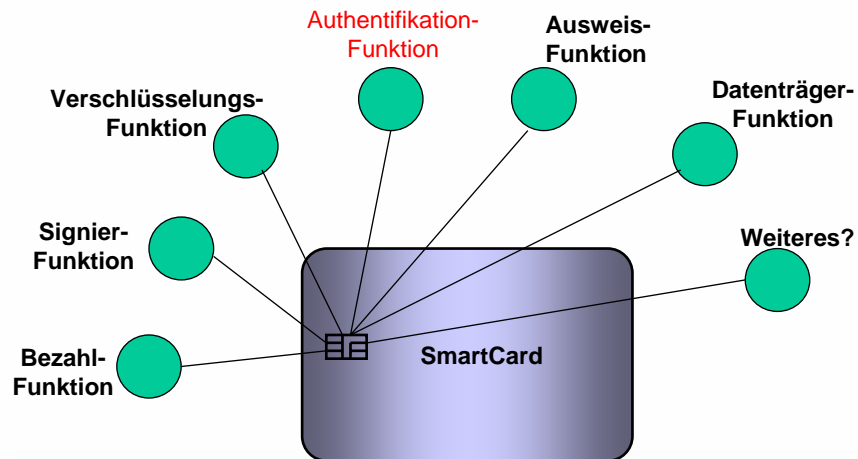
- **Prozessorkarten, Smartcards**, Preis 2-10 EURO
 - Karte mit internem Prozessor (8-Bit Worte, auch 32-Bit)
 - Karte kann intern Berechnungen durchführen
 - Trend: nachladbare Anwendungen, VM in der Karte u.a. JavaCard, Smartcard for Windows,...
- **Kontaktlose Karten** (Stromversorgung durch Induktion)

Programmierung von Smartcards: **zu beachten** u.a.

- Starke **Beschränkung der Ressourcen** (Kampf um jedes Byte)
- Software in der Karte ist **schwer zu debuggen**
- **"Patches"** des BS sind nach der Kartenausgabe **kaum** möglich
- Bei laufendem Programm kann jederzeit der Strom ausfallen:
Datenverlust!

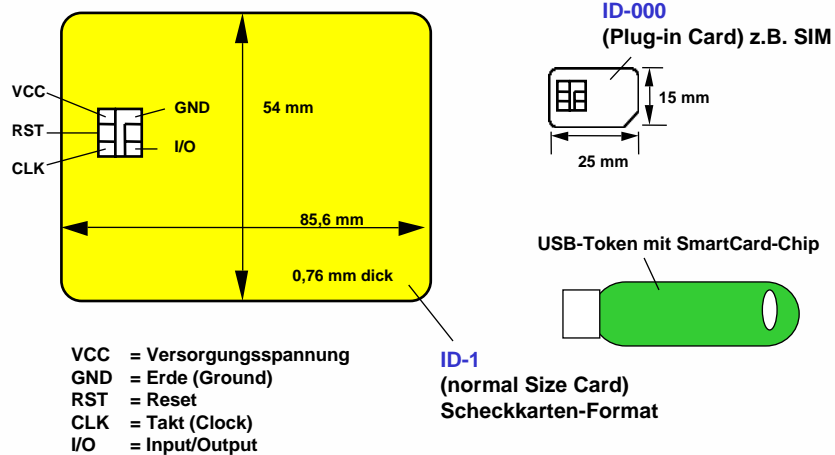
8.2.2 Smartcard als Trusted Device:

Smartcard bietet verschiedene Sicherheitsdienste, Authentifikation ist nur einer davon!



8.2.3 Kartenformate

ISO Standard 7816 legt Dimensionen, Protokolle etc. fest

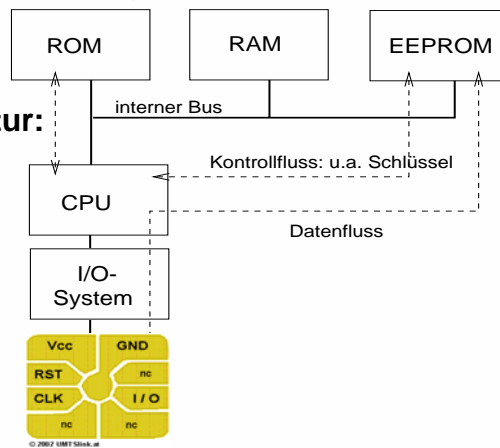


8.2.4 Architektur eines Smartcard-Mikrocontrollers

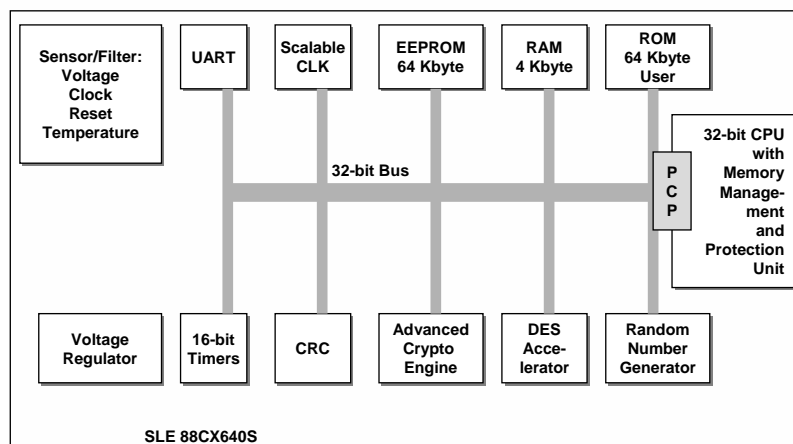
Bem.: Mikrocontroller ist der zentrale Baustein einer Chipkarte, unter dem Kontaktfeld der Karte eingebettet

Mikrocontroller-Architektur:

- CPU,
- ROM, RAM
- Ein/Ausgabe-Kanäle
- EEPROM
- Busse



Beispiel: Block-Diagramm eines High-end SmartCard Chips von Infineon



Komponenten des Smartcard-Mikrocontrollers

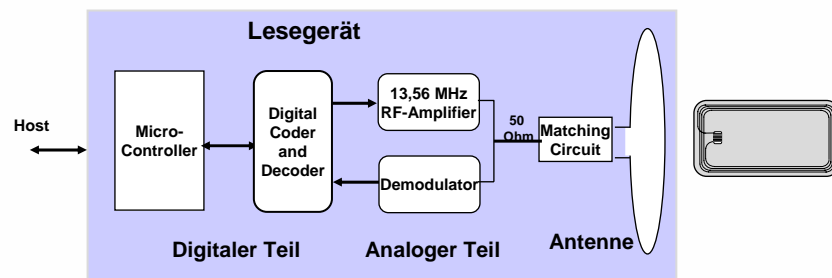
- **CPU**: 8-Bit, 16-Bit oder 32-Bit Prozessor (high-end Chip)
- **ROM** (Read only memory):
 - typische Größen 17-32KB, high end 320KB
 - Maskenprogrammiert: **Bedeutung?**
 - Karten-Betriebssystem, COS (Chip Operating System (COS))
 - enthält u.a. Verfahren zur PIN-Prüfung, kryptografische Verfahren, ...
- **RAM** (Random access memory):
 - Größen: 256- 512 Bytes, bis 16KB im high-end Bereich
 - flüchtiger Speicher, **Bedeutung?**
 - dient als Arbeitsspeicher: schnelle Lese- und Schreibzugriffe

Komponenten des Smartcard-Mikrocontrollers (cont.)

- **EEPROM** (Electrically Erasable Programmable):
 - typische Größen: 8-32KB, 132-400KB im high-end Bereich
 - nicht-flüchtiger Speicher, langfristige Speicherung von benutzerspezifischen, veränderbaren Daten,
 - **Beispiele** für Daten im EEPROM?
- **I/O-System**:
 - bitserieller Datentransfer, Datenrate mind. 9.6Kbit/s
 - Kommunikation mit Kartenleser über standardisierte Protokolle: idR über T0, T1 (siehe Folie 37)
- Optional: zusätzliche Komponenten, **Additional Units** (AU),
 - z.B. Krypto-Co-Prozessor, MMU, ein Zeitgeber, UART

8.2.5 I/O-Systeme:

- Kommunikation zwischen Lesegerät und Karte
- Anfangsfrequenz zwischen 1 und 5 MHz (üblich 3,57 MHz)
- Existierende Karten bieten **half-duplex Protokolle** (9,600 bit/s)
- Kommunikation: die Kommunikation wird **immer vom Terminal** initiiert, Terminal ist ‚Client‘, Karte ist ‚Server‘



Kommunikation zwischen Terminal und Karte (vergrößert):

- Karte erhält VCC und CLK, führt "power-on-reset" durch
- Karte sendet "answer to reset" (ATR) an Terminal
- **ATR-Nachricht** enthält Informationen u.a.
 - unterstützte Übertragungsprotokolle (z.B. T=1 (s.u.))
 - Baud Rate Adjustment and Clock Rate Conversion Factors
 - Historical Bytes (z.B. Info über Chip und Betriebssystem)
- Terminal sendet **Kommandos an Karte** (siehe Folie 36)
Beispiel: Kommando mit der Aufforderung zur PIN-Prüfung:
VERIFY CHV [PIN] [PIN Id]

Wichtigste Übertragungsprotokolle

- **T=0** Asynchron, halb-duplex, **byteorientiert** (ISO/IEC 7816-3)
- **T=1** Asynchron, halb-duplex, **blockorientiert** (ISO/IEC 7816-3)

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 8 36

Command

CLA	INS	P1	P2	Lc	Data field	Le
-----	-----	----	----	----	------------	----

CLA = Class (1 byte; Anzeige Command class, Command Chaining, Secure Messaging, Channel) z.B. GSM=A0
 INS = Instruction (1 byte)
 P1, P2 = Parameter (je 1 byte)
 L = Length (1 byte oder 3 bytes)
 c = command data field
 e = expected data
 SW1, SW2 = Status words (je 1 byte)

Response

Data field	SW1-SW2
------------	---------

Application Protocol Data Unit (APDU)

Prologue field			Information field	Epilogue field
NAD	PCB	LEN	[INF]	EDC
1 byte	1 byte	1 byte	up to 254 bytes	1 byte

← Error Detection Code →

Block-Typen:
 - Informations-Block
 - Receive-Ready-Block
 - Supervisory Block

APDU bei T1

NAD = Node Address Byte
 PCB = Protocol Control Byte
 LEN = Length
 EDC = Error Detection Code

Prof. C. Eckert FG Sicherheit in der Informationstechnik

TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 8 37

T=0: Asynchronous Character Transmission Protocol

- Keine Trennung zwischen TPDU und APDU
- Einfach zu implementieren (ca. 200 byte Code)
- Übertragung von Daten bei Command/Response nur in einer Richtung
- Antwortdaten müssen (falls vorhanden) mit GET RESPONSE-Kommando abgeholt werden
- wird z.B. von der SIM-Karte verwendet (GSM-System)

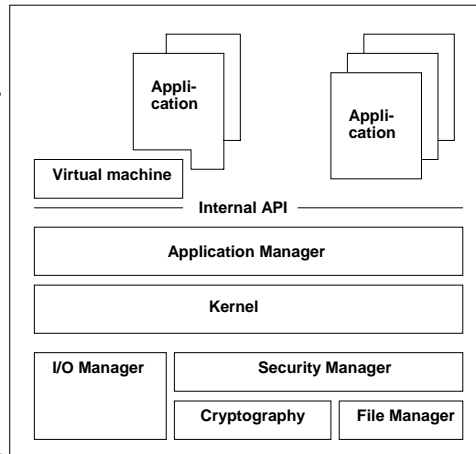
T=1: Asynchronous Block Transmission Protocol

- Saubere Trennung zwischen TPDU und APDU (siehe Folie 36)
- schwieriger zu implementieren (ca. 800 bytes Code)
- Übertragung von Daten in beide Richtungen möglich

Prof. C. Eckert FG Sicherheit in der Informationstechnik

8.2.6 Kartenbetriebssystem

- **Kernel:** u.a. Speicher-, Task-Management
- **File Manager:** Dateiverw.
- **I/O-Manager:** Datenverkehr (ATR, Senden, Empfangen, Error-Handling)
- **Application Manager:** mit dem Kommando-Interpreter
- **Security Manager:** Sicherheitskontrolle, Krypto & Secure Messaging
- **Interpreter:** Abwicklung von Applets und ausführbarem Code



Interface des Karten-Betriebssystems zum Terminal

- Standards (z.B. ISO/IEC 7816-4, GSM 11.11) definieren Instruktionssätze
- Verschiedene Karten variieren sehr stark

Beispiele:

- Datei-Operationen (Select, Read, Write, Seek,...)
z.B. *SELECT FILE [FID (EF, DF, MF) | AID (DF) | Pfad | ...]*
Return code zeigt an, ob Datei gefunden
- Authentifikation, Verschlüsselung
z.B. *VERIFY CHV [PIN] [PIN Id]*
CHANGE CHV [PIN Alt] [PIN neu] [PIN Id]
INTERNAL AUTHENTICATE [RAND] [Alg. Id] [Key Id]
GET CHALLENGE
EXTERNAL AUTHENTICATE [encAlg(Key, RAND)] [Alg. Id] [Key Id]
zur Auth. des Terminals gegenüber der Karte

Dateisystem einer Smartcard

Dateibaum im EEPROM (mit der Festplatte vergleichbar)

3 Klassen von Dateien:

- **Master File (MF)**: Wurzel des Dateibaums
- **Dedicated File (DF)**: "Directory für Anwendungen"
(Separierung von Daten aus Sicherheitsgründen)
- **Elementary File (EF)**: enthält Daten
 - external EF: auch außerhalb der Karte zugänglich
 - internal EF: nur innerhalb der Karte zugänglich

Zugriffsoperationen für EF-Dateien:

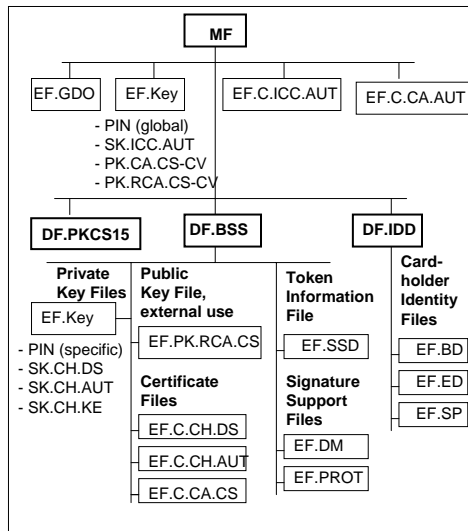
- APPEND, DELETE FILE, INVALIDATE Sperren der Datei,
- LOCK Reservieren der Datei, READ/SEEK Leses/Suchen
- REHABILITATE Entsperren, WRITE/UPDATE Schreiben

Zugriffskontrolle

- **Always (ALW)**: Keine Einschränkung.
- **Card holder verification 1 (CHV1)**:
Zugriff erst nach Eingabe der PIN #1.
- **Card holder verification 2 (CHV2)**:
Zugriff erst nach Eingabe von PIN #2.
(i.d.R. zum Zurücksetzen der Karte nach Falscheingabe von PIN1)
- **Administrative (ADM)**:
Kartenherausgeber, bzw. -hersteller definiert Zugriffsrechte und deren Verifikation.
- **Never (NEV)**: Niemals zugänglich

Bem.: PINs liegen üblicherweise in Dateien (z.B. EF_{CHV1})

Beispiel: Smartcard-Dateisystem für einen elektronischen Dienstausweis



MF = Master File (Root)
 DF = Dedicated File
 (repräsentiert eine Anwendung)
 EF = Elementary File (beinhaltet Daten oder Keys)
 GDO = Global Data Objects
 C = Certificate
 ICC = Integrated Circuit(s) Card
 AUT = Authentication (Key Usage)
 CA = Certification Authority
 RCA = Root CA
 CS = CertSign (Key Usage)
 CV = Card Verifiable Certificate
 PKCS = Public Key Crypto Standard
 BSS = Basic Security Services
 PK = Public Key
 IDD = Identity Data
 DS = Digital Signature (Key Usage)
 KE = Key Encipherment (Key Usage)
 SSD = Security Service Descriptor
 DM = Display Message
 PROT = Protokollierung
 BD = Basic Identity Data
 ED = Extended Identity Data
 SP = Specific Privileges

8.2.7 Sicherheitseigenschaften des Mikrocontrollers

- **Chip-Design:** manuelles Layout,
 - Vermeidung regelmäßiger Strukturen
 - **security through obscurity**, hier aber nur als Ergänzungen
- **interne Busse** sind nicht nach außen geführt,
 - d.h. nicht kontaktierbar,
 - Adress-, Daten-, Steuerbus: nicht abhörbar/beeinflussbar
- **Scrambling der Busse:** Funktionszuordnung verschleiern
- Verlagerung des **ROMs in tiefere Siliziumschichten** u.a. um
 - Reverse Engineering des Betriebssystems zu erschweren
- **Ionenimplantierte ROM-Codes**
 - verhindert das Auslesen des ROMs mittels Mikroskop,
 - Modifikation des Codes ist kaum noch möglich

- **RAM** ist gegen Störungen unempfindlich
- spezielle **Abschirmung** des **EEPROMs**, um das Aufzeichnen elektrischer Abstrahlung von außen zu verhindern
- Entfernung des Schutzschildes führt zur **Zerstörung** des Chips
- **weitere Schutzschicht** verhindert, dass die Speicherinhalte durch UV-Strahlung gelöscht werden können
- spezielle **Sensoren** zur Erkennung von Angriffversuchen, z.B.
 - Sensoren, die auf Licht reagieren
 - Wärme-Sensoren gegen Übertaktung des Chips
 - Widerstands- oder Kapazitätsmessung: erkennen, ob die Schutzschicht über der Schaltung noch vorhanden ist.
- **Überwachung** von Spannung/CLK-Frequenz

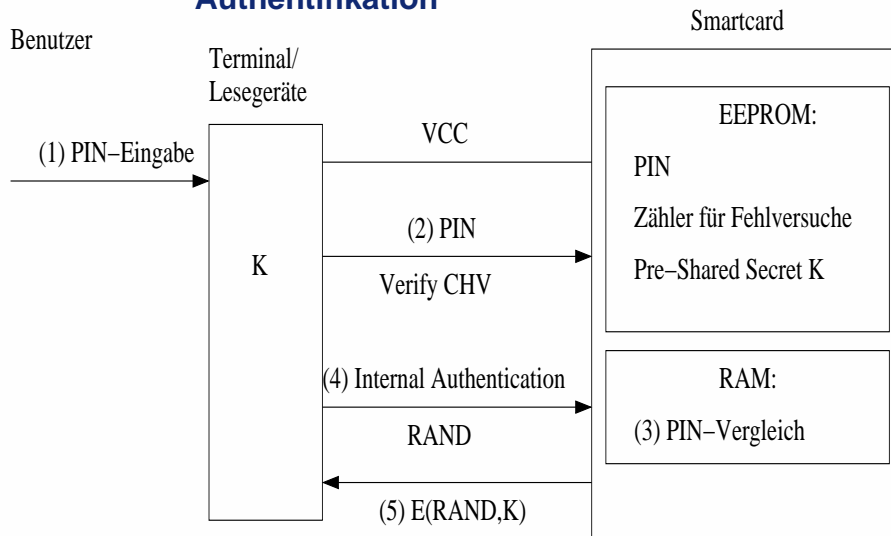
- **Irreversible Konversion** von Test- zu User-Mode bei der Chip-Fertigung,
 - im Test-Modus ist noch Zugriff auf den Speicher möglich
- Sicherstellung eines **in etwa identischen Stromverbrauchs** bei allen Chip-Instruktionen,
 - Abwehr von Seitenkanalangriffen, z.B. einfache und differenzielle Power-Analyse, **Bedeutung?**
- **verschlüsselte Kommunikation** zwischen
 - Lesegerät und Karte (Secure Messaging Protokoll)
 - zwischen den On-Chip-Komponenten
- Prüfsummen oder Signaturen zum Schutz der im EEPROM gespeicherten Schlüssel, PINs etc.

8.2.8 Authentifikation mittels Smartcard

- **Benutzer vs Karte:** idR PIN (CHV Card Holder Verification)
 - PIN in Datei in EEPROM gespeichert
 - Überprüfung: Abgleich wird in RAM ausgeführt
 - Kommandos: u.a. *Verify CHV, Change, Unblock CHV, Reset Retry Counter, Disable CHV* (z.B. bei GSM möglich)
 - Anforderung: Kommandos müssen gegen Analysen des elektrischen bzw. Zeitverhaltens stark sein
z.B. **Stromverbrauch** darf nicht unterschiedlich sein, wenn PIN falsch, oder korrekt ist
- **Karte/Anwendung versus Terminal:**
 - Challenge Response (idR symmetrisch)
 - Kommando: *internal Authentication*

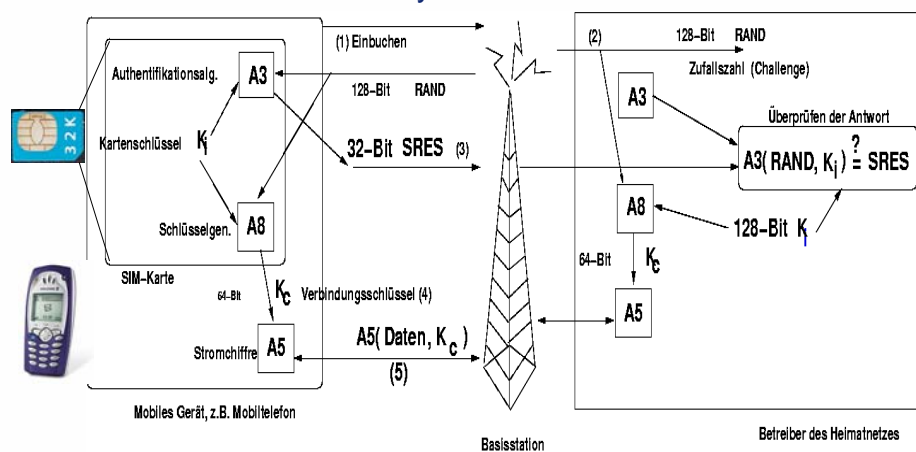
- **Terminal versus Karte:** Challenge Response
 - Kommando *external Authentication*:
Terminal fordert von Karte Random an (*ASK Random*), falls Terminal authentisch: BS ändert Zustand, Terminal erhält r/w-Rechte
- **Wechselseitige Authentifikation:** *Mutual Authentication*
 - Terminal fordert Kartenummer an (*Get Chip Number*)
 - Terminal fordert Random **R1** von Karte (*Get Challenge*), Terminal erzeugt selber **R2**
 - Terminal sendet: $E_CBC(R1|R2|Chipnummer, Key) = C$
 - Karte entschlüsselt C und prüft **R1** und Chipnummer
 - Karte erzeugt Antwort: $E_CBC(R2|R1, Key) = C'$
 - Terminal entschlüsselt C' und prüft **R2, R1**

Zusammenfassung der Schritte bei der Authentifikation



Beispiel: Authentifikation mittels der SIM-Karte

Architektur eines GSM-Systems



Schritte 1 – 3: Challenge-Response zur Authentifikation des Geräts

Schritt 4: Erzeugen des Verbindungsschlüssels

Schritt 5: Verschlüsseln der Daten

SIM-Karte:

Identifikation (u.a.): (EEPROM)

- International Mobile Subscriber Identity **IMSI**
- Temporary Mobile Subscriber Identity **TMSI**

Sicherheitsrelevante Daten: im EEPROM

- Individual Subscriber Authentication Key:
128-Bit Schlüssel **Ki**
- **PIN** zur Benutzer-Authentifikation,
- **PUK** PIN unblocking Key

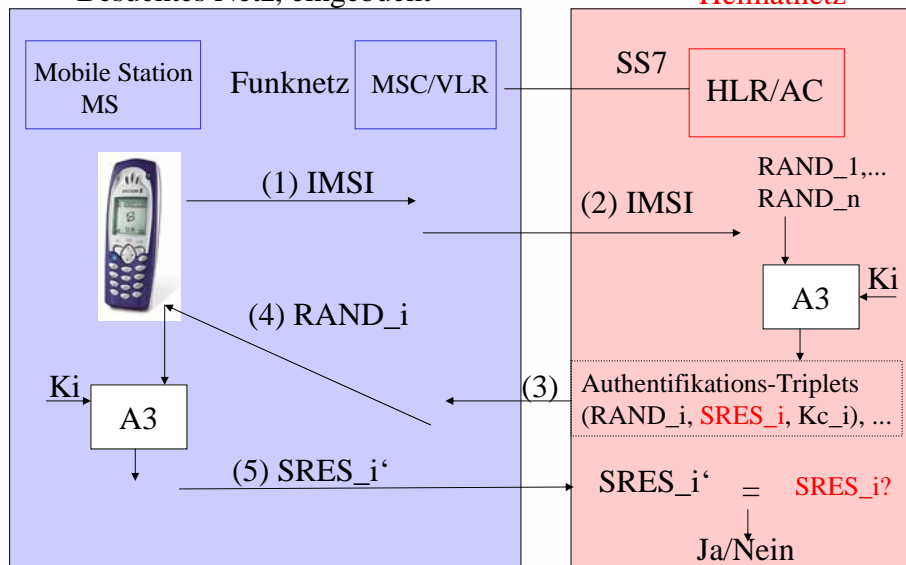
Kryptographische Verfahren: im ROM

U.a. Authentifikationsalgorithmus **A3**

Challenge-Response bei GSM/GPRS

Besuchtes Netz, eingebucht

Heimatnetz



8.2.9 Beispiel: die deutsche elektronische Gesundheitskarte

Spezifikation unter: http://www.sit.fraunhofer.de/german/SICA/sica_projects_gesundheitskarte/index.html

Original Layout
der eGK

Links: Vorderseite mit
Foto u. Braille-Code
Rechts: Rückseite



Chip:

- Mikroprozessorkarte mit Crypto-Controller, kontaktorientiert
- Dual-Interface mit Interface-Kontrolle, d.h. COS kann Nutzung einer Anw, auf ein bestimmtes Interface einschränken
- Übertragungsraten mindestens 115 kbps
- Speicherkapazität (EEPROM) noch offen (32 / (48) / 64 KB)

Betriebssystem der eGK

- Native **COS-Plattformen**: Betriebssystem enthält Code für die Kommandobearbeitung und die Sicherheitsfunktionen, oder
- **Interpreter-Plattformen** (Applet-Konzept);

Beispiele für Betriebssysteme mit eGK-Relevanz:

- Gemplus: SECCOS, G+D: STARCOS + SECCOS
- ORGA: MICARDO, Siemens: CARDOS,
- T-Systems: TCOS

- Evaluierung nach Common Criteria erforderlich, z.B. unter Nutzung des Protection Profiles für Secure Signature Creation Devices

Personalisierung, Card Management, Downloading

- Kartenpersonalisierung und Downloading bleiben (weitgehend) herstellerspezifisch
- Dedicated Files (Anwendungen) und elementary Files (Daten-Files) sollen **nachladbar** sein
- Dedicated Files und elementary Files sollen nach Kartenausgabe **löschar sein**
- Für die Installation von Anwendungen wird eine neues Kommando (CREATE APPLICATION) benötigt, um schrittweise Herstellerunabhängigkeit zu erreichen
- Der Speicherbedarf einer Anwendung soll bei Installation noch nicht festgelegt werden, d.h. **Anwendung soll sich dynamisch ausdehnen können**

File Management – File-Typen

- Unterstützung von **transparenten Files** „beliebiger“ Größe z.B. für Rezepte, Versichertendaten, ...;
- eGK kennt nicht die inhaltliche Struktur der Datei
- Unterstützung von **linearen Record-orientierten Files** z.B. für Tickets (u.a. für Objekt auf eGK oder einem Server)
- Unterstützung von **zyklischen Record-orientierten Files** z.B. für Logging;
- **File-Managementoperationen:**
 - Erzeugen, Löschen, Lesen, Update, Suchen, Append
 - Aktivieren/Deaktivieren von Files
 - Neues Kommando: DEACTIVATE RECORD z.B. zum Verbergen von Tickets oder sonstigen Einträgen
- **Zugriffskontrolle** bei jeder Operation: z.B. wenn nur ein Arzt schreiben darf, Kontrolle, ob zugreifende Instanz = Arzt

PIN Management der eGK

- **mehrere PINs** können unterstützt werden
- vorgesehen ist eine **Cardholder-PIN**;
- falls eine Signatur-Funktion im Sinne des SigG hinzukommt, dann ist zum Schutz des Private Signature Keys eine zusätzliche **Signatur-PIN** erforderlich
- **Minimale Länge** der PINs ist bei der Kartenpersonalisierung **einstellbar** (mindestens 4).
- **PINs sind änderbar**, nur Ziffern als PIN zulässig
- PINs können **kartenglobal** (d.h. anwendungsübergreifend) oder **anwendungsspezifisch** (d.h. nur innerhalb der betreffenden Anwendung bekannt) sein.

Weitere Sicherheitsfunktionen der eGK:

- **Schlüsselgenerierung** in der Karte (Schutz des private Key)
- Funktionen zur **eGK-Echtheitsprüfung** (RSA, Zertifikate)
- **Authentisierung** einer zugreifenden Instanz (RSA-basiert)
- **Prüfung von Zugriffsrechten** auf Daten und Schlüssel
- eGK kann **X.509-basierte PKI-Dienste** unterstützen:
 - elektronische Signatur (fortgeschritten oder qualifiziert z.B. für elektronische Patienten-Verfügungen nutzbar
 - Dokumenten-Verschlüsselung / Entschlüsselung (Nutzung z.B. bei elektronischen Rezept)
 - Client/Server-Authentisierung
- die eGK **überprüft die Zulässigkeit** der Kommandos

- Bestandteil der asymmetrischen Authentifikation:
„[Card Verifiable Certificates](#)“,
 - beinhalten in beglaubigter Form
 - den [Public Key](#) und „[Certificate Holder Authorization](#)“
(= Autorisierungsinstanz + Rolle des Karteninhabers,
z.B. Arzt oder Apotheker)
 - Bei einer C-2-C-Authentisierung mit Zertifikaten von unterschiedlichen Certificate Service Provider (CSP), muss das [Cross-Zertifikat](#) präsentiert werden, das den Public Key der anderen Zertifizierungsinstanz enthält
- Bem.: unterschrieben von der eigenen Zertifizierungsinstanz, da in der eGK nur der Public Key der eigenen Zertifizierungsinstanz als Prüfschlüssel vorhanden ist

- Die Anzahl der benötigten Cross-Zertifikate ist klein, z.B. bei 3 CSPs, einer eGK-Nutzungsdauer von 4 Jahren und jährlichem Schlüsselwechsel der Zertifizierungsinstanz: 108
- [Import](#) der Cross-Zertifikate von www-Server möglich
- Eine [Certificate Revocation List \(CRL\)](#) wird nicht benötigt, da eine Zertifikats-Nutzung an die betreffende eGK gekoppelt ist: besteht das Versicherungsverhältnis, dann ist auch die Benutzung der betreffenden eGK und damit die Benutzung des CV-Zertifikats gegeben.

Bem.: Eine CV-Zertifikatsinfrastruktur ist also [anders](#) geartet als eine X.509-PKI-Infrastruktur

- Unterstützung von sog. „**Security Environments**“, d.h. es können unterschiedliche Sätze von Zugriffsregeln gleichzeitig vorhanden sein, um verschiedene Anwendungsszenarios wie z.B. lokaler Zugriff auf eine Karte oder Zugriff durch eine internet-basierte Instanz zu unterstützen
- Unterstützung eines „**Trusted Channels**“ durch Secure Messaging: jedes Kommando und jede Antwort wird mit einer kryptografischen Prüfsumme versehen, und durch
- **Security Conditions** kann Verschlüsselung gefordert werden
- Security Conditions, die Keys, Files und Datenobjekten zugeordnet sind, sind **statisch, aber flexibel gestaltbar** (z.B. Lesen der Arzneimittel-Dokumentation, darf von Arzt oder Apotheker ausgeführt werden, ...)

Sicherheitsfunktionen - Management der Zugriffsrechte auf eGK am Beispiel eines DatenFile

Elementary File	Security Attributes	<p>Beispiel 1: Lesen: Arzt oder Apotheker</p> <p>AM = Read SC = EXT AUTH (asym) mit CHA.Arzt OR CHA.Apotheker)</p>	<p>Beispiel 4: Update: Arzt + PIN.CardHolder</p> <p>AM = Update SC = EXT AUTH (asym) mit CHA.Arzt AND VERIFY mit PIN-ID = 'xx'</p>
	File Content	<p>Beispiel 2: Update: nur Arzt</p> <p>AM = Update SC = EXT AUTH (asym) mit CHA.Arzt</p>	<p>Beispiel 5: Update: Arzt + ZDE</p> <p>ZDE = ZugriffsDatenElement ZDE = 0: kein Zugriff ZDE ≠ 0: Zugriff erlaubt</p> <p>ZDE ist nach PIN-Eingabe durch den Patienten änderbar.</p> <p>Diese Variante ist in ISO 7816-4 nicht definiert und wird von keinem COS unterstützt.</p>

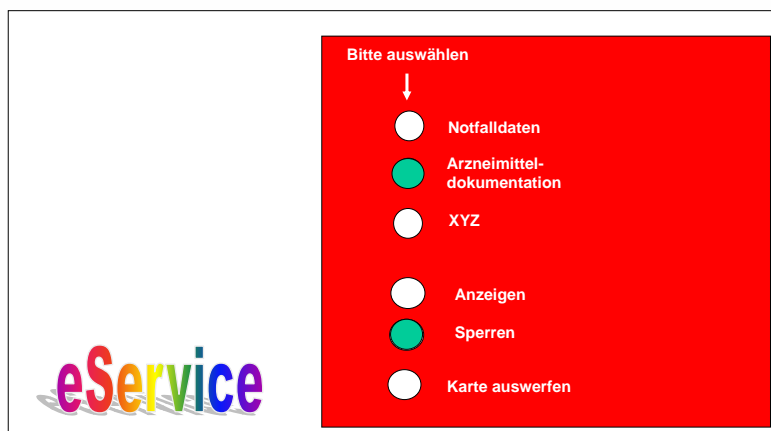
AM = Access Mode
SC = Security Conditions
CHA = Cert. Holder
Authorisation
(Authority.Role ID)

Beispielhaft: Patienten-Rechte-Handling –



Beispiel eines eService-Terminals
mit Sicherheitsmodul SMC.X
(Security Condition für Read Access zu
Files mit medizinischen Daten z.B.
EXT AUTH with SMC.X AND
PIN.Cardholder)

Patienten-Rechte-Handling
z.B. Sperren / Entsperrern der Nutzung von Anwendungen



Patienten-Rechte-Handling Z.B. Verbergen einzelner Einträge

Fazit: **Authentifikation mit Besitz**

- Smartcard-Einsatz wird **weiter zunehmen!**
z.B. Gesundheitskarte, Dienstaussweise, Studentenkarten,...
- Smartcard dient als **trusted, persönliches, personalisierbares Device!** u.a. vertrauenswürdige Signierumgebung
- Smartcard ist **unter Kontrolle** des Besitzers!
Unterschied z.B. zum TPM, das ist auch eine Art Smartcard, aber der TPM ist in Rechner integriert (TPM siehe spätere Kapitel)
- Viele **Sicherheitsmerkmale**, die die Smartcard absichern:
 - u.a. Nutzung als sicherer Schlüsselspeicher
 - stärkere Authentifikationsmechanismen mit Challenge Response oder auch ggf. Kombination mit Biometrie
- **USB-Token:** einfache Hardware-Unterstützung

8.3 Authentifikation mittels Biometrischer Merkmale

Biometrisches Merkmal:

Verhaltenstypische oder physiologische Eigenschaft eines Menschen, die diesen eindeutig charakterisieren

Anforderungen an biometrische Merkmale:

- **Universalität:** Jede Person besitzt das Merkmal
- **Eindeutigkeit:** Merkmal ist für jede Person verschieden
- **Beständigkeit:** Merkmal ist unveränderlich
- quantitative **Erfassbarkeit** mittels Sensoren
- **Performance:** Genauigkeit und -geschwindigkeit
- **Akzeptanz** des Merkmals beim Benutzer
- **Fälschungssicherheit**

Unterschiede zur wissensbasierten Authentisierung:

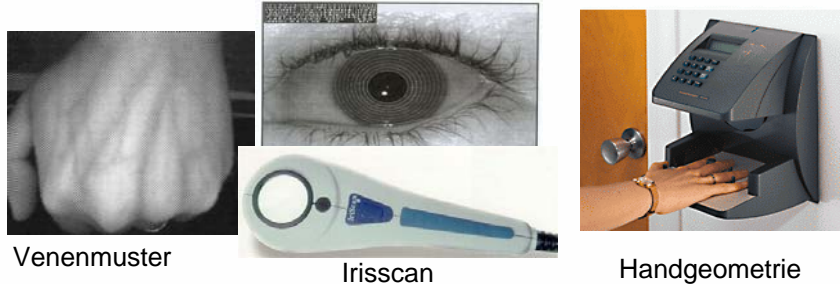
- Merkmal ist Personengebunden: **Konsequenz?**
- Charakteristische Merkmale müssen extrahiert und mit Referenzwert verglichen werden: **Probleme?**

8.3.1 Klassen biometrischer Merkmale

- **physiologische Merkmale (statisch):**
keine oder nur sehr begrenzte Möglichkeiten zur Auswahl oder Änderung von Referenzdaten
- **Verhaltensmerkmale (dynamisch):**
Merkmal ist nur bei bestimmter Aktion vorhanden;
Möglichkeiten zur Auswahl/Änderung von Referenzdaten

Beispiele für dynamische biometrische Merkmale
Unterschriften-Dynamik, Sprache, Tippverhalten (Keystroke)

Beispiele für statische biometrische Merkmale:
Fingerabdruck, Gesichtsbild, Handgeometrie, Retina
Venenmuster, Iris



Venenmuster

Irisscan

Handgeometrie

8.3.2 Einsatz und Anwendbarkeit

- **Fingerabdruckerkennung:** weit verbreitetes Verfahren für viele Anwendungen: preiswerte, kompakte Sensoren sind verfügbar (auf Handy, PDA, Laptop, ...)
- **Schreibdynamik** ist besonders geeignet für Willenserklärung (z.B. Signaturen): Unterschreiben als bewusster, willentlicher Akt
- **Iris-Erkennung** sehr fälschungssicher, aber aufwändige Technik und geringe Akzeptanz (vgl. Fraport-Pilot-Projekt)
- **Gesichtskontrolle:** gute Akzeptanz, aber zZ noch nicht ausgereift, vgl. <http://www.bsi.bund.de/literat/studien/biop/biopabschluss.pdf>; Studie zu Gesichtserkennungssystemen zum Einsatz bei Lichtbildausweisen

Gesichtserkennung

- z.B. mit **Elastic Bunch Graph Matching-Verfahren**.
- **Gitternetz** wird in Form eines markierten Graphen über das Gesicht gelegt und
- markante Stellen werden mit Knoten des Graphen, den **Landmarken**, markiert
- als Referenzwerte werden die **Länge** und der **Winkel** jeder Kante des Graphen gespeichert.
- **Authentifikation:**
 - Lokalisierung des Gesichts und der Landmarken,
 - Graph wird mit Referenzwert verglichen,
 - d.h. **Vergleich der Längen und der Winkel der Kanten**.



Beispiele für **nicht erfolgreichen** Einsatz von Gesichtserkennungssystemen:

- (1) 2003: Gesichtserkennung am **Bostoner** Flughafen
 - Im Sommer 2002 wurde am Bostoner Flughafen getestet
 - es versagte in 96 Fällen, in 153 Fällen war es erfolgreich.
- (2) 2003 Gesichtserkennung in **Tampa**, Florida
 - flächendeckende Video-Überwachung mit automatischer Auswertung durch Gesichtserkennungs-Software
 - Versuch wurde abgebrochen, weil sich aus der Video-Überwachung keine Verhaftungen ergeben hätten.
- (3) 2003: Pilotversuch am **Nürnberger** Flughafen
 - elektronische Passbild-Überprüfung gescheitert:
 - Zitat: "Derzeit leistet der Prototyp des Modells weniger als das geschulte Auge eines Polizisten"

Andere Biometrie-Systeme:

Pilotprojekt ABG am Fraport

- "Automatisierten und Biometriegestützten Grenzkontrolle" (ABG)
- Start im April 2004, Testlauf war zunächst auf 6 Monate angelegt
- Reisepassdaten sowie Iris-Merkmale werden beim BGS registriert.
- ca. 5.000 Vielflieger haben beim BGS ihre Iris fotografieren lassen,
- Bei Ein- und Ausreisen: Selbst-Einscannen der Reisedokumente und ein Blick in eine Iris-Kamera
- hohe Treffergenauigkeit, aber geringe Akzeptanz bei Reisenden

Analoge Projekte:

- seit 2002 am Flughafen in Sydney und auch Schipol,
- Vereinigten Arabischen Emirate: ohne Iris-Check keine Einreise

Weiteres bekannt?

Frankfurt



Schipol

Aktuelle Diskussion: Biometrie in Ausweisdokumenten

USA: geplanter Start: 26. Oktober 2004:

- Für die Einreise in die USA wäre dann ein Pass mit biometrischen Merkmalen zur Identifikation des Einreisenden notwendig geworden (Einreise für 90 Tage ohne Visum)
- Verschoben auf den 26. Oktober 2005, betroffen u.a. D, Japan, Australien;

Deutschland:

- In Pässen und Personalausweisen dürfen neben dem Lichtbild und der Unterschrift weitere biometrische Merkmale von Fingern, Händen oder Gesicht des Inhabers aufgenommen werden.
- Alle Merkmale und Angaben über die Person dürfen auf den Ausweispapieren verschlüsselt gespeichert werden.
- Es darf keine bundesweite Datei eingerichtet werden.
- Die biometrischen Merkmale dürfen nur dazu verwendet werden, die Echtheit des Dokumentes und die Identität des Inhabers zu prüfen.

8.3.3 Vorgehen bei biometrischer Authentifikation

1. Messdatenerfassung durch **biometrischen Sensor** und Digitalisierung (Feature Extraction)
2. **Enrollment:** Registrierung eines Benutzers : Aufnahme, Auswahl und Speicherung der **Referenzdaten**
z.B. 5-7 verschiedenen Fingerabdruck-Werte
3. Bei Authentifikation: Erfassung der aktuellen **Verifikationsdaten** (mittels Sensoren)
4. Verifikationsdaten digitalisieren (u.a. ggf. normieren)
5. mit gespeichertem Referenzwert vergleichen, Toleranzschwellen sind notwendig



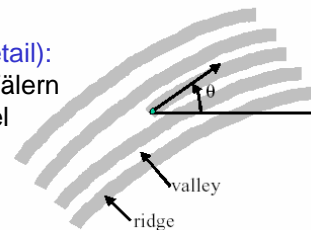
Beispiel: Fingerabdruck Sensor



Graustufen-Bild eines Fingerabdrucks

Merkmal: **Minutien** (lat. für Kleinigkeit, Detail):
End- und Verzweigungspunkte, Wirbel, Tälern
mit Ortskoordinaten und Tangentenwinkel

Muster von Rillen (ridge) und Tälern,
ist charakteristisch für jeden Mensch



Verarbeitung von Fingerabdrücken



Rillennmuster nach dem
Herausfiltern des Hinter-
grunds, z.B. Filtern von
Veränderungen durch Schmutz,
Verletzungen etc.



Feature-Extraktion: Bestimmen
der Minutien (10 –100) als Endpunkte
u. Verzweigungswinkel, Richtungen
Danach: Abgleich des Minutien-
Musters mit Referenzwerten:
Vergleich (match) benachbarter Minu-
tien

Häufig betrachtete Merkmale für Fingerabdruck-Minutien

- Ortskoordinaten x und y
- Linienwinkel θ

Darstellung einer Minutie als Tripel (x, y, θ)

Darstellung eines **Fingerabdrucks als Folge von Tripeln**:

$((x_1, y_1, \theta_1), (x_2, y_2, \theta_2), \dots, (x_n, y_n, \theta_n))$

Einfacher Algorithmus zum Minutienvergleich

- Abstand und Winkeldifferenz zweier Minutien $(x_i, y_i, \theta_i), (x_j, y_j, \theta_j)$:

$$d = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

$$\Delta\theta = \begin{cases} |\theta_i - \theta_j|, & \text{falls } |\theta_i - \theta_j| \leq 180^\circ \\ 360^\circ - |\theta_i - \theta_j|, & \text{falls } |\theta_i - \theta_j| > 180^\circ \end{cases}$$

Vergleichskriterium für zwei Fingerabdrücke

- Wähle **Toleranzschwellen** $dTol$ und θTol und einen
- „**Match-Score**“ k abhängig vom gew. Sicherheitsniveau.
- Zwei **Minutien** gelten als **übereinstimmend**, falls $d \leq dTol$ und $\Delta\theta \leq \theta Tol$.
- Zwei **Fingerabdrücke** gelten als **übereinstimmend**, wenn mindestens k übereinstimmende Minutien im Rahmen der Toleranzen $dTol$ und θTol gefunden wurden.

Iris-Erkennung:

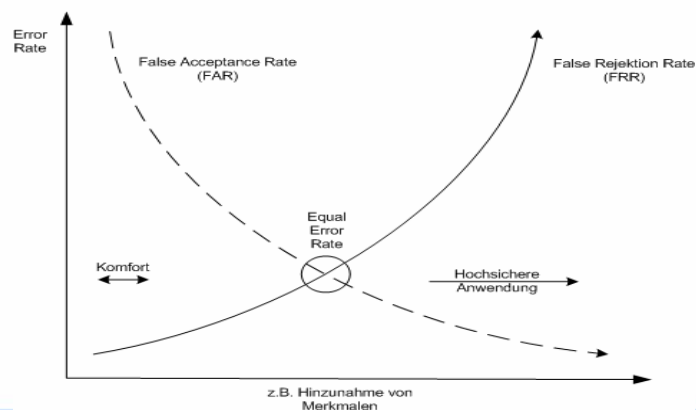
- idR auf der Basis des Algorithmus von J. Daugman;
- Überblicksartikel unter

<http://www.cl.cam.ac.uk/users/jgd1000/irisrecog.pdf>

8.3.4 Problembereiche

- Abweichungen zwischen Referenz- und Verifikationsdaten sind unvermeidlich,
- 2 Fehlertypen:
 - Berechtigter Benutzer wird abgewiesen,
⇒ **Akzeptanzproblem** (false negative)
 - Unberechtigter wird authentifiziert, Kontrollen zu locker
⇒ **Sicherheitsproblem** (false positive)
- Leistungsmaße zur Bewertung der Güte eines Systems
 - **False Acceptance Rate (FAR):**
Wahrscheinlichkeit für fälschliche Akzeptanz einer unberechtigten Person (false accepts)

- **False Rejection Rate (FRR):**
Wahrscheinlichkeit für fälschliche Rückweisung einer berechtigten Person
- **Equal Error Rate (EER):**
Gemeinsamer Wert für FAR und FRR



- **Bem.:**
 - Fehlerrate idR **ungeprüfte Angaben** von Herstellern
 - häufig in unrealistischen Szenarien erhoben
 - idR proprietäre Erfassungsgeräte, Vergleichsalgorithmen
- **Fehlende Standardisierung**, fehlende einheitliche Tests
- Probleme bei **Interoperation**: u.a. heterogene Umgebungen
z.B. bei Biometrie-basierten Kontrollen von Ausweisen:
 - Geräte/Software zur Erhebung von Referenzdaten sind nicht kompatibel mit Verifikationssoftware
 - Matching-Resultate fehlerhaft?!
- Ergebnisse einer 3-jährigen Studie
BioTrusT: www.biotrust.de: aktuelle Produkte besitzen noch **erhebliche Fehlerraten** (Stand 2002, aber noch aktuell)

8.3.5 Sicherheitsprobleme bei biometrischen Techniken

Angriffstrategien:

- **Direkte Täuschung** des biometrischen Sensors durch Attrappen u.a. Gummi-Finger
- **Einspielen von Daten** unter Umgehung des biometrischen Sensors
 - Wiedereinspielen abgehörter Daten (Replay-Angriffe)
 - Einspielen eigens verschaffter, digitalisierter Daten

Probleme: **enge Kopplung** zwischen Merkmal und Person

- (1) Bedrohung der informationellen Selbstbestimmung
- (2) Gefahren durch gewaltsame Angriffe gegen Personen
- (3) Problem der öffentlichen Daten und rechtliche Aspekte

ad (1) informationelle Selbstbestimmung (Grundrecht)

- Background: Volkszählungsurteil 1983

Das Grundrecht gewährleistet insoweit die Befugnis des Einzelnen, grundsätzlich selbst über die Preisgabe und Verwendung seiner persönlichen Daten zu bestimmen. Einschränkungen dieses Rechts auf "informationelle Selbstbestimmung" sind nur im überwiegenden Allgemeininteresse zulässig."

- Statische biometrische Merkmale: eindeutige Identifikation,
 - **kein Verändern** des Merkmals (der Referenzwerte) möglich
 - flächendeckende, ggf. **unbemerkte** Merkmalerhebung?
 - Erstellung von Aufenthalts- und Bewegungs**profilen**, z.B. am Arbeitsplatz?!
 - Analysen über **Gesundheits-, Gemütszustand, ...?**

ad (2) gewaltsame Angriffe

- Problem: Gewalt**kriminalität**: menschliche Schlüssel
- Lösungen? spezielle Kontrollen: Lebend-Checks, Notfallvorsorge, überzeugend?

ad (3) Problem der öffentlichen biometrischen Daten

- Behandlung wie für PINs, Passwörter, Schlüssel, aber
- Sicherheit des biometrischen Systems darf **nicht** von der Geheimhaltung der Daten abhängen,
- **authentische Herkunft** der Daten vom Sensor ist zu gewährleisten! Wie?

Ad (3): **Rechtliche Aspekte: z.B.**

Biometrie für Signaturerstellung:

- Deutsche Signaturverordnung (seit November 2001)
- erlaubt Biometrie als vollwertige Alternative, falls
 - **Mechanismenstärke hoch**: Problem?!
 - Bei **Mechanismenstärke mittel** nur als Sekundär-Methode erlaubt und
 - Unakzeptabel für **Mechanismenstärke niedrig**

Fazit Biometrie?

- Quo vadis Biometrie: Hype oder Zukunfts-Technologie?
- Lösung für digitale Identität im ubiquitous Computing?

8.4 Authentifikationsprotokolle

Basis: Client-Server-Architektur

8.4.1 Anforderungen

- Wechselseitige Authentifikation
 - Häufig gewünscht: **Single-Sign on**, Bedeutung?
 - Ggf. auch Schlüsselaustausch für vertrauliche und integere Datenübertragung
 - ‚Vater‘ aller Dinge: Needham-Schroeder Protokoll 1978
 - Basis: vertrauenswürdiger Authentifizierungsserver AS
 - Pre-Shared Secrets: jeder Client A hat geheimen **Master-Key** K_A mit AS vereinbart
- Aufgabe des AS: Authentifikation und Schlüsselverteilung

Szenario: Alice möchte mit Bob kommunizieren:

Notation: $\{M\}_{K_X}$: M wurde mit K_X verschlüsselt: $E(M, K_X)$

Protokollschritte des Needham-Schroeder Protokolls

Von	An	Nachricht
(1) Alice	AS	Alice, Bob, $Nonce1$
(2) AS	Alice	$\{K_{AB}, Bob, Nonce1, \{K_{AB}, Alice\}_{K_{Bob}}\}_{K_{Alice}}$
(3) Alice	Bob	$\{K_{AB}, Alice\}_{K_{Bob}}$
(4) Bob	Alice	$\{Nonce2\}_{K_{AB}}$
(5) Alice	Bob	$\{Nonce2 - 1\}_{K_{AB}}$

Nonce: (number used once) idR eine Zufallszahl

Problem: Protokoll enthält folgende Schwachstelle

- Bob geht davon aus, dass K_{AB} (Schritt 3) aktuell vom AS stammt

- Aber: Wiedereinspielungen mit geknackten K_{AB} sind möglich
- Lösung: Nachrichten mit **Timestamps** versehen,
aber: welches Problem handelt man sich damit ein?

Fazit:

- Entwicklung korrekter Protokolle ist nicht einfach,
- einige ‚Daumenregeln‘ für das Protokolldesign

8.4.2 Einfache Leitlinien zum Protokolldesign

Basis: Papier von Abadi und Needham 1996

Regel 1: Vollständige Information:

Alle relevanten Informationen (z.B. Namen der Partner) sollten in einer Protokollnachricht codiert werden.

Regel 2: Verschlüsselungszweck

klären, mit welchem Ziel Verschlüsselung angewandt wird:

- Gewährleisten der Vertraulichkeit,
- Nachweis der Authentizität des Absenders
- Verknüpfung unterschiedlicher Nachrichten-Bestandteile

Bem: unterschiedliche Schlüssel für unterschiedliche Zwecke verwenden, z.B. Signieren, Verschlüsseln

Regel 3: Vorsicht bei Doppelverschlüsselung

Redundanz verursacht unnötige Kosten, ggf sogar Lücken

Regel 4: Digitale (elektronische) Signaturen

Erst Signieren (nach dem Hashen), dann Verschlüsseln,
Warum?

Beispiel: CCITT X.509 Protokoll:

Austausch signierter, verschlüsselter Nachrichten

Protokollauszug:

- A soll Urheber der verschlüsselten Nachricht $\{ Y_A \}_{K_E^B}$ sein (Bem.: K_E^B ist der öffentliche Schlüssel von B)

- Nachricht A an B:

A, $\{ T_A, N_A, B, X_A, \{ Y_A \}_{K_E^B} \}_{K_D^A}$

Angreifer X

- kann die Signatur von A entfernen und verschlüsseltes Y_A mit seinem eigenen Schlüssel K_D^X signieren
- und diese modifizierte Nachricht an B senden:

X, $\{ T_A, N_A, B, X_A, \{ Y_A \}_{K_E^B} \}_{K_D^X}$

Regel 5: frischer Schlüssel

Frischenachweis über Zeitstempel oder Nonces
(vg. Problem bei Needham-Schroeder Protokoll)

Weit verbreitete Weiterentwicklung des Needham-Schröder Protokolls: **Kerberos-Protokoll**

8.4.3 Kerberos-Protokoll

Name: gr. Mythologie: **3-köpfiger Hund**, der den Eingang zum Hades bewacht.



- 1983 im Athena Projekt am MIT entwickelt (+ IBM,DEC)
- zZ im Einsatz: Version 4 (nur TCP/IP, einige Sicherheits-Probleme)
Version 5 (RFC 1510)

Ziele von Kerberos:

- **Authentifikation** von Subjekten, genannt Principals:
u.a. Benutzer, PC/Laptop, Server
- Austausch von **Sitzungs-Schlüsseln** für Principals
- **Single-Sign-on** für Dienste in einer administrativen Domäne (*realm*) (bzw. auch Inter-realm)

Design

- Pro Domäne **ein vertrauenswürdiger** Server
KDC = **Key Distribution Center**,
- Aufgabe: Authentifizierung der Clients seiner Domäne
- Hierarchie von Authentifikationsservern:
- Aufgabe: Bereichsübergreifende Authentifizierung:
transparent über die KDCs

Design (Forts.)

- Needham-Schroeder Variante mit Timestamps
- **Einsatz symmetrischer Kryptoverfahren:**
 - In Version 4: DES
 - In Version 5 nicht festgelegt, Nachricht enthält Tag, das das verwendete Verfahren identifiziert
 - **Bem.:** Microsoft hat eine Variante von Kerberos als proprietäres Protokoll in Windows2000XP eingesetzt, Variante kann auch asymmetrische Verfahren
- **Pre-Shared Secrets:**
- Benutzer-Authentifikation basiert auf **Passworten**
 - KDC verwaltet **gehashte** (MD5) Benutzer-Passworte

Master-Keys:

- jeder Principal A hat geheimen Master-Key K_A mit KDC vorab vereinbart
- bei Benutzern: aus gehashtem Passwort abgeleitet, mit DES-CBC u. **gleichem KDC-Key** verschlüsselt abgelegt
- Jeder Server-Rechner (Principal) S in der Domäne hat mit KDC ebenfalls geheimen Master-Key K_S vereinbart

Single-Sign-on (SSO) Funktionalität:

- **Ziel:** Zur Dienstenutzung eines der Principals muss der Benutzer sich nicht beim Principal authentifizieren
- **Kerberos-Ansatz:**
Principals müssen nicht selber Authentifikationsinformationen über die Benutzer verwalten!

Idee:

- **Trennung** der eigentlichen Authentifizierung (zentral), diese erfolgt durch den (die) KDC(s)
- und der **Prüfung** auf Aktualität, Plausibilität, Zulässigkeit dies erfolgt dezentral durch Server (Principals)

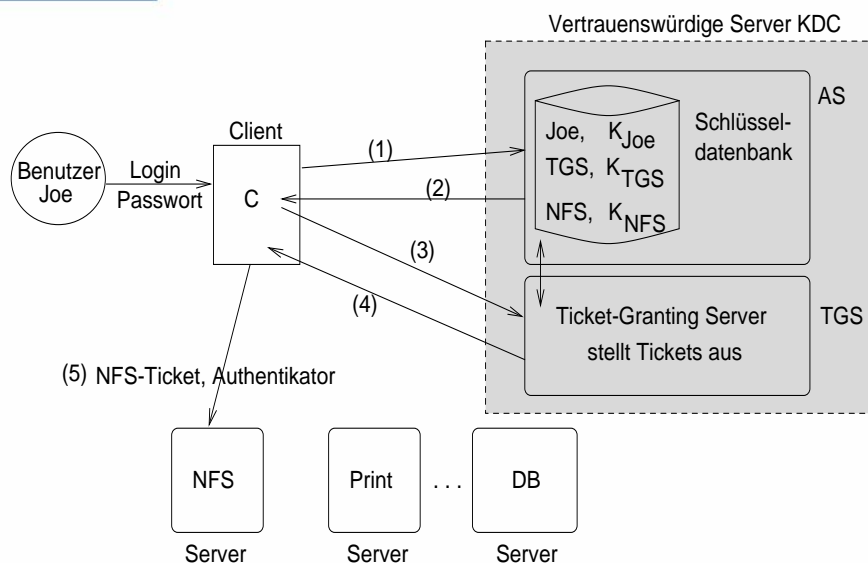
Vorgehen:

- Benutzer A (Principal) beantragt beim KDC einen Authentifikationsnachweis, um mit **Dienst S** zu kommunizieren
- der KDC stellt ein **Authentifikations-Ticket für Principal A** aus
- **Principal S** kann Gültigkeit des Tickets überprüfen

Bem.: Ticket enthält **keine Autorisierung** zur Dienstenutzung
d.h. Rechtemanagement ist dezentral, durch **S**

Vergößerter Ablauf: (vgl. Folie 98)

- Benutzer loggt sich mit Kennung und Passwort auf lokalem Rechner ein
- Lokaler Rechner sendet (Version 5) Kennung und aktuelle Systemzeit verschlüsselt mit Master-Key als Schlüssel an KDC und fragt ein Ticket für Server TGS an
- KDC extrahiert Benutzer-Master-Key aus seiner DB und prüft die verschlüsselte Zeit (< 5 Min.) auf Aktualität,
 - falls ok: Erstellt ein Ticket zur Nutzung des TGS und
 - Sendet das Ticket Granting Ticket (TGT) zurück
- Lokaler Rechner sendet TGT zum TGS (Ticket Granting Service): beantragt Ticket für Zugriff auf Server (z.B. NFS)



Zu klären:

- wie kann Principal S die Authentizität des Tickets prüfen?
- Wie weist Benutzer(rechner) nach, dass er berechtigter Besitzer des Tickets ist

Ticketstruktur: $T_{c,s}$ ist nur für den Principal **C** (z. B. **Joe**) und den Server **S** (z. B. **NFS**) gültig

- $T_{c,s} = S, C, addr, timestamp, lifetime, K_{c,s}$, es gilt
 - S Name des Servers, C Name des anfordernden Clients,
 - *addr* dessen IP-Adresse, aktuelle Zeit, Lebenszeit des Tickets (*lifetime*),
 - Sitzungs-Schlüssel $K_{c,s}$ für Kommunikation zw. S und C
- Ticket wird mit Master-Key von S verschlüsselt, $\{ T_{c,s} \}_{K_S}$

- Client erhält mit Ticket auch den gemeinsamen Schlüssel $K_{c,s}$ von KDC, verschlüsselt mit Master-Key des Benutzers: $\{ K_{c,s} \}_{K_C}$,

Authenticator-Konzept: (für dezentrale Prüfungen)

- **Idee:** Benutzer-Rechner muss dem Principal die Kenntnis des gemeinsamen Schlüssels $K_{c,s}$ nachweisen
- **Authenticator** wird von *Client C* erzeugt und zusammen mit dem Ticket $T_{c,s}$ an Principal S gesendet
- $A_C = \{ C, addr, timestamp \}_{K_{C,s}}$
- Principal S entschlüsselt Authenticator und prüft: IP-Adresse des Senders = *addr*, Gültigkeit der timestamp

Protokollschritte zusammengefasst

	Von	An	Nachricht
1	Client	KDC	Joe, TGS, $Nonce1$, $\{Joe, Time\}_{K_{Joe}}$
2	KDC	Client	$\{K_{Joe,TGS}, Nonce1\}_{K_{Joe}}$, $\{T_{Joe,TGS}\}_{K_{TGS}}$
3	Client	TGS	$\{A_{Joe}\}_{K_{Joe,TGS}}$, $\{T_{Joe,TGS}\}_{K_{TGS}}$, NFS, $Nonce2$
4	TGS	Client	$\{K_{Joe,NFS}, Nonce2\}_{K_{Joe,TGS}}$, $\{T_{Joe,NFS}\}_{K_{NFS}}$
5	Client	NFS	$\{A_{Joe}\}_{K_{Joe,NFS}}$, $\{T_{Joe,NFS}\}_{K_{NFS}}$

Protokollanalyse:

- Welche Schutzziele werden erreicht?
 - welche Konzepte tragen dazu bei,
 - nach welchen Protokollschritten sind die Ziele erreicht,
 - werden sie wirklich erreicht?
 - was Wissen die Beteiligten, was müssen sie verwalten, ...
- welche Rolle spielt der Authentikator, was passiert, wenn man ihn nicht verwendet?
- wo steckt in dem Protokoll das Challenge-Response Prinzip
- Welche Sicherheitsschwächen besitzt das Protokoll?
- welche Angriffsmöglichkeiten ergeben sich daraus?
- wie könnte man die Schwächen beheben?

Anmerkungen:

- Tickets werden automatisch von Client gecashed, erst beim Log-out werden die Tickets gelöscht
- Auffinden von zuständigen KDC-Servern:
 - z.B. durch manuelle Konfiguration idR in *krb5.conf*
 - *alternativ*: KDC-Dienste über DNS bekannt geben
- Gültigkeitsdauer eines Tickets:
 - In Version 4 max. 1280 Minuten (ca 21 Stunden)
 - In Version 5: Anfangs- und Endzeit angeben, flexibel
- Nach Ablauf der Ticket-Gültigkeit:
 - Benutzer muss *kinit*-Programm ausführen, mit Benutzername, Passwort: Initial-Ticket wird erzeugt
- Individuelles ‚Kerborisieren‘ von Services, erfordert Zugriff auf Sourcen, um Kerberos zu nutzen

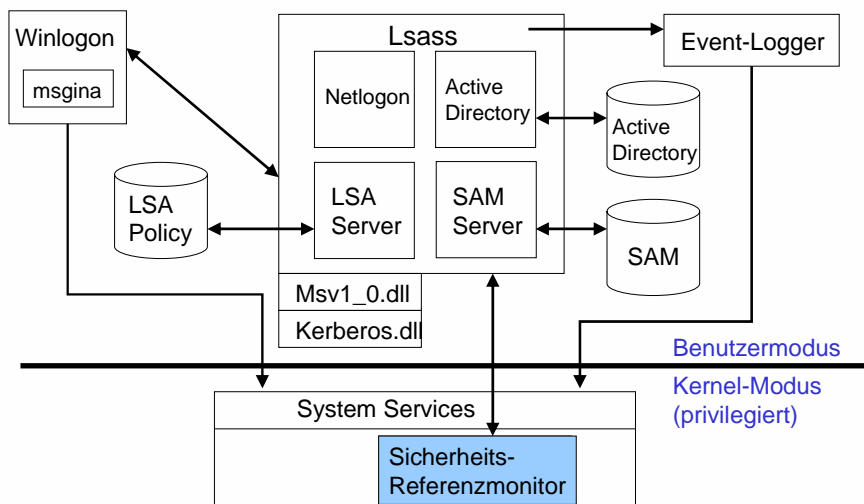
Inter-Realm Authentifikation unter Kerberos

- Version 4:
 - Basis: KDCs der Reals besitzen gemeinsamen geheimen Schlüssel (Pre-Shared secret)
 - Benutzer erfragt bei seinem Domänen TGS ein **Ticket für remote TGS**
 - Abläufe und Ticket-Ausstellung **wie gehabt**
 - **Problem**: bei n Realms:
 - $n(n-1)/2$ sichere Schlüsselvereinbarungen zwischen Realm-KDCs erforderlich
- Version 5: spezielle Ticket Flags, u.a. FORWARDED
 - Ausstellen eines TGS-Ticket mit anderer IP-Adresse
 - keine Vereinbarung von Pre-Shared Secrets

Fazit:

- Sicherheit von Kerberos: siehe Diskussion Folie 102
- Einsatz: weit verbreitet, idR mit **LDAP** Verzeichnisdienst
 - standardmässig in Windows 2000/XP integriert für domänenweites Login
- Aber Problem: Kerberos ist Authentifizierungs- aber **nicht Autorisierungssystem**
 - Zugriffskontrolle benötigt eindeutige Kennung des Principals, unterschiedlich für verschiedene Server!
 - ID- und Accessmanagement zZ noch hot-topic
- Andere Single-Sign-on Authentifizierungssysteme:
 - Microsoft Passport (vgl. Buch)
 - **Liberty Alliance Ansätze:**
Föderierte Identität, Circles of Trust (vgl. Übung)

8.5 Fallbeispiel: Authentifikation unter Windows 2000/XP



Windows-Sicherheits-Subsystem (siehe Folie 106): umfasst

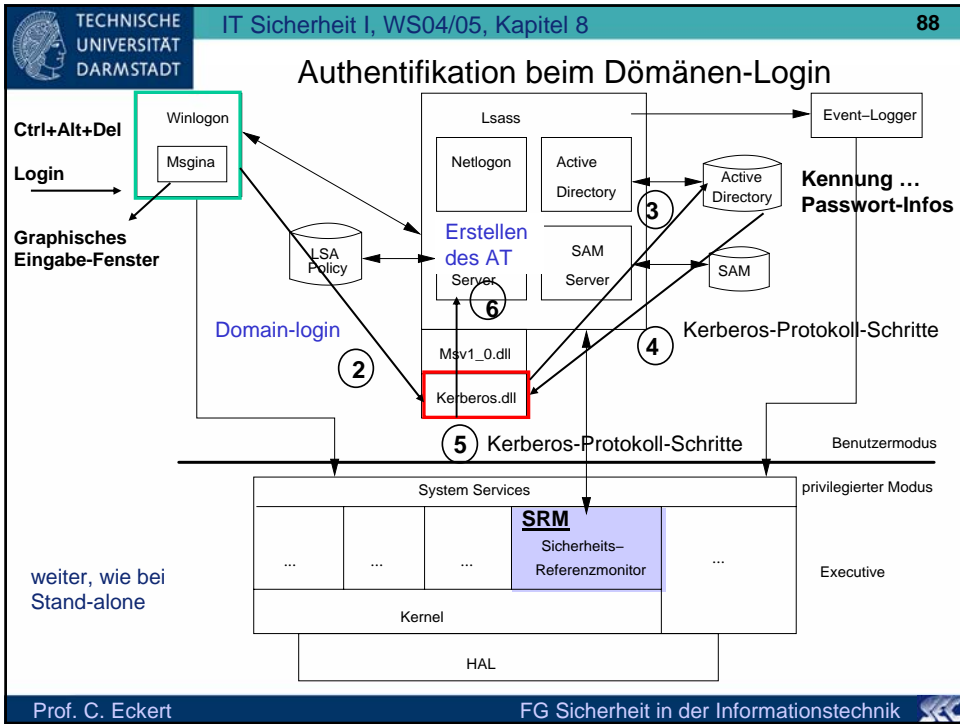
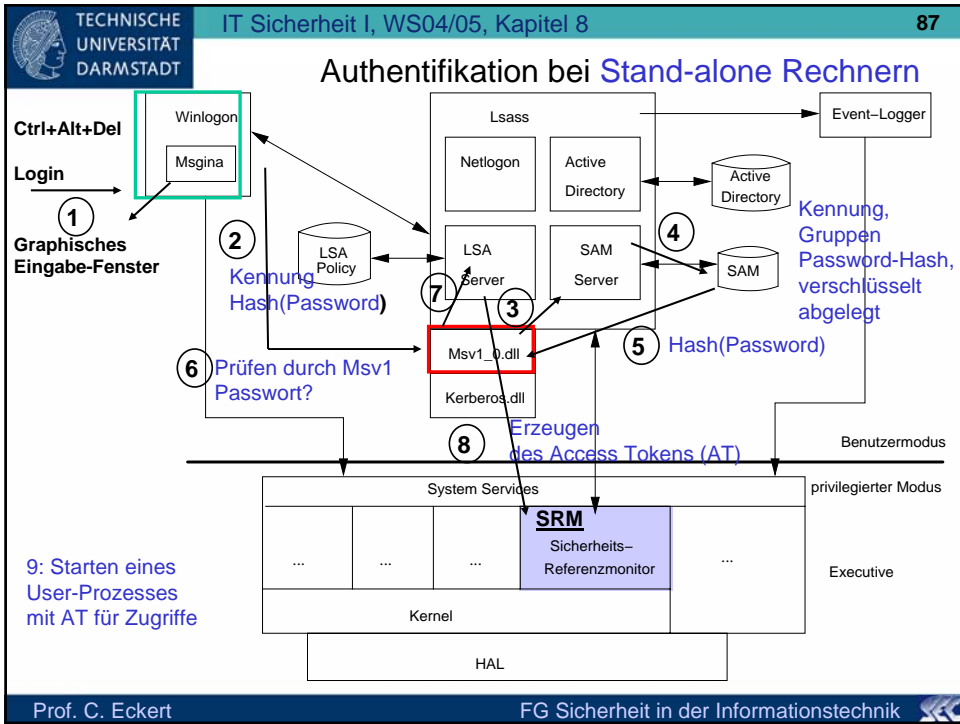
- Komponenten für die Authentifikation und
- Komponenten zur Durchführung der Zugriffskontrolle

Komponenten im Überblick:

- **Sicherheits-Referenzmonitor**: Bestandteil des Kerns
 - kontrolliert die Zugriffe auf Objekte,
- **Local Security Authority Subsystem (Lsas)**: user-mode Prozess
 - verwaltet die lokale Sicherheitsstrategien des Systems, mit
 - Festlegungen der zum **einzu-loggen berechtigten Benutzer** welche **Password-Strategie** zu verwenden ist, etc.
 - **Benutzer-Authentifizierung** (Ausstellen von Access Token)

- **Lsas Policy Datenbank**: Bestandteil der Registry enthält die verschiedene Strategien des Lsas
- **Security Accounts Manager (SAM)**: verwaltet u.a. lokale Benutzerkennungen, Gruppen, Passworte
- **Active Directory**: Verzeichnisdienst,
 - DC mit Informationen über Objekte in einer Domäne
 - Informationen über Benutzer u. Gruppen der Domäne
 - verwaltet Passworte sowie Berechtigungen
 - jede Domäne kann eigene Sicherheitsstrategie festlegen
- **Authentifizierungs-Paket**: enthält Dynamic Link Libraries (dll),
 - dlls: implementieren Authentifizierungsstrategie
 - Standard: **Kerberos** und **Msv1_0** (default für Stand-alone)

- **Msv1_0:**
 - Senden von Benutzername und gehashtes Passwort zum SAM
 - Überprüfung des gehashten Wertes mit dem im SAM gespeicherten Passwort
- **Kerberos** (proprietäre Variante):
- Protokoll wie beschrieben
 - Kerberos KDC ist Bestandteil des Active Directories
- **Winlogon: Logon** – Prozess
 - aktiv, wenn CTR-ALT-ENT Tasten gedrückt werden
 - Abwickeln des interaktiven Dialogs zum Login
- **Netlogon:** bearbeitet Anfragen von LAN Manager 2 (NT)
Ablauf wie bei lokaler Authentifikation



Kapitel 9 Access- und Trust-Management

- Konzepte und Verfahren zur Zugriffskontrolle (access),
Entwicklung: Nutzungskontrolle für Content (inkl. DRM)
- Trusted Computing, TPM

9.1 Accessmanagement

Rechte-Verwaltung, Kontrollen

9.1.1 Rechteverwaltung

- Zugriffskontrolllisten
- Capability-Listen
- Domain Type Enforcement
- Lock-Key-Konzept

Subjekte	Objekte		
	Datei 1	Datei 2	
Bill	owner, r,w	w	
Joe	r,x		
Alice		owner, r,x	

↑
Objekt-Sicht, Zugriffskontrollliste

9.1.1.1 Zugriffskontrollliste: Access control List (ACL)

- Spaltenweise Realisierung der Matrix
- ACLs: am häufigsten eingesetztes Konzept in klassischen BS
u.a. in UNIX/Linux, Windows 2000, XP
- **Beispiel:** ACL (Datei1) = ((Bill, {owner, r, w}), (Joe, {r,x}))
- Eine ACL ist eine geschützte Datenstruktur des
Betriebssystems, **wodurch/wie geschützt?** Mit ACL? Henne-Ei?
- **Vorteile von ACLs:**
 - vergebene Rechte sind effizient für Objekte bestimmbar
 - Rechterücknahme ist meist effizient realisierbar
 - dezentrale Kontrolle möglich, denn über ACL sind Rechte
idR direkt mit Objekt verknüpft, **aber** reicht das für eine
vertrauenswürdige dezentrale Zugriffskontrolle wirklich?

• **Nachteile von ACLs:**

- Bestimmen der **Subjekt-Rechte** i.d.R. **sehr aufwändig**
- Aufwändige ACL-Kontrollen bei jedem Zugriff
- **schlechte Skalierbarkeit** bei sehr dynamisch wechselnder Menge von Subjekten, falls ACL für **einzelne Subjekte/Nutzer** vergeben werden

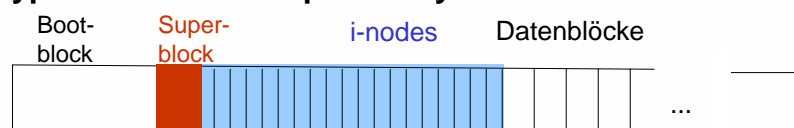
• **Beispiel:**

- Vergabe von Zugriffsrechten bei Web-Diensten oder Web-Portalen mit
- beliebig dynamischer Anzahl von Nutzern,
- was wäre hier ein **geeignetes Konzept** für die Granularität der Rechtevergabe?

9.1.1.2 Beispiel: Rechte und ACLs unter UNIX/Linux

- Zu schützende Objekte: Dateien, Verzeichnisse
- Dateien/Verzeichnisse werden BS-intern über einen Datei-Deskriptor, die **i-node** (index node), beschrieben
 - die i-node enthält u.a. Verweise auf die Adressen der Plattenblöcke mit Dateiinhalten
 - Name des Besitzers der Datei u. ACL (siehe Folie 10)
- die i-nodes werden auf der Festplatte verwaltet,
- beim Öffnen der Datei (open-call) wird inode eingelagert

Typisches Unix-Festplattenlayout:



Identifikation von Subjekten unter Unix:

- Rechtevergabe an **Datei-Besitzer** und an **Gruppen**
- diese müssen im System eindeutig identifizierbar sein.
- **Eindeutige Identifikation**: UserID, GroupID, jeweils 32-Bit Zahl
- jeder Prozess erbt diese Attribute von seinem Vater
- **Bem.:**
 - uid zw. 0 und 99 typischerweise für **Systemprozesse**
 - **uid=0 ist Superuser**, egal welche Kennung,
 - Zusammenhang: Kennung und uid in: /etc/passwd
 - Zusammenhang: Gruppen und guid in /etc/group
 - idR spezielle Gruppe **wheel** : *alle Systemadmin*
meist: su-Kommando nur für *wheel*-Mitglieder zulässig

- Unix unterscheidet **effektive** und **reale** uid und guid, Zugriffsberechtigungen hängen von **effektiver** Identität ab,
- die effektive Identität kann sich dynamisch ändern, meist aber effektive = reale uid
- Temporäre Rechte-Vergabe über **setuid**-Konzept/Recht (s.u.)

Zugriffs-Rechte: **Standard-Rechte** r, w, x: Bedeutung klar

Verzeichnisrechte: spezifische Interpretation der Rechte

- **x Suche-Recht**: Verzeichnis darf als Teil eines Pfadnames durchlaufen, die Dateien dürfen geöffnet werden
- **r Lese-Recht**: Auflisten der Dateien des Verzeichnisses
- **w Schreib-Recht**: Hinzufügen oder Entfernen von Dateinamen

Spezielle Rechte:

sticky-Bit:

- letzter x-Eintrag der ACL wird in ein t umgewandelt,
- für ein Verzeichnis bedeutet das gesetzte Bit:
nur Datei-Owner darf Datei löschen, [wo sinnvoll?](#)

suid-Bit: Ziel: temporäre Weitergabe (Delegation) von Rechten

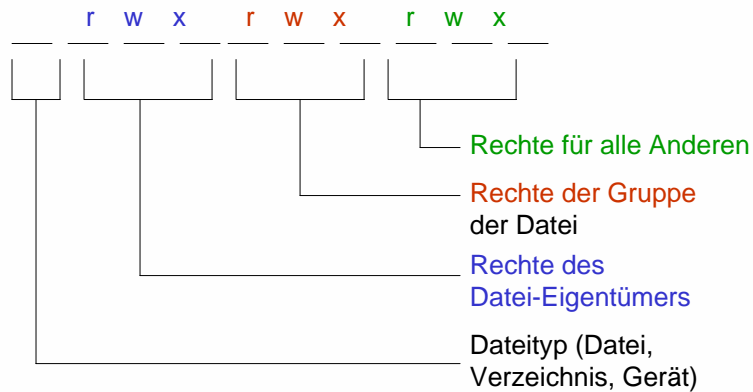
- gesetztes suid-Bit: angezeigt durch **s bei Owner-Rechten**
- durch chmod- Kommando kann Bit gesetzt werden
- Ausführung einer Datei/Programm mit gesetztem suid-Bit:
[effektive uid_caller := uid_program_owner](#),
- **Beispiel:** /bin/passwd-Kommando: Zugriff auf /etc/passwd
-r-sr-xr-x root /bin/passwd und -rw- r-- r- root /etc/passwd
- **Sicherheitsprobleme mit suid-Bit? Beispiel?**

Rechte des Superusers: uid=0:

- **Prozess-Kontrolle:** u.a.
Ändern der uids, Ein-, Ausschalten von Audits,
Senden von Signalen, Verändern von Prios etc.
- **Geräte-Kontrolle:** u.a.
shut-down, reboot, Lese-Schreibzugriffe auf beliebige
Speicherbereiche, verändern von Zeiten
- **Netzwerkkontrolle:** u.a.
Promiscuous-Modus einschalten: alle Pakete beobachtbar,
Ausführen von Diensten an Trusted Ports, Beispiele?
- **Dateisystemkontrolle:** u.a. r,w, löschen beliebiger Dateien,
Mounten, unmounten, Ausführung beliebiger Programme,
Kennungen ändern

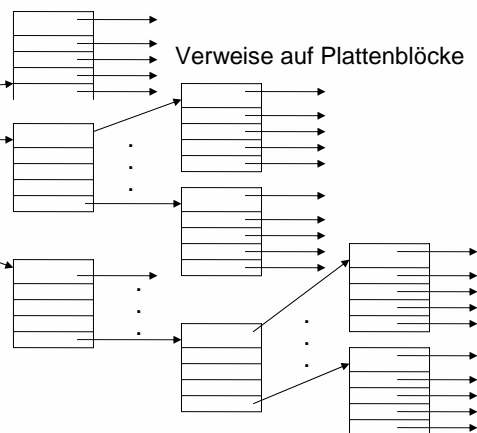
Unix-ACL: stark vereinfachte ACL

- Standard-Unix: Rechtevergabe nur an:
Eigentümer (owner der Datei), Gruppe, Rest der Welt
- sehr grob, keine Vergabe an einzelne Benutzer



owner joe, uid
group student, guid
type regular file
perms rwxr-xr-x
accessed Feb 12 1999 3:00 P.M.
modified Feb 11 1999 10:16 A.M.
Adressen der 10 ersten Plattenblöcke
einfach indirekt
zweifach indirekt
dreifach indirekt
Unix-i-node

ACL als Bestandteil der i-node unter Unix

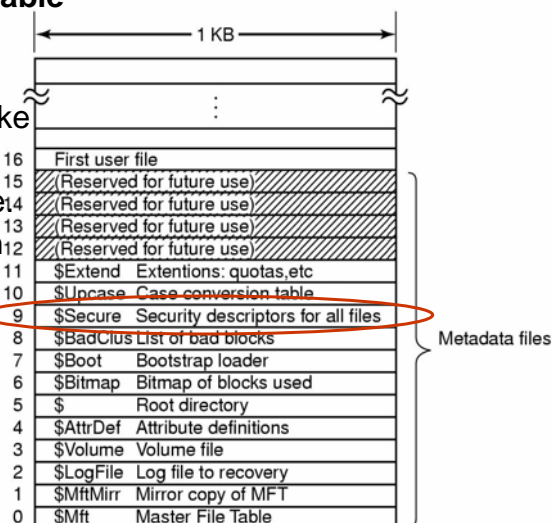


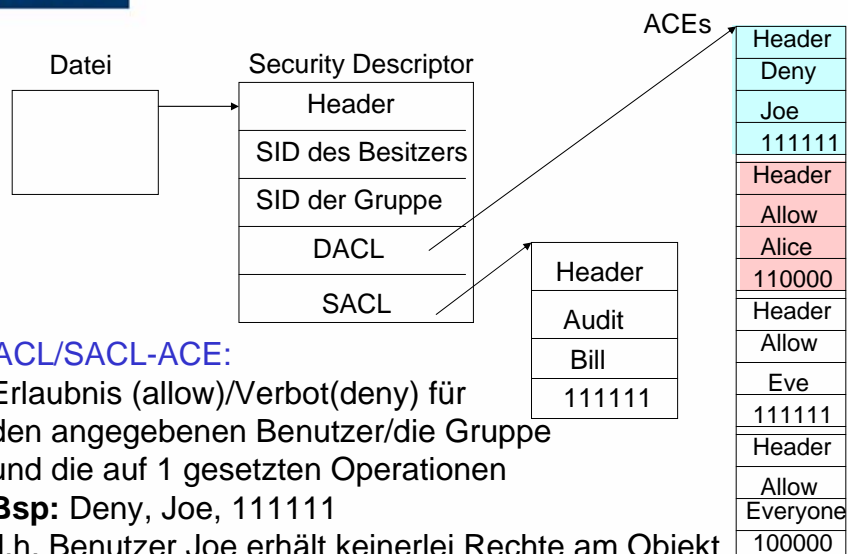
9.1.1.3 Beispiel ACL-Konzept unter Windows 2000/XP

- differenzierte ACL mit **Berechtigungen** bzw. **Verboten** für **einzelne Benutzer**, aber auch für Gruppen
- Berechtigungen/Verbote in separater Datenstruktur, dem **Security Descriptor**, gespeichert
- Security Descriptoren aber nicht in Datei-Deskriptor sondern
- in globaler Tabelle, der **Master File Table (MFT)** verwaltet
- **Security Descriptor** enthält: (siehe Folie 13)
 - **SID** = systemweit eindeutige Security ID des Besitzers des Objekts und der Gruppe
 - **DACL** = Discretionary ACL: Liste von ACEs
 - **ACE** = Access Control Element mit allow/deny
 - **SACL** = System ACL, spezifiziert die Operationen, deren Nutzungen zu protokollieren sind

NTFS: Master File Table

- **Eintrag 0:**
Adressen der MFT-Blöcke
- **Eintrag 2:** Log-Datei,
Z.B. neue Verzeichnisse,
Strukturelle Änderungen
- **Eintrag 3:** Infos über
das Volume: Größe etc.
- **Eintrag 9:** **Security
Descriptors**





DACL/SACL-ACE:

- Erlaubnis (allow)/Verbot(deny) für
- den angegebenen Benutzer/die Gruppe
- und die auf 1 gesetzten Operationen
- **Bsp:** Deny, Joe, 111111
d.h. Benutzer Joe erhält keinerlei Rechte am Objekt
- **Bsp SACL:** alle Aktionen von Bill sind zu protokollieren

9.1.2 Capability-Konzept

Zeilenweise Realisierung der Matrix

- Capability, **Zugriffsticket** mit Objekt-UID und Rechtebits
- Capability-Besitz berechtigt zur Wahrnehmung der Rechte
- Für jedes Subjekt s eine Capability-Liste (Clist)
- **Beispiel** Clist (Joe) = ((Datei1, {r,x}), ((Datei2, {w}))
- Beispiele von Systemen mit Capability-Konzept:
IBM System/38, Mach- μ -Kern (Ports), Amoeba,
POSIX Capabilities (z.B. von Linux ab 2.2 unterstützt)

Vorteile von Capabilities:

- Einfache Bestimmung der **Subjekt – Rechte**
- **Einfache Zugriffskontrolle:** nur noch Ticketkontrolle!

Nachteile:

- **Rechterücknahme** schwierig, Kopien suchen! (**wieso Kopien?**)
- **keine Subjekt-Ticket Kopplung**, Besitz berechtigt auto-automatisch zur Wahrnehmung der Rechte, **wieso Problem?**
- **Objekt -Sicht auf Rechte schwierig**: wer darf was!
- in der einfachen Ausprägung **ungeeignet für verteilte Umgebungen**, **warum? Ticketkonzept erscheint prädestiniert?!**

Heute: **Mischformen**: ACL **und** Capabilities in Standard-BS

Beispiel UNIX/Linux (Windows analog)

- Vor Datei-Nutzung: Datei muss geöffnet werden (open)
- Kern prüft anhand der ACL die Berechtigung und
- stellt **file-handle** (entspricht Capability) für Zugreifer aus

9.1.3 DTE Domain and Type Enforcement

- vereinfachtes Capability und Rollen-Konzept, spezielle Mandatory Access Policies

Konzepte:

- jedem Subjekt wird genau eine **Domäne** als Attribut zugeordnet, z.B. *domain project*
- jedem zu schützenden Objekt wird ein **Typ** zugeordnet, z.B. *type budget*
- **Zugriffsrechte** werden für Typen an Domänen vergeben,
- das Domänen-Attribut ist eine Art Capability mit den Rechten des Subjekts in der Domäne (vereinfachte Rolle)
- Spezielle Dateien bilden die Entry-Points zu Domänen

- durch die Ausführung eines solchen Entry-Points:
 - Subjekt kann (falls exec Recht für Entry-Point in der Domäne) die Domäne wechseln

Domäne: wird beschrieben durch

- Entry-Point Prozedur, um in die Domäne zu wechseln
- Rechte an verschiedenen Typen

Bem.:

- Zu jedem Zeitpunkt kann ein Subjekt sich nur in einer Domäne aufhalten und diese Rechte nutzen
- das Domänen-Attribut kann als Bestandteil der Prozess-Beschreibung implementiert werden, das Typ-Attribut als Bestandteil der Objekt-Beschreibung (z.B. inode)

9.1.4 Lock/Key-Konzept

- Jedes Subjekt s besitzt **Key-Liste**: $(\dots, (o, K), \dots)$,
z.B. implementiert als kryptographische Schlüssel
- Jedes Objekt o besitzt **Key-Liste**: $(\dots, (L, \alpha), \dots)$
 L ist das „Schloss“, z.B. verschlüsseltes Objekt
 α ist eine Menge von Zugriffsrechten
- Zugriffsversuch: s auf o mit Rechten β
 - (1) s sucht (o, K) aus seiner Key – Liste
 - (2) Prüfung: $\exists L : K = L?$ wenn ja \Rightarrow Schlüssel passt!
für (L, α) muss gelten: $\beta \subseteq \alpha$ d.h. Rechte sind zulässig
- Rechterücknahme ist einfach:
Verändern des Schlosses L in Objekt – Liste
 \Rightarrow Schlüssel „passen“ nicht mehr! **Beispiele bekannt?**

9.2 Zugriffskontrolle

- „klassischer“ Ansatz: **Referenzmonitor**, d.h. jeder Zugriff wird durch eine zentrale Komponente kontrolliert
- **Kontrolle:**
 - Bestandteil der **Trusted Computing Base** des BS – Kerns
z.B. Dateisystem: Kontrolle bei Dateizugriffen
 - Bem. Trusted Computing Base und Trusted Computing Plattform sind etwas Unterschiedliches

9.2.1 Allgemeines Prinzip

- idR Aufteilung in **Berechtigungs-** und **Zulässigkeitskontrolle**
- ermöglicht eine Verteilung der Aufgaben ohne die Berechtigungsinformation zu replizieren
- woher ist **Prinzip bereits bekannt?**

Berechtigungskontrolle:

- beim erstmaligem (oder ggf. gemäß Policy, wenn das gefordert wird) Zugriff auf ein Objekt
- von vertrauenswürdigen Systemdiensten (z.B. Dateisystem)
- bei einer erfolgreichen Überprüfung: Ausstellung einer Berechtigungsbescheinigung, z.B. File- oder Object-Handle

Zulässigkeitskontrolle

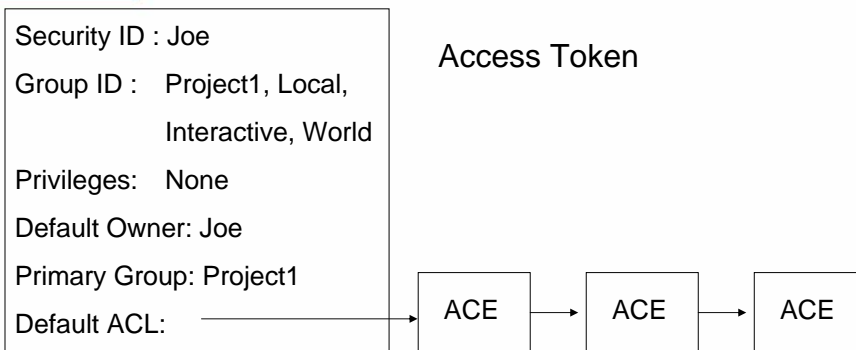
- durch Objektverwalter (z.B. user-level Server)
- bei Objekt-Zugriff: Prüfen der Gültigkeit der Bescheinigung
- kein Zugriff auf die Rechteinformation notwendig
- ggf auch negative Konsequenzen?

Prinzipielles Problem: wer kontrolliert die Kontrolleure? Trust?!

9.2.3 Beispiel: Zugriffskontrolle unter Windows 2000/XP

- Benutzer erhält beim Login eine Datenstruktur, das **Access Token** (AT) vom BS ausgestellt (siehe Kap. 8)
- Alle Prozesse und Threads des Benutzers erhalten Kopie des AT (Vorsicht, AT nicht mit Capability verwechseln!)
- **Aufbau eines AT:**
 - Subjektidentifikation SID und Gruppen-ID
 - Ggf spezielle Privilegien wie das Shut-down Recht
 - Default ACL: wird bei der Erzeugung eines Objekts verwendet, um Default-Rechte zu vergeben

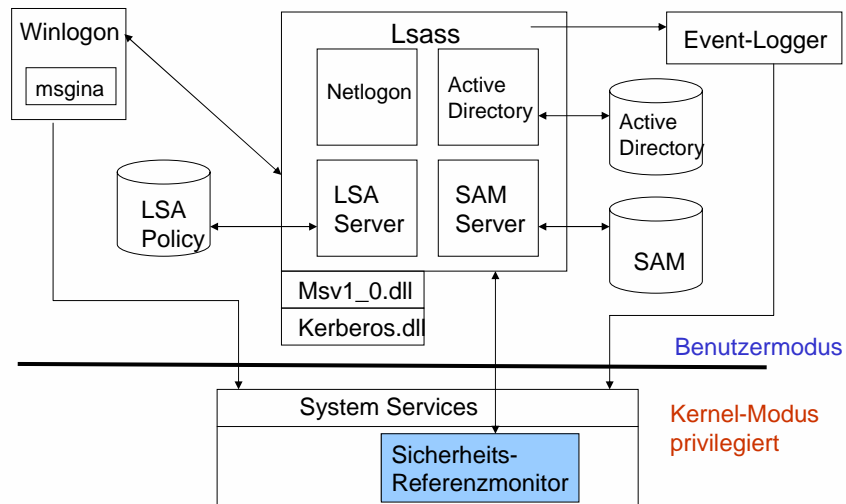
Header	Expiration time	Groups	Default CACL	User SID	Group SID	Restricted SIDs	Privileges
--------	-----------------	--------	--------------	----------	-----------	-----------------	------------



Nach Prozess Start:

- nur noch Rechtevergabe, -rücknahme, keine anderen Veränderungen am AT
- **Impersonation:** Konzept zur Weitergabe von ATs, z.B. Delegation von Nutzeridentität an Server

Kontrolle durch Referenzmonitor unter Windows 2000/XP



(vgl. Folien Kapitel 8)

Zugriffskontrolle:

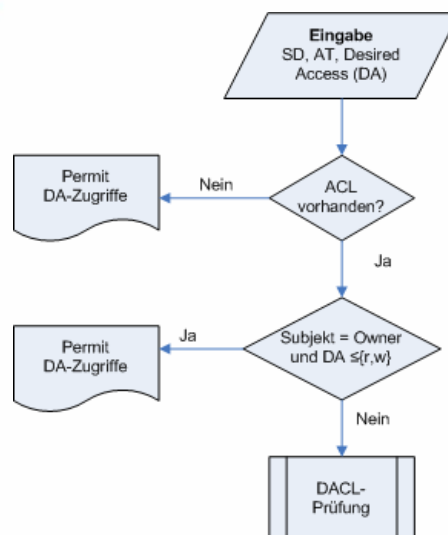
- Überprüfung der Zugriffsberechtigung eines Prozesses/Threads S beim Zugriff auf Objekt O
- Aufruf der **SeAccessCheck** bzw. **AccessCheck Operation** des **Sicherheitsreferenzmonitors**
- **Eingabe:**
 - Security Descriptor des Zielobjekts O,
 - Access Token des zugreifenden Prozesses S mit (SID, Gruppen SIDs (ggf. deaktiviert), Privilegien)
 - gewünschte Zugriffsoperationen, Desired-Access-Maske
- **Ausgabe:**
 - erlaubt, verboten

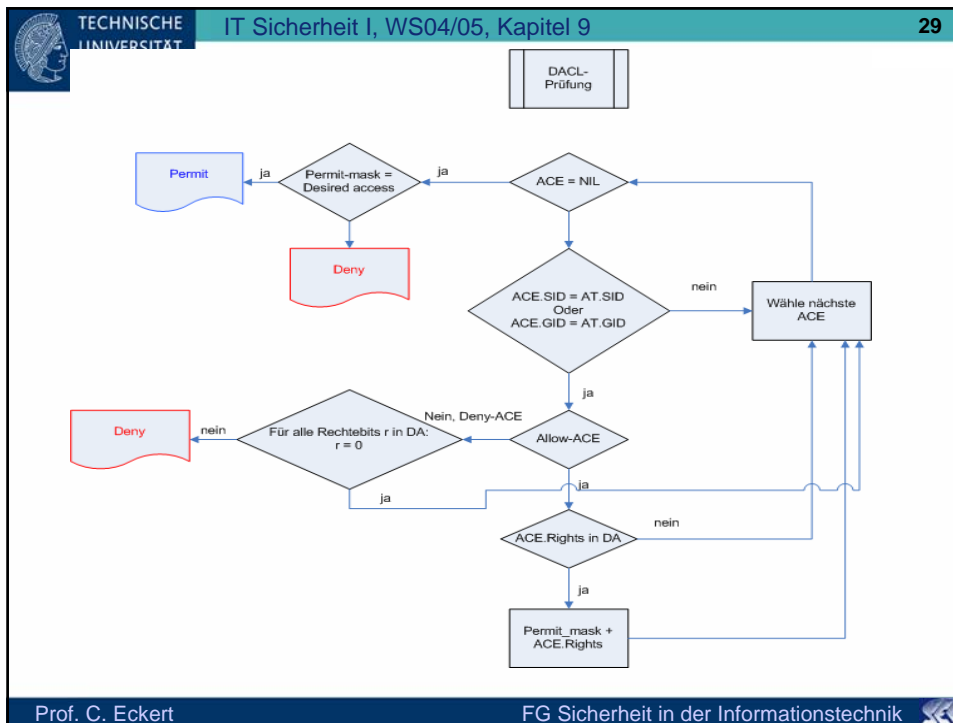
Algorithmus: etwas vereinfachter Ablauf zur Kontrolle

- Falls keine ACL festgelegt, dann ist Zugriff erlaubt
- Falls Subjekt der Objekt-Owner ist, dann besitzt es automatisch read und write-DACL Rechte, keine DACL-Prüfung, falls keine weiteren Rechte angefordert, **sonst: DACL-Prüfung**

DACL-Prüfung: ACEs werden nach **FIFO** durchlaufen:

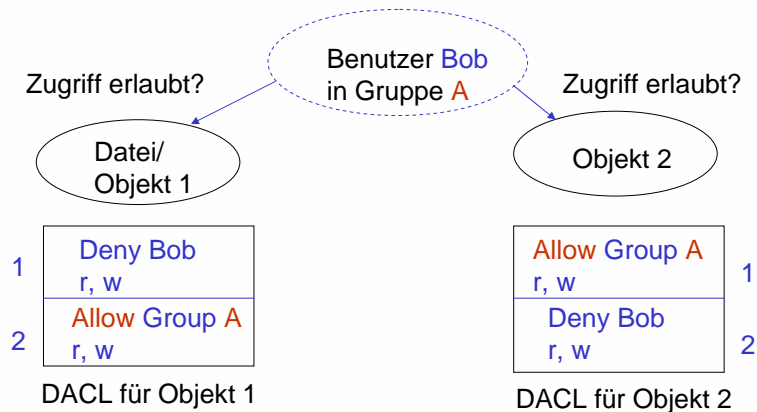
- falls die **SID in der ACE** mit der SID oder einer aktivierten Gruppen-SID im AT übereinstimmt, dann
 - falls es eine **Allow-ACE** ist : Rechte der ACE werden gewährt, falls diese angefordert wurden,
 - falls damit **alle angeforderten Rechte gewährt:** Algorithmus terminiert mit **Access allowed**





- TECHNISCHE UNIVERSITÄT DARMSTADT IT Sicherheit I, WS04/05, Kapitel 9 30
- falls es eine **Deny-ACE** ist :
 - Zugriffsverbot, falls in ACE mindestens eines der angeforderten Rechte verboten ist,
 - Algorithmus terminiert mit **Access denied**
 - Falls DACL vollständig durchlaufen und
 - **nicht** alle geforderten Rechte **gewährt** wurden:
 - Algorithmus terminiert mit **Access denied**
- Reihenfolge** der ACEs ist wichtig, **warum?!**
- Bemerkungen:**
- Ältere Win32 Funktionen wie *AddAccessAllowedAcc*:
 - neue ACE-Einträge wurden **am Ende der DACL** eingefügt,
 - DACL musste manuell konfiguriert werden
- Prof. C. Eckert FG Sicherheit in der Informationstechnik

Beispiel: Zugriff auf Objekt 1 bzw. auf
Objekt 2 durch Benutzer B in Gruppe A,
Welche Zugriffe sind erlaubt?



Bemerkung (cont.)

- Neuere Windows2000 Funktionen wie *SetSecurityInfo*, ordnen ACEs in der **Deny-vor-Allow Reihenfolge**
- Objekt-Owner erhält immer das w-Recht auf die DACL, **wann/warum ist das nützlich, oder doch eher problematisch?**
- Alle W2K/XP **Betriebssystemroutinen** werden im **privilegierten Modus** ausgeführt: d.h.
 - sie nutzen für Zugriffe **keine Handles**, sondern nur
 - Zeiger für **direkte Objektzugriffe**
 - Konsequenz: **keinerlei Berechtigungsprüfung** für BS-Routinen! (Erinnerung: auch Treibersoftware!)
 - Windows2000/XP Kern wird als **vertrauenswürdig** (trusted) betrachtet

Übergang von zentralen auf auf verteilte Dateisysteme

Übliches Szenario:

- Client-Server-artige Architektur: spezielle **Datei-Server**
- Datei-Server verwaltet Dateien,
- Client-Rechner greifen (über das Netz) darauf zu

Annahme:

- Bekannte Konzepte aus zentralen Systemen werden 1:1 übertragen (vgl. frühere Versionen von z.B. NFS)
- d.h. ACLs für Dateien auf dem Server verwaltet
- Datei-Server führt Berechtigungsprüfung durch, stellt ein File-Handle für nachfolgende Zugriffe aus

Welche Probleme treten in dem Szenario auf? Lösungen?

9.3 Verschlüsselnde Dateisysteme

Situation: große Menge sicherheitsrelevanter Information wird in Dateisystem gespeichert und verwaltet

Angriffe:

- **Stehlen** von Geräten (z.B. PDAs)
- **Vergessen**, unbeaufsichtigtes Liegenlassen von Geräten:
- **Unbeschränkter, (physischer) Zugriff** auf (mobilen) Rechner: Viren, Würmer, Trojaner, Bluetooth-Attacken etc.
- **Booten** von anderem Speichermedium:
 - Umgehen der built-in Sicherheitsdienste des BS

Konsequenz: Traditionelle Zugriffskontrolle reicht nicht aus

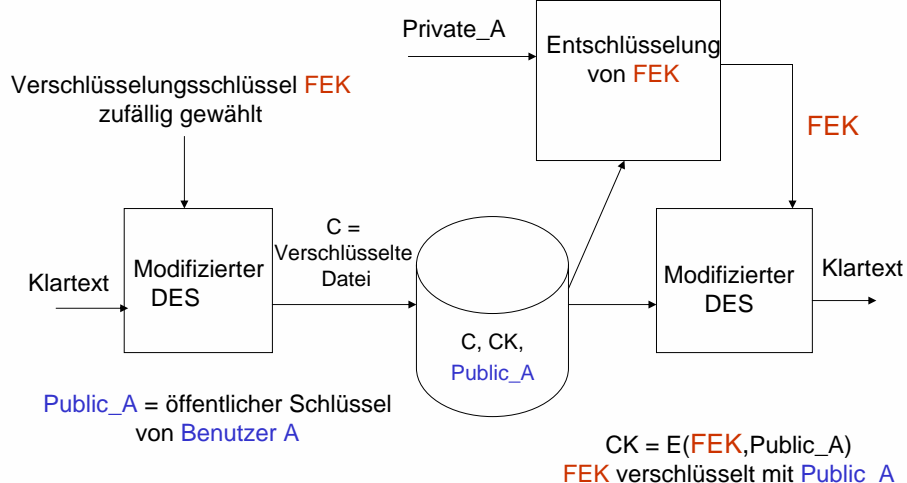
Lösung: **Verschlüsselnde Dateisysteme**, am Beispiel EFS

9.3.1 Microsoft Encrypting File System (EFS), W2K/XP

Prinzipielles:

- Nutzung eines hybriden Verschlüsselungs-Konzepts
- Verschlüsselung einer Datei mit **symmetrischem** Datei-Schlüssel **FEK**
- **FEK**: wird mit öffentlichem Schlüssel **Public_A** des Datei-Owners **asymmetrisch** verschlüsselt
- für Dateizugriff ist privater Benutzerschlüssel **Private_A** notwendig
- privater Benutzerschlüssel **Private_A** wird verschlüsselt auf Festplatte verwaltet, **Alternativer Aufbewahrungsort?**
- Verschlüsselung von **Private_A** mit einem Benutzer-**Passwort**

Ver- und Entschlüsseln einer Datei

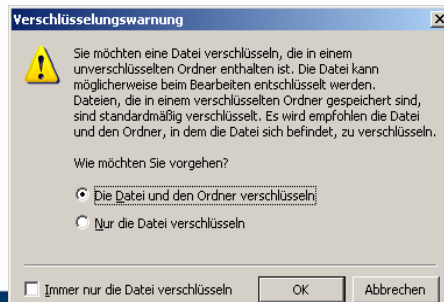
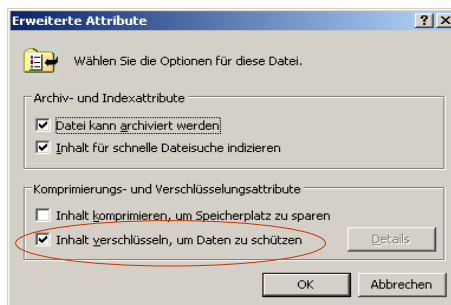
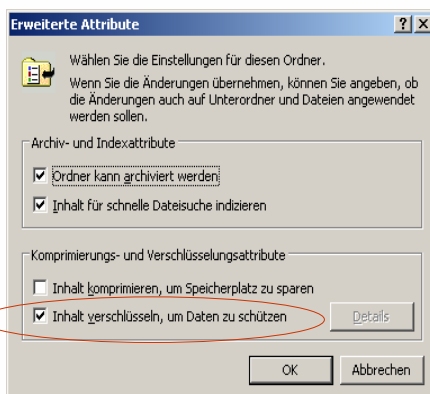


beim Login des Benutzers:

- Entschlüsselung des Schlüssels **Private_A** und Speicherung im virtuellen Adressraum des Benutzerprozesses
- Löschen des Schlüssels aus Adressraum **beim Logout**
- Auswahl von Dateien/Ordner **zur Verschlüsselung** über **Eigenschaften, Erweiterte Attribute**
- Automatische Datei/Ordner-**Entschlüsselung** beim Öffnen,
 - entweder durch Benutzer oder
 - mit Recovery Schlüssel des Admins
- Automatische Verschlüsselung beim Close:
Nutzen des bereits erzeugten **FEK** dafür

Verschlüsseln einer einzelnen Datei

Verschlüsseln eines Ordners



Verschlüsselung: einige Probleme mit EFS

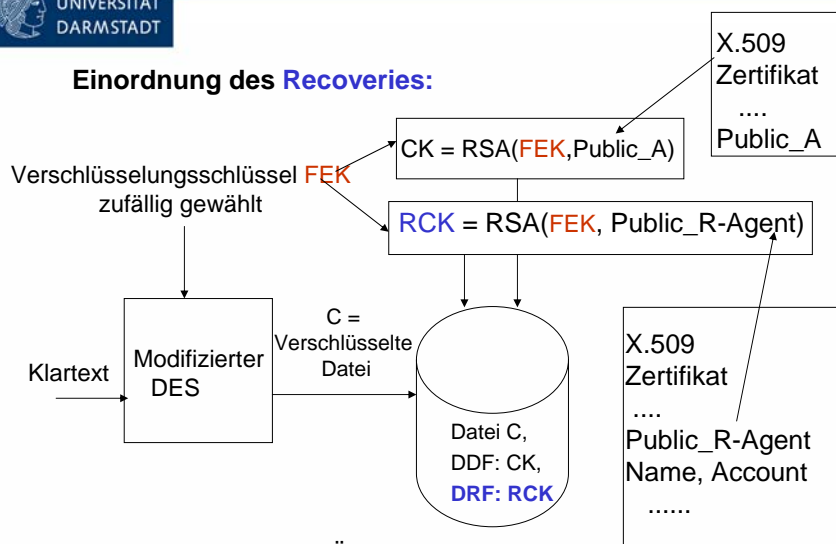
- nur NTFS-Dateisysteme, keine FAT16/32
- Kopien von EFS-Dateien auf Diskette sind **unverschlüsselt**
- **kein verschlüsselter Transfer** von EFS-Dateien über das Netz
- WS2000: **Passwortänderung durch Angreifer:**
Zugriff auf Dateien z.B. mit Startdisk und Programm chntpw
- MS-Empfehlung: **syskey Dienst** starten im
 - Modus 2 (zusätzliches Passwort beim Systemstart) oder
 - Modus 3 (Schlüssel auf Diskette gespeichert, für Start notwendig)
- Bem.: **syskey.exe** nur mit Admin-Rechten ausführbar
 - erzeugt **Random-Systemschlüssel**,
 - Modus 1: Systemschlüssel wird auf Festplatte abgelegt

- Umgang mit **temporären Backup-Kopien:**
 - Anlegen beim Öffnen (entschlüsselte EFS-Datei)
 - Löschen beim Schließen der Datei, aber
 - Blöcke werden nicht wirklich freigegeben
 - **Lösung:** Cipher.exe: Löschen von Dateien auf Festplatte
- Windows 2000:
Änderung des Benutzer-Passwortes durch Admin:
 - rechtmäßiger Benutzer kann Dateien nicht mehr entschlüsseln
 - **Lösung:** privaten Schlüssel u. Zertifikat auf externes Medium (z.B. Diskette) exportieren, nach Import sind Dateien wieder nutzbar
 - **Alternative:** **Recovery-Feature** nutzen

Schlüsselrecovery-Feature von EFS

- Nutzung von EFS erfordert mindestens einen **Recovery Agent**
- default: **Administrator** ist Recovery-Agent
- von jedem Datei-Schlüssel wird zusätzlich eine Kopie mit einem **Recovery-Schlüssel** (Public-Key) verschlüsselt
- zu Recovery-Zwecken verschlüsselter Datei-Schlüssel: **gespeichert in EFS-Dateiattribut DRF** (Data Recovery Field)
- Schlüssel werden durch **Recovery-Agents** ausgestellt,
- öffentlicher Recovery-Schlüssel: im X.509 Agent-Zertifikat enhanced Key-usage Feld verweist auf Einsatz: Recovery
- **privater Recovery-Schlüssel** sollte ausserhalb des Rechners sicher aufbewahrt werden

Einordnung des Recoveries:



Bem: bei jedem Datei-Öffnen:

Prüfen der Gültigkeit der Zertifikate (Benutzer und Recovery-Agent)

Probleme mit der EFS- Recovery-Funktionalität?

- Schwachstelle: **nachlässiger Administrator**

Szenario: **Stand-alone Rechner**

- Angreifer hat physischen Zugriff
- kann das Passwort der Administrator-Kennung ändern
- Einloggen als Administrator,
- mit dem Administrator-Recovery-Schlüssel:
Entschlüsseln beliebiger Dateien

Frage: was sind die Nachlässigkeiten? Abwehr?
was ist mit Domänen-Rechnern?

Fazit: Nutzen von EFS sinnvoll?

- **In Domäne? Für Notebooks, mobile Geräte etc?**
- Alternativen?

9.4 Digital Rights-Management (DRM)

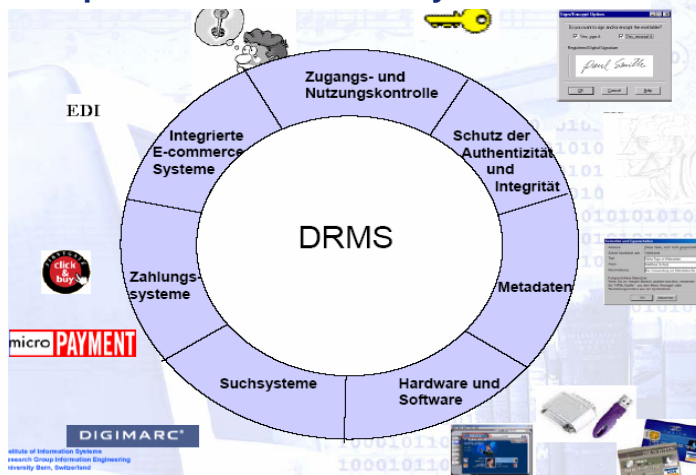
9.4.1 Definition: DRMS

Wir verstehen unter DRM-Systemen: (nach N. Rump, Uni Bern)

- elektronische **Vertriebssysteme für digitale Inhalte**, die die
- sichere **Verbreitung und Verwertung** digitaler Inhalte, z.B. über das Internet, Datenträger (CDs, DVDs, etc.), mobile Abspielgeräte oder Mobiltelefone ermöglichen.
- Sie ermöglichen eine effiziente **Rechteverwaltung** und eröffnen
- **neue Geschäftsmodelle** (z.B. kostenpflichtiger Download, Abo, Pay-Per-View)
- DRMS setzen verschiedene **Schutzmechanismen** ein, u.a. Verschlüsselung, Kopierschutz oder digitale Wasserzeichen.

- Schwächste Form: verhindern/erschweren des **Zugangs** zu digitalem Inhalt
- Stärkste Form: DRMS erlauben die **individuelle Abrechnung** der Nutzung.

Komponenten eines DRM-Systems



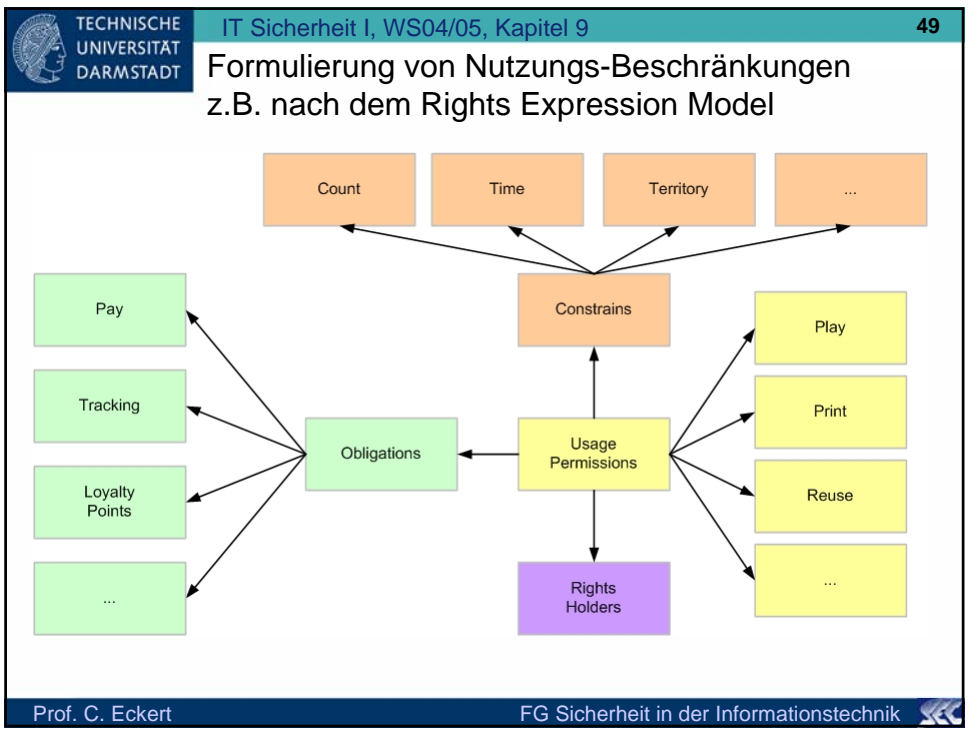
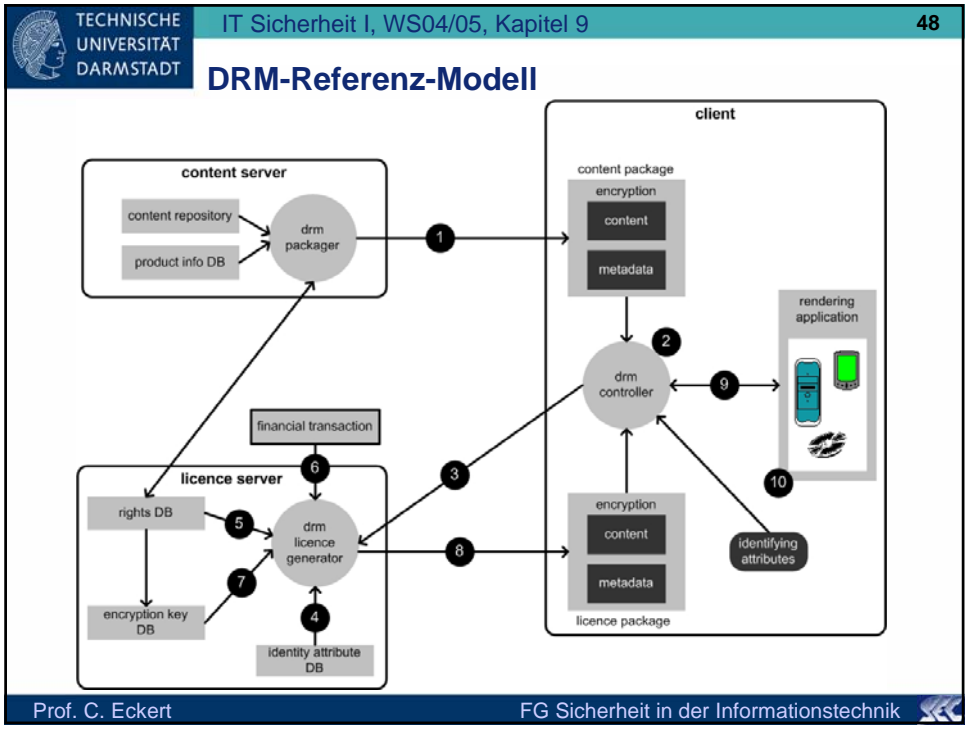
9.4.2 DRM Referenz-Modell (nach Rosenblatt)

- drei Bereiche: (siehe Folie 48)
 - Server, der die Inhalte speichert (*Content Server*),
 - der *Lizenz Server* (*License Server*) und
 - der *Kunde* (*Client*).

Einsatz:

- (1) Der Nutzer (*Client*) **lädt** (z.B. von einer Web-Site) die verschlüsselten digitalen Inhalte (*Content Package*).
- (2) Starten (Doppelclick) einer **automatischen Anfrage**,
 - Aktivierung des *DRM Controllers*.
 - Controller sucht die nötigen Informationen, um die gewünschte **Lizenz ausstellen** zu können.

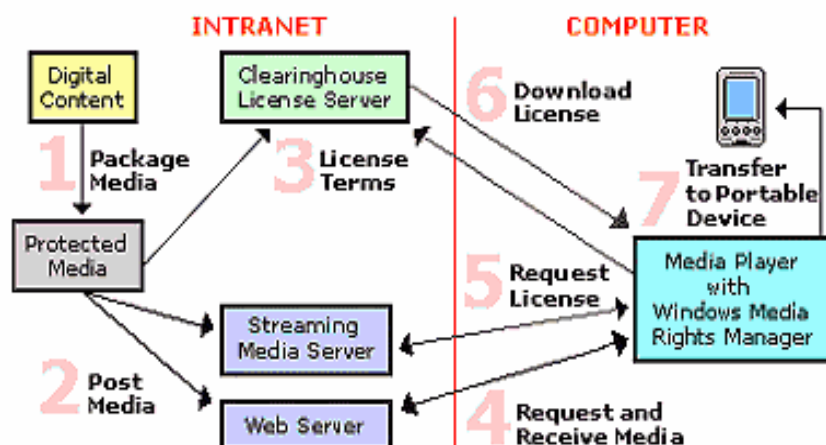
- (3) Controller sendet die **Identität** (*Identity*) des Nutzers und des digitalen Inhaltes zum Lizenz Server (*License Server*).
- (4) Der Lizenz Server identifiziert den Nutzer mit Hilfe der Identifications Datenbank (*Identities*).
- (5) Der Lizenz Server erfasst die **Rechteinformationen** gemäß der Anfrage des Nutzers (*Rights*).
- (6) Falls eine **Zahlung** notwendig ist, wird eine finanzielle Transaktion gestartet (*Financial Transaction*).
- (7) Erstellung der **verschlüsselten Lizenz** (*Encryption Keys*), die die Rechte (*Rights*), die Nutzeridentität (*Identity*) enthält
- (8) Die Lizenz wird dem Nutzer geschickt.
- (9) Entschlüsselung des digitalen Inhaltes und "**Freigabe**" des Inhaltes an das Wiedergabegerät (*Rendering Applications*).
- (10) **Wiedergabegerät** spielt, zeigt, etc. den digitalen Inhalt



9.4.3 Beispiel: *Windows Media Rights Manager*

- Dateiformat "Windows Media File" (wm, wma, wmv) vom MS
- **geschütztes** und **verschlüsseltes** Datenformat
- Nutzung einer solchen Datei erfordert einen Schlüssel
- Eigentümer digitaler Inhalte können diese mit dem "Windows Media Rights Manager" schützen und die zugehörigen **Lizenzen definieren** und verwalten.
- Der **Windows Media Rights Manager** wandelt digitale Inhalte in ein geschütztes und verschlüsseltes Dateiformat um.
- Digitale Inhalte wie Videos oder Musikstücke werden in "digital media Dateien" umgewandelt, die auch **Metadaten** wie den Eigentümer oder den Nutzer enthalten können

Windows Media Rights Manager Flow



Funktionsweise:

- (1) **Verpacken und Verschlüsselung:** Der Inhalt wird verschlüsselt und zusätzliche Information (z.B. Eigentümer) werden eingebunden (*Package Media*). Der Schlüssel zum Entpacken ist in der separaten Lizenz enthalten
- (2) **Distribution:** Die Inhalte werden auf dem Web-Server gespeichert, können aber auch mittels CD oder e-mail vertrieben werden (*Post Media*).
- (3) **Server für Lizenzen einrichten:** Der Eigentümer bestimmt ein "Clearing House", das seine Rechte wahrnimmt (*License Terms*) und Lizenzen an Kunden verkauft.

Funktionsweise: (cont.)

- (4-6) **Erwerb der Lizenz:** Nutzer wählt digitalen Inhalt aus.
 - Datei kann auf seinem Computer abgespeichert (*Request and Receive Media*) werden
 - Zur Nutzung ist der Erwerb eines lizenzierten Schlüssels notwendig (*Request License / Download License*)
- (7) **Abspielen der Media File:** erfordert einen Player, der den Windows Media Rights Manager unterstützt.
 - Der Nutzer kann dann die Datei gemäß den erworbenen Lizenzbedingungen nutzen.
 - Einschränkungen wie max. Abspielzahl, Ablaufdatum, oder Anzahl Kopien sind möglich.
 - Weitergabe an Drittperson erfordert von dieser ebenfalls den Erwerb eines Schlüssels

9.4.4 Anforderungen an technische Durchsetzung

- Nutzungskontrolle muss **dezentral beim Client** erfolgen
- wie sicherstellen, **dass Client das nicht deaktiviert?**

Versuch einer Antwort:

- Reine Software-Lösungen sind hierfür nicht geeignet.
- **Trusted Computing** (siehe 9.5) bietet:
 - **Secure I/O**: Dadurch wird das Mitschneiden verhindert
 - **Memory curtaining**: Digitale Inhalte in ungeschützter (unverschlüsselter) Form können nicht kopiert werden
 - **Sealed storage**: Nur der DRM-Client, der die Inhalte gespeichert hat, kann auf diese zugreifen
 - **Remote attestation**: Lässt nur die Verarbeitung durch vertrauenswürdigen DRM-Client zu

Alternativen zur rein technischen Lösung? Z.B.

- Fehlverhalten bringt Nachteile: Strafen?
- Wohlverhalten lohnt sich: **Anreizsysteme, Rabattsysteme?**
technische Umsetzung: z.B. anonymes Einlösen von Punkten?
Arbeiten hierzu u.a. im Projekt FlexPay (FG-SEC)

Vorteile von DRMS:

- Nutzungskontrollen, legale Nutzungen, Integrität, Authentizität
- ...? **Was noch?**

Nachteile von DRMS:

- geringe Akzeptanz: Einschränkung der Nutzungen auf spezifische Geräte, keine Privatkopien, Privatheit
- ...? **Was noch?**

Einige offene Probleme im Zusammenhang mit DRM:

- Werden/können DRM-Technologien **standardisiert** werden?
- Standardisierungsgremien (OMA, ...) versus marktdominierende Unternehmen (Apple, MS, ...)
- Wie kann **Interoperabilität** bzgl. Inhalten, Rechten und Geräten erreicht werden?
- Wie gestaltet man technisch die **Weiter- bzw. Rückgabe von Rechteobjekten**?
- Aus juristischer Sicht:
Was bedeutet „**Fair Use**“ (Private Kopie)?
- **Internationale**, Rechtliche Rahmenbedingungen? USA, EU?
- Welche Chancen haben **alternative DRM-Modelle**?
- Hinweis: CAST-Workshop zum Thema DRM: 10.02.05

9.5 Trustmanagement (vgl. Kapitel 2.1 Vertrauen (Trust))

9.5.1 Einführung

- Vertrauen ist graduell, wird auf- und ggf. abgebaut: man **vertraut** einem Subjekt/Objekt, aber ist es **vertrauenswürdig**?
- Nachgewiesene **Qualitätsmerkmale** schaffen Vertrauen: Sicherheit, Verlässlichkeit, Korrektheit
- **Vertrauensaufbau** durch: u.a.
 - Reputation, Rating, Wohlverhalten, Verträge,
- **Vertrauensgrade** abhängig vom Kontext, Beispiel?
- Aufbau von **Vertrauensstrukturen** in der IT-Sicherheit:
 - PKI mit CAs, Zertifikaten, Signaturen
 - Web-of-Trust
 - Integritäts-, Authentizitätsprüfungen?

Frage: wann ist ein System, Programm **vertrauenswürdig**?

- **Vertrauenswürdigkeit:**
„Eine Komponente reagiert wie erwartet/spezifiziert“
- **Erwartetes Verhalten** muss spezifiziert sein: wie?
 - komplizierter Weg: Code erfüllt Beweisverpflichtungen
 - **einfacherer Weg:** Hashwert des getesteten Codes
- Ansatz über **einfacheren Weg** benötigt u.a.
 - eine **vertrauenswürdige Basis**, die Hashwerte berechnet
 - Aufbau einer **Vertrauenskette**: Booten, Laden von SW, ...
 - Sicheres **Nachvollziehen** der Änderungshistorie
 - **Sichere Speicherung** der Hashwerte, Signierschlüssel, ...
 - **Policy** und deren **Enforcement**: welche Änderungen sind zulässig, mit wem darf interagiert werden, ...

Vertrauenswürdige Basis in heutigen Systemen?

- Probleme in aktuellen Systemen: u.a.
 - DMA-Geräte umgehen **Speicherschutz**
 - **Treiber umgehen** aus Performance-Gründen **BS-Kontrollen**
 - Kein **durchgehender Schutz** von der Hardware-Ebene aus

Anforderungen

- Trennen von **Policies** und den **Mechanismen** zur Umsetzung
- **Sicherheitsfunktionen, Basismechanismen in HDW**
- Basismechanismen an **einheitlichen Schnittstellen** anbieten
- Basismechanismen zu komplexen Diensten veredeln (z.B. BS)
- **Durchsetzung von Policies** mit Diensten von BS und Hardware

Fazit: vertrauenswürdige Architektur: aufeinander abgestimmte Hardware, Firmware, Systemsoftware u. Anwendungen

Trusted Computing Initiative:

Ziel: Erfüllung der Anforderungen

9.5.2 Background: TCPA/TCG

- Trusted Computing Platform Alliance (**TCPA**)
 - 1999 Gründung durch Compaq, HP, IBM, Intel, Microsoft
 - 2001 Version 1.0 der TCPA Main Specification
 - 2002 Version 1.1a, 2003 Version 1.1b
 - 2003 bereits über 200 Mitglieder, zu inflexibel!
- Trusted Computing Group (**TCG**) (2003)
 - Übernimmt Rechtsnachfolge der TCPA
 - 2003 Version 1.2 der TCG TPM Specification

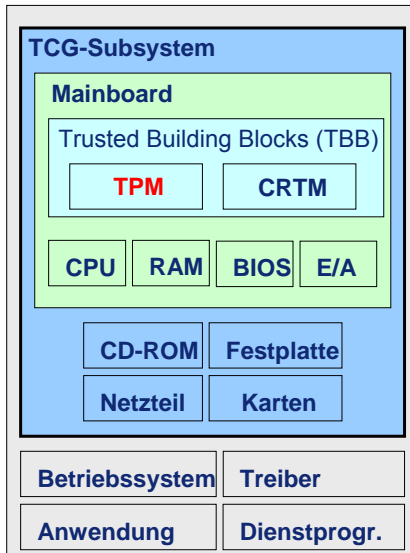
Plattform: Umfasst alle technischen Komponenten

Mainboard, BIOS, CPU, angeschlossene Geräte, ..

System: Plattform plus Bootloader, BS und Anwendungen

9.5.3 Überblick: TCG-Architektur

PC



- **TPM-Chip** Smartcard-Funktion
 - an den PC gebunden
 - u.a. Sicherer Speicher
- **RTM** Root of Trust for Integrity Metrics:
 - **vertrauenswürdige Berechnung von Hashwerten:** BIOS, Bootloader, ...
 - RTM: TBB mit CRTM (Core Root of Trust Measurement)
 - **Aufgabe:** mit Hashwerten:
 - **Attestierung**, d.h. Nachweis der Plattform-Authentizität

Aufbau von Vertrauen

- Schrittweiser Prozesse:
 - startet mit Vertrauensankern, den **Roots of Trust**
 - den Roots of Trust wird ohne weitere Kontrollen vertraut
- **Roots of Trust:**
 - **RTM: Trust for Measurement:** (bis auf CRTM-Teil im TPM) Hashwertberechnung beim Booten (weiter unten genauer), Basis ist CRTM (meist im BIOS)
 - **RTS: Root of Trust for Storage:** (im TPM implementiert) Erstellung und Verwaltung kryptographischer Schlüssel, Schlüsselhierarchie, Wurzel ist Storage Root Key im TPM
 - **RTR: Root of Trust for Reporting** (im TPM implementiert) Ausstellung von Bescheinigungen über Integritätswerte

Eindeutige Identität eines TPM:

- Basisdaten werden bei der Herstellung in den TPM-Chip integriert: dazu gehören der **Endorsement Key** u.a. Zertifikate
- Chip ist damit **eindeutig identifizierbar** → **Privacy-Probleme!**
- **Problemlösung:** Signieren von Daten des TPM erfolgt:
 - **nicht** durch eindeutigen Endorsement Key sondern
 - durch **Attestation Identity Keys (AIK): Pseudonyme**

Besitzerkonzept:

- Chip ist **passiv**, muss erst explizit aktiviert werden
- Aktivierung erfordert die Kenntnis des **Besitzer-Passworts** (siehe nächste Folie) und **physische Anwesenheit** am Gerät
- Besitzer: nicht notwendig der Nutzer des PC, Laptops etc.,
Besitzer kann z.B. der Sys-Admin sein, **Effekt?**

- Besitzübernahme: *Take_Ownership*-Kommando des TPM,
- Ausführung: **Nachweis der physischen Anwesenheit** am Gerät;
z.B. beim Thinkpad: Fn-Taste während des Bootens
- *Take_Ownership*-Kommando:
 - Eingabe:** SHA-1 Hashwert **h** eines **Passwortes**, das ist
damit als **Besitzer-Passwort** festgelegt.
 - Ausführung:** $E(h, \text{Endorsement Key}_{\text{Public}})$ und
Speicherung im nicht flüchtigen Speicher des TPM
- **Effekt:** Bindung des gehashten u, signierten Benutzer-
passworts an die spezifische Plattform
- Nachweis der Kenntnis des Passwortes für TPM-Kommandos,
die **Owner-Berechtigung erfordern** (z.B. Zugriff auf Schlüssel),
- Besitzerwechsel ist nur über spezielle Reset-Sequenz möglich

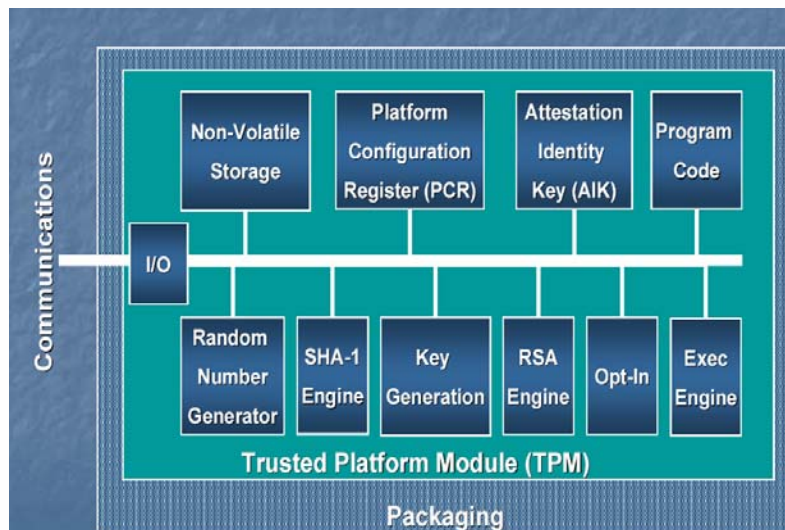
9.5.4 TPM-Chip



9.5.4.1 Überblick

Fest auf Mainboard an Southbridge angebunden

- Hersteller: Atmel, Infineon, National Semiconductor, STMicroelectr.
- Bereits integriert in
 - IBM Thinkpad / Netvista, diverse HP PCs
 - PC Mainboards von Intel erhältlich, Bis jetzt nur in Version 1.1b
- Aufbau
 - ▶ 33 MHz-RISC-Prozessor und RAM (ähnlich wie SmartCard)
 - ▶ Nicht flüchtiger Speicher (non volatile storage)
 - ▶ Counter: Tick-Counter, Boot-Counter
- Funktionen (s. Folie 64): Chip muss explizit aktiviert werden!
 - ▶ Generator für symmetrische und asymmetrische Schlüssel (RSA)
 - ▶ Hardware-basierter Zufallszahlengenerator (RNG)
 - ▶ Signaturerstellung, Hash-Berechnung, Ver-/Entschlüsselung



Funktionen des TPM (Forts.):

- Verschlüsseln kryptographischer Schlüssel (**Binding bzw. Wrapping**), um diese auch außerhalb des TPM zu verwalten
- Hardware-unterstützter, **geschützten Speicher: PCR-Register**, Zugriff nur über Sicherheitsdienste (z.B. Verschlüsseln)
- Erstellung signierter Auskünfte (**Attestation, Reporting**), zur Signaturerstellung dienen **Attestation Identity Keys**
- Versiegeln (**Sealing**): Verschlüsselung von Daten unter Einbeziehung von Hashwerten aus PCR-Registern: Entschlüsseln erfordert den korrekten Zustand (PCR-Register)
- **Opt-In**: nur autorisierte (Nachweis der Kenntnis eines Besitzer-Passwortes) Besitzer der Plattform können den Chip aktivieren (default: deaktiv) bzw. deaktivieren.
- **vertrauenswürdiger Timer**, z.B. für Gültigkeit von Zertifikaten

9.5.4.2 Besondere Schlüssel

Endorsement Key (EK): eindeutige Identität des TPM-Chip

- ▶ RSA-Schlüsselpaar (2048 Bit) im nicht-flüchtigen Speicher
- ▶ Eindeutiger, dem TPM zugeordneter Schlüssel
- ▶ Wird bei Produktion des Chips intern generiert oder importiert
- ▶ Privater Schlüssel-Teil nur im TPM (nicht exportierbar)
- ▶ Öffentlicher Schlüssel-Teil zum Verschlüsseln von sensiblen Befehlen an TPM

Storage Root Key (SRK): Master-Key für Schlüsselhierarchie

- ▶ RSA-Schlüsselpaar (2048 Bit) im nicht-flüchtigen Speicher
- ▶ Wurzel der Schlüsselhierarchie (mit Passwort schützbar)
- ▶ Privater Schlüssel-Teil nur im TPM (nicht exportierbar)
- ▶ Wird bei jeder Besitzerübernahme neu generiert (kein Recovery!)

Attestation Identity Keys (AIK): Pseudonyme zum Signieren von Reports

- ▶ RSA-Schlüsselpaar (2048 Bit) im nicht-flüchtigen Speicher
- ▶ Pseudonyme zur Signatur Erstellung (Privacy)
- ▶ Privater Schlüssel-Teil nur im TPM (nicht exportierbar)
- ▶ Nur zum Signieren von Informationen, die vom TPM kommen

Flüchtiger und Nicht-flüchtiger TPM-Speicher

Kryptographische Funktionen	Nicht-flüchtiger Speicher	Flüchtiger Speicher
RNG	DIR0, ... (160 Bit)	Key-Slot 0
SHA-1	Endorsement Key (2048 Bit)	...
HMAC	Storage Root Key (2048 Bit)	Key Slot 9
Schlüssel-generierung	Owner Auth Secret (160 Bit)	PCR 0
Ver- und Entschlüsselung	ggf. AIKs	...
		PCR 15

9.5.4.3 Schlüsselmanagement

Schutz kryptographischer Schlüssel

- ▶ TPM dient als sicherer Speicher
- ▶ Geschützter Speicher auch außerhalb des TPMs (externer Datenträger)
- ▶ Verschlüsselte Speicherung in sogenannten „Key-Blobs“

Schlüsselhierarchie

- ▶ Speicherung der Schlüssel in einer frei definierbaren Baum-Struktur
- ▶ Wurzel: **Storage Root Key (SRK)**
- ▶ Zwischenknoten: nur Verschlüsselungs-Schlüssel
- ▶ Blätter: Verschlüsselungs-Schlüssel, Signatur-Schlüssel,...

Schlüsselattribute

- ▶ Verwendungszweck: Ver- / Entschlüsselung oder Signatur-Erstellung
- ▶ **Migrierbarkeit**: Wenn ein Schlüssel mittels TPM generiert wurde und als nicht migrierbar eingestuft ist, dann ist der private Teil nicht exportierbar

Zertifikate zur Bestätigung bestimmter Eigenschaften

Endorsement Credential (EC)

- ▶ Zweck: Schlüsselerzeugung und Übertragung des Endorsement Key ist korrekt
- ▶ Inhalt: Hersteller, TPM-Version, öffentlicher Teil des EK
- ▶ Signatur: TPM Hersteller (verantwortlich für Funktion)

Conformance Credential (CC)

- ▶ Zweck: Design und Implementierung des TPMs und der Plattform korrekt
- ▶ Inhalt: Aussagen zu Funktions-Tests, ob Anforderungen der TCG erfüllt sind
- ▶ Signatur: beliebige Instanz, die zur dieser Evaluierung in der Lage ist

Zertifikate (Forts.)

Platform Credential (PC)

- ▶ Zweck: Bindung TPM an Plattform
- ▶ Inhalt: Sicherheitseigenschaften der Plattform, und Verweis auf EC und PC
- ▶ Signatur: Plattform Hersteller

Validation Credential (VC)

- ▶ Zweck: Hardware-Komponente ist vertrauenswürdig
- ▶ Inhalt: Sicherheitseigenschaften der Komponente
- ▶ Signatur: Instanz, die zur Evaluierung in der Lage ist

(Attestation) Identity Credential (IC)

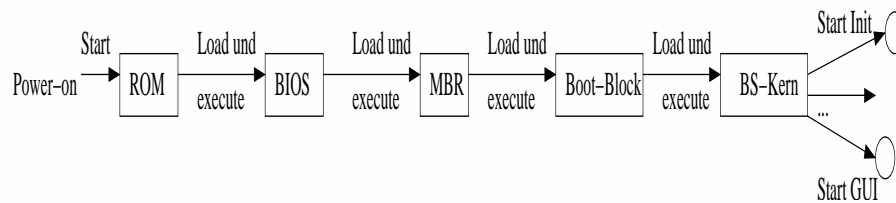
- ▶ Zweck: Gültigkeit des zugehörigen AIK
- ▶ Inhalt: Zusatzinformationen zum AIK
- ▶ Signatur: ausgewählte Privacy CA

9.5.5 Trusted Boot

Ziele

- Erkennung einer manipulierten Systemkonfiguration
- Verwendung für Remote Platform Attestation

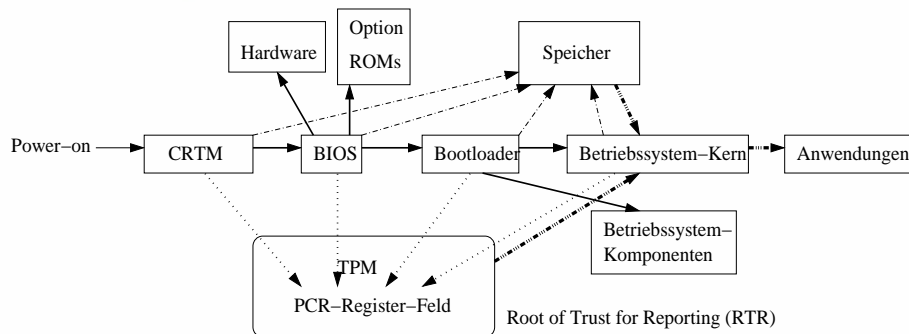
Erinnerung: Boot-Sequenz



Trusted Boot-Prozess: Vorgehen

- Aufbau einer **Vertrauenskette** vom BIOS Boot Block (BBB) bis zur Anwendungs-Ebene (nur bis zum Bootloader spezifiziert)

Aufbau einer Vertrauenskette



Legende

- Berechnung von Hashwerten
- Speicherung im sicheren TPM-Speicher
- .-.-.- Erstellen von Integritäts-Reports
- Loggen von Aktivitäten

Trusted Boot (Forts.)

- **Integritäts-Messungen** vor dem Laden der jeweiligen Firmware / Software
- **Speicherung der Werte im geschützten (flüchtigen) Speicher** des TPMs: Platform Configuration Register (PCR)
- **Inkrementelles Vorgehen**: weitere Integritäts-Messungen werden mit dem Inhalt des PCR konkateniert und gehasht:

$$\text{PCR}(i)_{t+1} = \text{SHA-1}[\text{PCR}(i)_t \mid \text{neue Messung}]$$

Zu beachten:

- TPM berechnet nur Hashwerte, aber **er prüft nicht**, ob die Software wirklich vertrauenswürdig ist!
- TPM bietet nur Mechanismus, Policy ist BS-Aufgabe

Fazit: Vorgehen ist kein sicheres Booten!

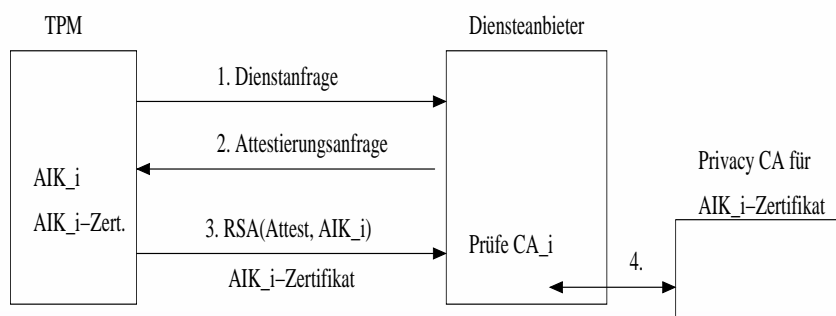
9.5.6 Remote Platform Attestation

Ziel: Entferntem Anfrager Zustand der Plattform bescheinigen

Basis: Credentials, um Vertrauen in die Integrität der Plattform zu steigern

Vergrößerter Ablauf:

- **Szenario:** Plattform möchte Dienste eines Anbieters nutzen



Ablauf

- Anfrage des Diensteanbieters nach Zustand des TPM
- Root of Trust for Reporting (RTR): erstellt Attestierungen:
 - Wahl eines AIK , Zugriff auf Daten im PCR
 - Attest= Sig(Daten|Nonce, privaterTeil des AIK)
- Senden: Attest und AIK Credential an den Anfrager
- Anfrager prüft die Vertrauenswürdigkeit derjenigen CA, die das AIK Credential erstellt hat
- Positive Überprüfung:
 - Anfrager prüft Attest mit öffentlichem AIK-Schlüssel
 - falls Attest seinen Anforderungen entspricht:
angefragte Kommunikation mit der Plattform aufnehmen

Probleme?

Erstellung eines AIK

AIK generieren

- TPM Befehl: TPM_MakeIdentity
- RSA-Schlüsselpaar (2048 Bit) wird mittels TPM generiert
- Signatur des neu erstellten Schlüssel wird generiert (privater Teil des EK)
- Namen vergeben (frei wählbar, z. B. Nutzer oder Zweck)
- Namen der gewünschten Privacy CA angeben

Zusammenstellen der Informationen

- TSS Befehl: TSS_CollateIdentityRequest
- Informationen aus Schritt 1 bündeln
- Credentials: Endorsement, Platform, Conformance hinzufügen

Senden an Privacy CA

- Verschlüsselt mit Public Key der Privacy CA

Privacy CA entschlüsselt Nachricht und prüft

- Credentials (Zweck: Anfrage von echter TP)
- Name der Privacy CA (Attestierung von dieser CA gewünscht)
- Signatur des AIK-Schlüssels (Öffentlicher Teil des EK aus EC)

Erstellung eines AIK (2)

Privacy CA erstellt eine Antwort

- Symmetrischen Schlüssel K erzeugen
- Zertifikat (Attestation Identity Credential) mit K verschlüsseln
- Schlüssel K mit öffentlichem Teil von EK verschlüsseln
- Hashwert vom öffentlichen Teil des AIKs verschlüsselt mit senden

Privacy CA sendet diese Antwort an den Anfrager zurück

TPM entschlüsselt (TPM_activateIdentity)

- Entschlüsseln des symmetrischen Schlüssels K
- Entschlüsseln und prüfen des Hashwertes des AIKs

Zertifikat entschlüsseln (TSS_recover_TPM_identity)

9.5.7 Grenzen des TPM und einige Probleme

- Kein Schutz vor physikalischen Angriffen (Seitenkanalangriff)
- **Tamper-Evident**, aber nicht **Tamper-Resistant**
- Passives Gerät, d.h. kann nicht aktiv eingreifen (u.a. Policies)
- TPM ist **plattform-**, **nicht personenbezogen** wie Smartcard
- Keine Möglichkeit kryptographische Algorithmen nachzuladen
- **Vertrauenskette** beim Booten ist **lückenhaft**, nicht alle Komponenten werden erfasst (viel zu viele),
- Attestierung der Konfigurierung ist stets lückenhaft
- **Vertrauenswürdigkeit** (im Sinne korrekter Funktionalität) attestiert der TPM bzw. die Plattform **nicht!**
- **kein Schutz** vor Viren, Würmer, Trojaner!

Einige Probleme im Zusammenhang mit dem TPM

Backup-Konzept

- **fehlt**, Zerstörung des Chips zerstört nicht-migrierbare Schlüssel: SRK, EK, AIK: keine Entschlüsselung extern gespeicherter, verschlüsselter Daten mehr möglich

Attestierung/Sealing

- Attestierung/Sealing erfordert rel. stabile Konfiguration, **aber** das Nachinstallieren von **Patches** ist in hoher Frequenz notw.

DoS Angriffe

- Entfernte Anforderung rechenintensiver Funktionen des TPMs
- Temporäre Deaktivierung des TPMs (Reboot nötig)

Einige Probleme im Zusammenhang mit dem TPM (Forts.)

Replay Attacke

- Zustand z1 zum Zeitpunkt t1 speichern, fortfahren
- nach Reboot: Wiederherstellung des Zustands z1
- durch **Reboot-Counter** (ab Version 1.2) erkennbar

Kritik/Probleme:

Bem: richtet sich gegen Dienste, die den TPM nutzen,
Der TPM stellt nur Basismechanismen bereit!

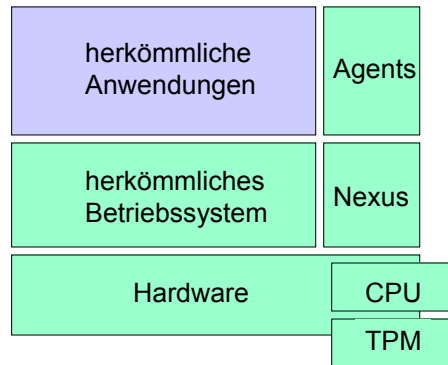
- DRM: Content-Anbieter können bestimmte Atteste fordern, damit der Content nutzbar wird (z.B. Lizenzen vorzeigen)
- Nutzer-PC gelangt unter ‚fremde‘ Kontrolle, da nur die Auflagen des Anbieters erfüllt werden, nicht die Nutzerbelange

9.5.8 Vertrauenswürdige Plattform-Architekturen

Zu einer vertrauenswürdigen Plattform gehört auch ein **BS**

Microsoft: **NGSCB** (Next Generation Secure Computing Base)

- **Idee:** sicherheitskritische Module im **Nexus** zusammengefasst.
- **Nexus in geschützter Hardware-Umgebung** ausgeführt,
- **Nexus besitzt spezifischen Privilegien**, mehr Rechte als herkömmliche Betriebssystem



Ursprüngliche Architektur, basierend auf TPM

Aktuelle Arbeiten: Longhorn, rel. klassisch Aufweichung der ursprünglichen Trennung

