IT Security

Prof. Dr. Henning Pagnia

DHBW Mannheim

Herbst 2022

Wichtiges zur Vorlesung

Prof. Dr. Henning Pagnia

- Wirtschaftsinformatik
- Email: *pagnia@dhbw-mannheim.de*
- Telefon: *0621 / 4105-1131*
- Raum: *149 B*

Sicherheit ?

Vier Mal hintereinander

Bereits vor einer Woche hätten vier Blitzeinschläge das europäische Rechenzentrum getroffen und die Stromzufuhr zu Speichersystemen für Westeuropa unterbrochen. Die Probleme hätten bis zum vergangenen Montag angedauert. Es gebe zwar eine Notversorgung für den Strom, aber dennoch seien einige ältere Server vom Netz gegangen. Zu einem größeren Datenverlust sei es aber nicht gekommen. Lediglich in weniger als 0,000001 Prozent des Speicherplatzes in der belgischen Gemeinde St. Ghislain seien Daten unwiderruflich verloren gegangen, schrieb Google in den Statusmeldungen zu seiner Cloud Platform.

(Quelle: TecChannel.De, 20. August 2015)

Sicherheit ? (Forts.)

Ich wär so gerne Millionär ...

In Ekaterinenburg (Russland) zahlte ein Bankkunde 2000 Rubel (ca. 60 EUR) per Bankautomat auf sein Konto ein. Auf dem Konto gutgeschrieben wurde jedoch ein Betrag von 2 Milliarden Rubel! Als der Kunde das auf seinen Kontoauszügen entdeckte, wollte er die Bank sofort über den Irrtum zu informieren, doch der Bankangestellte im Schalterraum zeigte keinerlei Interesse.
Also ging der Kunde wieder zum Automaten, wo er munter Geld abhob und gleich wieder einzahlte. Schließlich war er Besitzer von stolzen 20 Milliarden Rubel (ca. 600 Mio. EUR) sowie mehrerer Schuhkartons voll Bargeld. Erst als er die Kartons in der Bank präsentierte, zeigten sich der Angestellte hinreichend geschockt und schaltete sofort alle Automaten ab.
(Quelle: Risks Digest, Vol. 24.40, 28. August 2006)

Sicherheit ? (Forts.)

Kleiner Vertipper

Eine falsche Gewichtseingabe im Bordcomputer ist der Grund für einen Beinahe-Crash am Flughafen Melbourne vor vier Wochen. Die Airbus-Maschine der Emirates mit 255 Passagieren an Bord kam erst kaum vom Boden weg und streifte dann mit dem hinteren Teil den Asphalt der Startbahn, ehe sie es knapp über den zweieinhalb Meter hohen Zaun des Flughafengeländes schaffte.
Der Pilot kehrte sofort um und landete wieder. Im Bordcomputer des Airbus A340-500 sei das Gewicht der Maschine beim Start falsch eingegeben worden, teilte die Transportsicherheitsbehörde am Donnerstag mit. Ein Sprecher kündigte eine weitere umfangreiche Untersuchung an und bezeichnete den missglückten Start als "sehr schweren Zwischenfall".
[...] Die Maschine wurde so beschädigt, dass sie auch vier Wochen später noch nicht wieder im Dienst war. Nach Informationen der Zeitschrift "Aviation Herald" gab die Crew eine '2' statt einer '3' in den Computer ein und reduzierte damit das Startgewicht der Maschine um 100 Tonnen auf 260 Tonnen.
(Quelle: Spiegel Online, 30. April 2009)

Sicherheit ? (Forts.)

Schon ein Kinderwunsch: Ferngesteuertes Auto

Der US-Journalist Andy Greenberg setzte sich [für das US-Tech-Magazin "Wired"] ans Steuer eines Geländewagens, der von den US-IT-Sicherheitsexperten Charlie Miller und Chris Valasek per Funk manipuliert wurde. Die Hacker haben eine Sicherheitslücke in der Online-Anbindung der Multimedia-Stereoanlage des neuen Jeep Cherokee ausgemacht, die ihnen die Manipulation von Fahrzeugsystemen ermöglicht.
Greenberg erduldete stoisch die Hip-Hop-Musik auf Maximallautstärke und die eiskalte Luft aus der Klimaanlage. Dann griffen die Hacker ausgerechnet auf einer Autobahnbrücke ohne Seitenstreifen auch noch in die Getriebe-Automatik des Wagens ein und stellten den Antriebsstrang auf Leerlauf. Greenberg konnte laut eigener Beschreibung rein gar nichts tun, während im Rückspiegel schwere Lastwagen immer größer wurden. Er bettelte per Telefon-Verbindung zu den Hackern um Gnade und rettete sich schließlich, indem er die Zündung des Wagens einmal aus- und wieder einschaltete und auf den nächsten Parkplatz fuhr. Dort [...] stellten die Hacker die Bremsen des Wagens ab, so dass der Jeep mit Greenberg am Steuer unkontrolliert in einen Graben rutschte.
(Quelle: welt.de, 22. Juli 2015)

Sicherheit ? (Forts.)

Aus dem Gleichgewicht

Nach Meldungen der kanadischen Presse soll im Sommer 2001 ein Transatlantikflug der Canadian Air in ernsthafte Schwierigkeiten geraten sein. Den Berichten zufolge soll ein Computerprogramm ein Leck in einem Tank als Ungleichgewicht gemeldet haben. Um dieses Ungleichgewicht zu korrigieren, wurde Kerosin von einem intakten Tank in den defekten Tank gepumpt. So waren schließlich beide Tanks leer. Die Passagiere verdanken ihr Leben den Fähigkeiten und dem Geschick des Piloten und dem glücklichen Umstand, dass das Flugzeug auf einem Flughafen auf den Azoren notlanden konnte.

(Quelle: John Johnson, in: Risk Digest, 6. März 2002)

Sicherheit ? (Forts.)

Die Bahn fährt immer

Fahrgäste der Bahn haben es am Samstag sehen können: Auf den Anzeigentafeln verschiedener Bahnhöfe zeigt WannaCry sein Gesicht. Auch die Kameras auf den Bahnhöfen sind betroffen. Abgesehen davon soll der Betrieb normal laufen.

[...] Der Schädling habe die Systeme der Anzeigentafeln auf den Bahnhöfen befallen, bestätigte das Unternehmen am Samstag. Nach Angaben des Bundesinnenministeriums ist zudem die Videoüberwachung auf Bahnhöfen betroffen.

[...] Die Ransomware hat am Freitag unter anderem die Rechner des britischen Gesundheitssystems befallen, was auf der Insel zu erheblichen Störungen bei der medizinischen Versorgung der Bevölkerung geführt hat. Auch spanische Unternehmen, darunter der Netzbetreiber Telefónica, sind betroffen. Inzwischen scheint die weitere Verbreitung des Virus zumindest verlangsamt, nachdem ein Sicherheitsforscher einen Weg gefunden hat, wie sich der Wurm stilllegen lässt. Infizierten Systemen hilft das aber nicht.
(Quelle: heise.de, 13. Mai 2017)

Sicherheit ? (Forts.)

Sicherheitsüberprüfung

<Mastercard-Logo>

Sehr geehrte Kundin, sehr geehrter Kunde,
Ihr Konto wurde von unserem System für eine zufällige Überprüfung ausgewählt dabei stellten wir fest, dass Sie Ihre persönlichen Daten seit über einem Jahr nicht mehr aktualisiert haben. Ihr Konto stellt mit veralteten Daten ein Sicherheitsrisiko dar infolgedessen sahen wir uns leider gezwungen Ihr Kartenkonto vorerst zu schließen.

Um Ihr Konto wieder zu öffnen und Ihre Karte freizuschalten bringen Sie bitte innerhalb der nächsten 48 Stunden Ihre Daten auf den neusten stand andernfalls bleibt Ihr Konto geschlossen. Bitte beachten Sie, dass danach nur noch ein Wiedererlangen der Karte möglich ist wenn Sie bei Ihrer Hausbank einen Antrag auf Neueröffnung erteilen.

<WEITER>* * Link zu https://fixed 3for3 .com/3847zjfnfuaszfaudjasmdu7348u54.php

Wir danken Ihnen für Ihre Mithilfe und bitten die Umstände zu entschuldigen.

Mit freundlichen Grüßen,
Ihr Mastercard-Team
(Quelle: aus meinem persönlichen Postfach, 30. April 2020)

Sicherheit ? (Forts.)

Was in der Zeitung steht ...

On 9 Mar 2000, *The Boston Globe* reported that a hacker had broken into an MIT computer system and changed the grades of 22 students in a cell biology class. Some grades were raised and some were lowered but not in any sensible pattern. Teacher Harvey Lodish announced to his class (on Thursday March 2) that a cheating scandal had been uncovered. Suspicions did not point to any particular students in the class. No motive could be inferred and it was believed that an unknown third party had done the hacking for no discernable reason.

(to be continued)

Sicherheit ? (Forts.)

Was in der Zeitung steht ...

On 10 Mar 2000, *The Boston Globe* reported that the mystery had been solved. The grades were recorded in a spreadsheet and a teaching assistant had unknowingly sorted the student name column without also sorting the grades columns. No intruder, no hack, no cheating scandal. Officials discovered the source of the mistake only after spending a full week ruling out the possibility of infiltration.
It seems to me that bound paper ledger books would be a much better tool for keeping grade records, at least for this teacher and his assistants.

Quelle: www.boston.com, find Archives, search for „Lodish“.
From: Mark Lutton <mlutton@ma.ultranet.com>

Ablauf der Vorlesung

Themengebiete

- Angriffsmodellierung
- Klassifikation sicherer Systeme
- Kryptografie
- Benutzerverwaltung und -authentisierung
- Dateischutz und Zugriffskontrolltechniken
- Speicherschutz und Betriebssystemintegrität
- Angriffe in Netzwerken, insb. im Internet
- Firewall und Intrusion Detection Systeme

Literaturliste

[Ande2008] *Ross Anderson*: Security Engineering, 2nd edition, Wiley, 2008, ISBN-13: 978-0470068526.

[Beut2015] *Albrecht Beutelspacher*: Kryptologie – Eine Einführung in die Wissenschaft vom Verschlüsseln, Verbergen und Verheimlichen, 10. Auflage. Springer Spektrum, 2015. ISBN 978-3-658-05975-0.

[Buch2016] *Johannes Buchmann*: Einführung in die Kryptographie, 6. Auflage. Springer Spektrum, 2016. ISBN 978-3-642-39774-5.

[Ecke2018] *Claudia Eckert*: IT-Sicherheit: Konzepte – Verfahren – Protokolle, 10. Auflage. De Gruyter Oldenbourg, 2018. ISBN 978-3-11-055158-7.

[GS2003] *Simon Garfinkel und Gene Spafford*: Practical Unix and Internet Security, 3. Auflage, O'Reilly & Associates Inc., 2003, ISBN: 0596003234

[PP2016] *Christof Paar und Jan Pelzl*: Kryptografie verständlich – Ein Lehrbuch für Studierende und Anwender. Springer Vieweg, 2016. ISBN 978-3-662-49296-3.

[Schn1995] *Bruce Schneier*: Applied Cryptography, 2nd edition. John Wiley & Sons, 1995. ISBN-13: 978-0-471-11709-4.

[Schn2004] *Bruce Schneier*: Secrets & Lies, Dpunkt, 2004, ISBN-13: 978-3898643023

[TB2016] *Andrew S. Tanenbaum und Herbert Bos*: Moderne Betriebssysteme, 4. Auflage, Prentice Hall, 2016, ISBN-13: 978-3868942705

Was bedeutet "Computersicherheit"?

Eine eher intuitive Definition

Ein Computer ist genau dann sicher,
wenn sich seine Hard- und Software
erwartungsgemäß *verhalten.*

Was heißt „erwartungsgemäß"?

- System ist funktionsfähig
- System arbeitet gemäß seiner Spezifikation
- Daten sind dauerhaft gespeichert
- Daten sind gegen unberechtigten Zugriff geschützt

Was bedeutet "Computersicherheit"? (Forts.)

Zitate

„Security is a process, not a product."

Bruce Schneier (2000):
www.schneier.com/crypto-gram/archives/2000/0515.html

„Security is a trade-off."

Bruce Schneier (2008):
www.schneier.com/essays/archives/2008/01/the_psychology_of_se.html

Was bedeutet "Computersicherheit"? (Forts.)

Aspekte von Sicherheit

- Vertraulichkeit von Informationen
- Datenintegrität
- Systemintegrität
- Zugangskontrolle
- Verfügbarkeit des Systems
- Protokollierung der Systemaktivität

Computersicherheit

Gefährdungen der Sicherheit

- Naturkatastrophe (Stichwort: „höhere Gewalt“)
- Softwarefehler (bei Entwurf oder Implementierung)
- Benutzungsfehler
- unberechtigte Benutzer (Spionage, Sabotage)
- unbegründetes Vertrauen

Was bedeutet "Computersicherheit"? (Forts.)

Gegenmaßnahmen

- Hardware-Redundanz (auch: geographische Redundanz)
- Software-Redundanz (Replikation, Backups)
- Software-Engineering-Techniken:
  formale Spezifikation, strukturierter Entwurf
  (Wiederverwendung; ist aber auch kritisch $\Rightarrow$ *Wieso?*)
- regelmäßige Sicherheitsupdates (Sicherheit als "Prozess")
- Virenscanner, Firewalls, Intrusion Detection, ...
- Manuals, Check–Listen, Schulungen
- Vier-Augen-Prinzip (bei Bedienung und Entwicklung)
- Honey-Pots, Honey-Nets
- Programmbeweise, Tests (bezgl. Sicherheit möglich?)
- n-Version Programming

Aus dem StGB *

§202a: Ausspähen von Daten

(1) Wer unbefugt sich oder einem anderen Zugang zu Daten, die nicht für ihn bestimmt und die gegen unberechtigten Zugang besonders gesichert sind, unter Überwindung der Zugangssicherung verschafft, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.
(2) Daten im Sinne des Absatzes 1 sind nur solche, die elektronisch, magnetisch oder sonst nicht unmittelbar wahrnehmbar gespeichert sind oder übermittelt werden.

---

* *Quelle:* http://www.gesetze-im-internet.de/stgb/

Aus dem StGB (Forts.)

§202b: Abfangen von Daten

Wer unbefugt sich oder einem anderen unter Anwendung von technischen Mitteln nicht für ihn bestimmte Daten (§202a Abs. 2) aus einer nichtöffentlichen Datenübermittlung oder aus der elektromagnetischen Abstrahlung einer Datenverarbeitungsanlage verschafft, wird mit Freiheitsstrafe bis zu zwei Jahren oder mit Geldstrafe bestraft, wenn die Tat nicht in anderen Vorschriften mit schwererer Strafe bedroht ist.

Aus dem StGB (Forts.)

§202c: Vorbereiten des Ausspähens und Abfangens von Daten

(1) Wer eine Straftat nach §202a oder §202b vorbereitet, indem er

1. Passwörter oder sonstige Sicherungscodes, die den Zugang zu Daten (§202a Abs. 2) ermöglichen, oder
2. Computerprogramme, deren Zweck die Begehung einer solchen Tat ist,

herstellt, sich oder einem anderen verschafft, verkauft, einem anderen überlässt, verbreitet oder sonst zugänglich macht, wird mit Freiheitsstrafe bis zu einem Jahr oder mit Geldstrafe bestraft.

($\Rightarrow$ "Hacker-Paragraph", 8/2007)

Aus dem StGB (Forts.)

§263a: Computerbetrug

(1) Wer in der Absicht, sich oder einem Dritten einen rechtswidrigen Vermögensvorteil zu verschaffen, das Vermögen eines anderen dadurch beschädigt, dass er das Ergebnis eines Datenverarbeitungsvorgangs durch unrichtige Gestaltung des Programms, durch Verwendung unrichtiger oder unvollständiger Daten, durch unbefugte Verwendung von Daten oder sonst durch unbefugte Einwirkung auf den Ablauf beeinflusst, wird mit Freiheitsstrafe bis zu fünf Jahren oder mit Geldstrafe bestraft.

Aus dem StGB (Forts.)

§303a: Datenveränderung

(1) Wer rechtswidrig Daten (§202a Abs. 2) löscht, unterdrückt, unbrauchbar macht oder verändert, wird mit Freiheitsstrafe bis zu zwei Jahren oder mit Geldstrafe bestraft.
(2) Der Versuch ist strafbar.
(3) Für die Vorbereitung einer Straftat nach Absatz 1 gilt §202c entsprechend.

Aus dem StGB (Forts.)

§303b: Computersabotage

(1) Wer eine Datenverarbeitung, die für einen anderen von wesentlicher Bedeutung ist, dadurch erheblich stört, dass er

1. eine Tat nach §303a Abs. 1 begeht,
2. Daten (§202a Abs. 2) in der Absicht, einem anderen Nachteil zuzufügen, eingibt oder übermittelt oder
3. eine Datenverarbeitungsanlage oder einen Datenträger zerstört, beschädigt, unbrauchbar macht, beseitigt oder verändert,

wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.
(2) Handelt es sich um eine Datenverarbeitung, die für einen fremden Betrieb, ein fremdes Unternehmen oder eine Behörde von wesentlicher Bedeutung ist, ist die Strafe Freiheitsstrafe bis zu fünf Jahren oder Geldstrafe.

Wichtig für Unternehmen: Sicherheitspolicy

Definieren einer (**umfassenden**) „Security Policy“

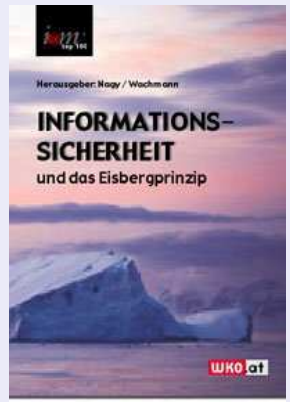
- **Was** soll geschützt werden und **warum**?
- **Wovor** soll es geschützt werden? ($\Rightarrow$ *Threat-Modelling*)
- Gegenmaßnahmen ($\Rightarrow$ *Counterattacks*)
- Gebäudesicherheit
  $\Rightarrow$ Sicherheitsbereiche, Hinterausgänge, Nachtarbeit, ...
- Informationssicherheit
  $\Rightarrow$ öffentliche Telefonate, Kopiererzugang, Heimarbeiter, ...
- Organisatorische Sicherheit
  $\Rightarrow$ Personalschulungen, Überprüfung von Aushilfen, Diebstahl-Handling, Datensicherung, Desaster-Recovery, ...

Sicherheitspolicy (Forts.)

Erstellen meist in drei Schritten

- Allgemeine Beschreibung der Sicherheitsgrundlagen und Sicherheitsorganisation des Unternehmens
  ⇒ Posten eines Sicherheitsbeauftragten
- Implementierungsregeln für die einzelnen Aspekte
  ⇒ Ermittlung des Schutzbedarfs einzelner Systeme bzw. Anwendungen, Aufbau einer technischen Sicherheitsinfrastruktur
- Regeln für die Endnutzer der EDV-Systeme
  ⇒ Passwort-Wahl, allg. Kenntnisnahme der Bedrohungen
- Vgl. "Informationssicherheit und das Eisbergprinzip"

  (⇒ http://sicherheitskultur.at/#eisberg)
  – viele interessante Themen, gute Link-Sammlung



- Ebenfalls interessant: BSI-Grundschutz (⇒ http://www.bsi.de)

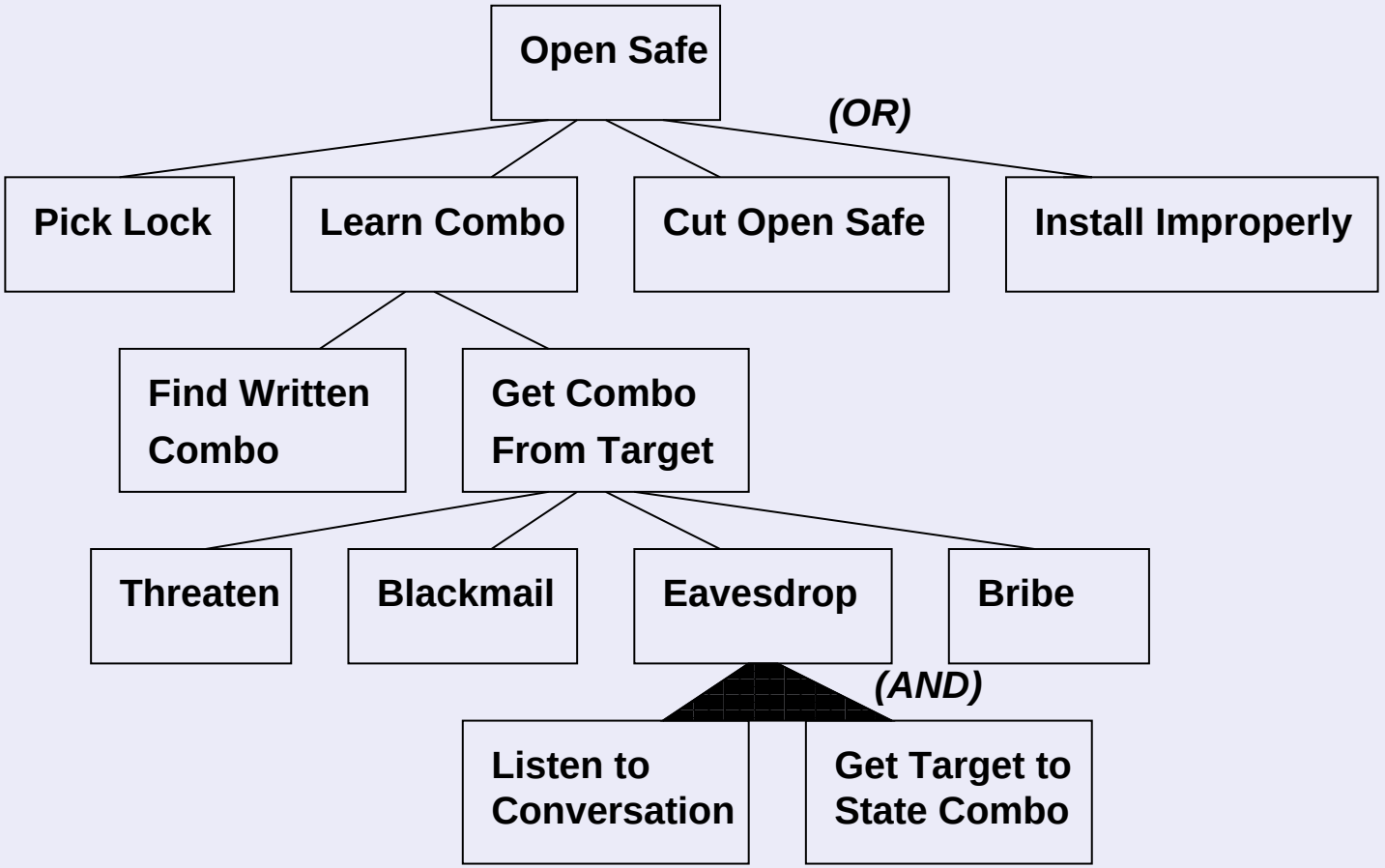
Threat Modelling

Attack Trees

- eingeführt von Bruce Schneier, 1999
- Top-Down Zerlegung
- methodische Beschreibung des Bedrohungspotenzials
- Abschätzung existierender Risiken:
  - Bewertung nach möglichen bzw. unmöglichen Angriffen
  - Bewertung nach Kosten eines Angriffs
  - Bewertung nach Wahrscheinlichkeiten eines Angriffs
  - Finden des kostengünstigsten Angriffs
  - Finden aller "erschwinglichen" Angriffe

Attack Tree: Beispiel

Beispiel: Wandtresor



Attack Tree: Beispiel (Forts.)

Textuelle Darstellung des Attack-Tree

Ziel: Öffnen des Tresors

1. Brechen der Zahlenkombination des Schlosses (OR)
2. Öffnen mittels Schweißbrenner (OR)
3. Tresor fehlerhaft installieren (OR)
4. Zahlenkombination erfahren (OR)

4.1. Zettel mit Kombination finden (OR)

4.2. Kombination vom Verantwortlichen erfahren (OR)

4.2.1 Verantwortlichen bedrohen (OR)

4.2.2 Verantwortlichen erpressen (OR)

4.2.3 Verantwortlichen belauschen (OR)

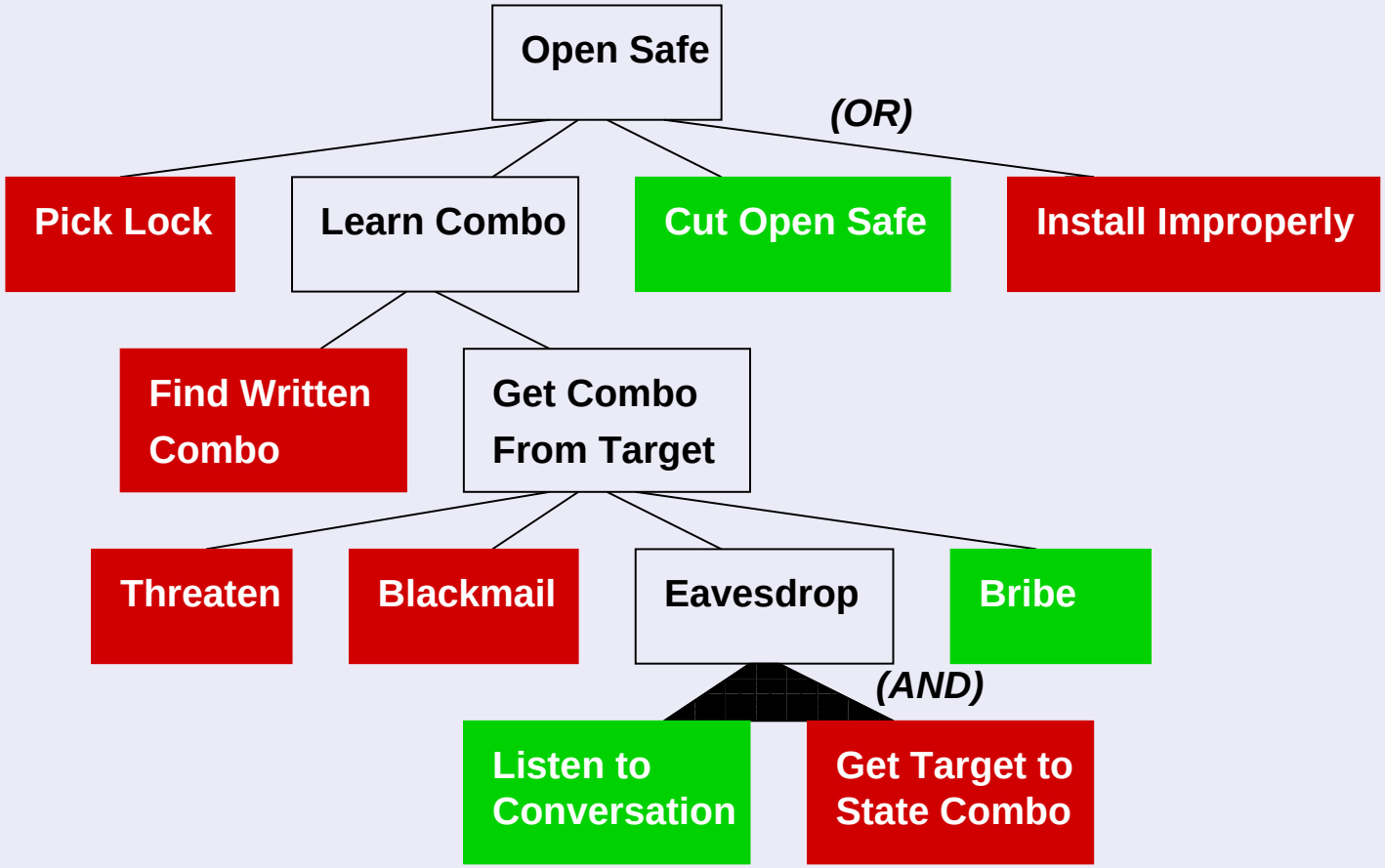
4.2.3.1 Konversation belauschen (AND)

4.2.3.2 Verantwortlichen dazu bringen, die Komb. selbst zu verraten (AND)

4.2.4 Verantwortlichen bestechen (OR)

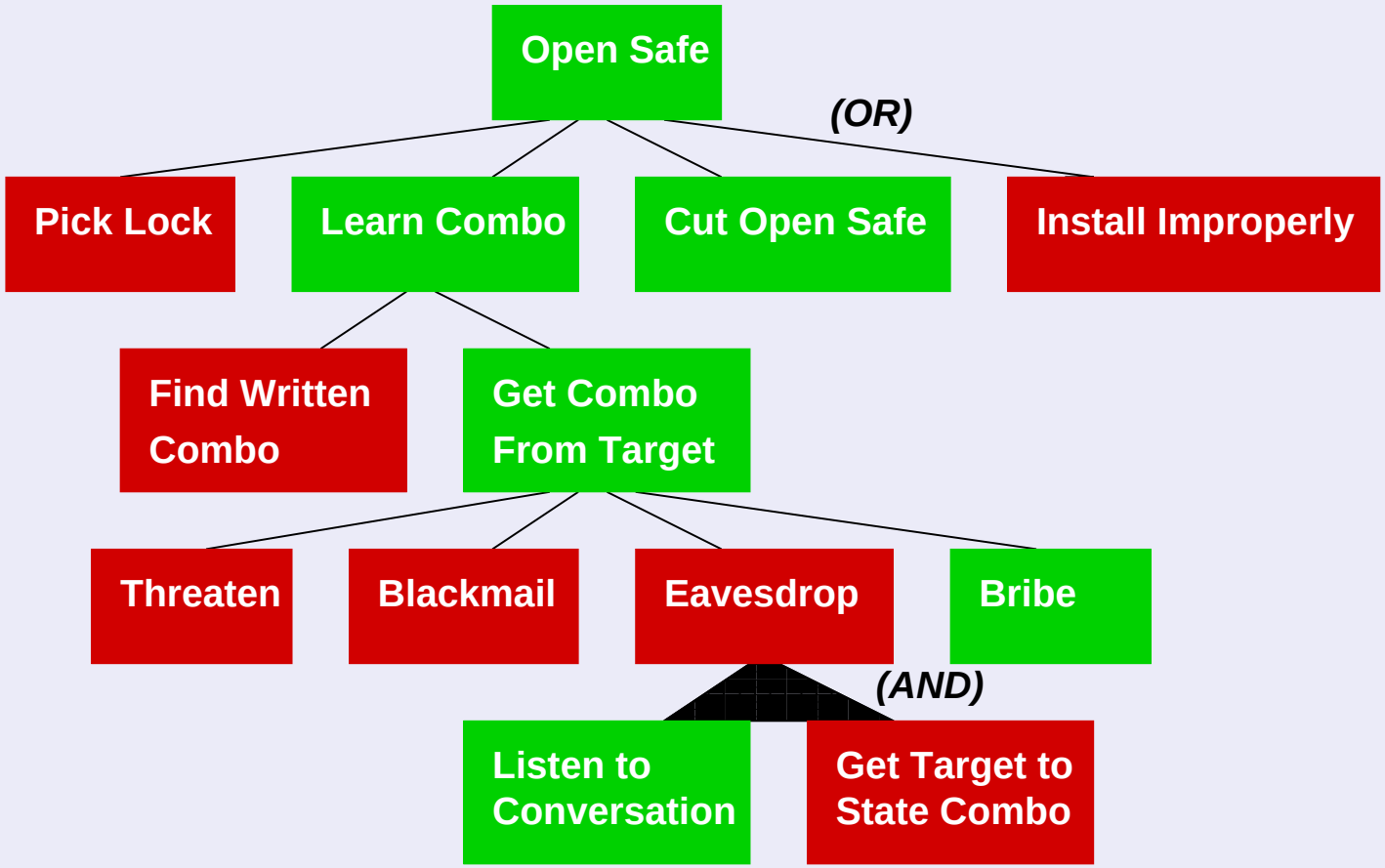
Attack Tree: Beispiel (Forts.)

(1) Bewerten der Blattknoten: möglich (grün) / unmöglich (rot)



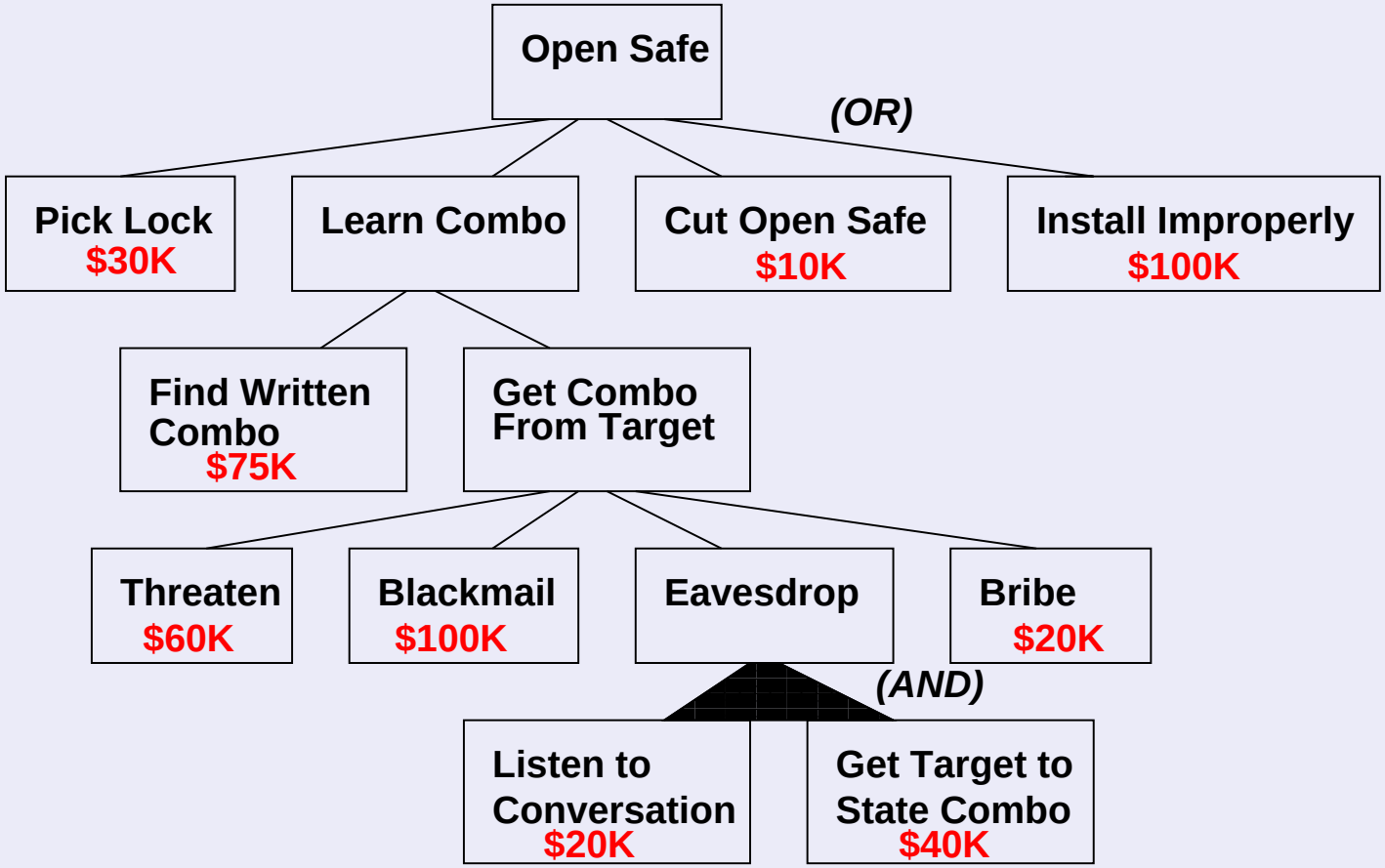
Attack Tree: Beispiel (Forts.)

(2) Berechnen der inneren Knoten



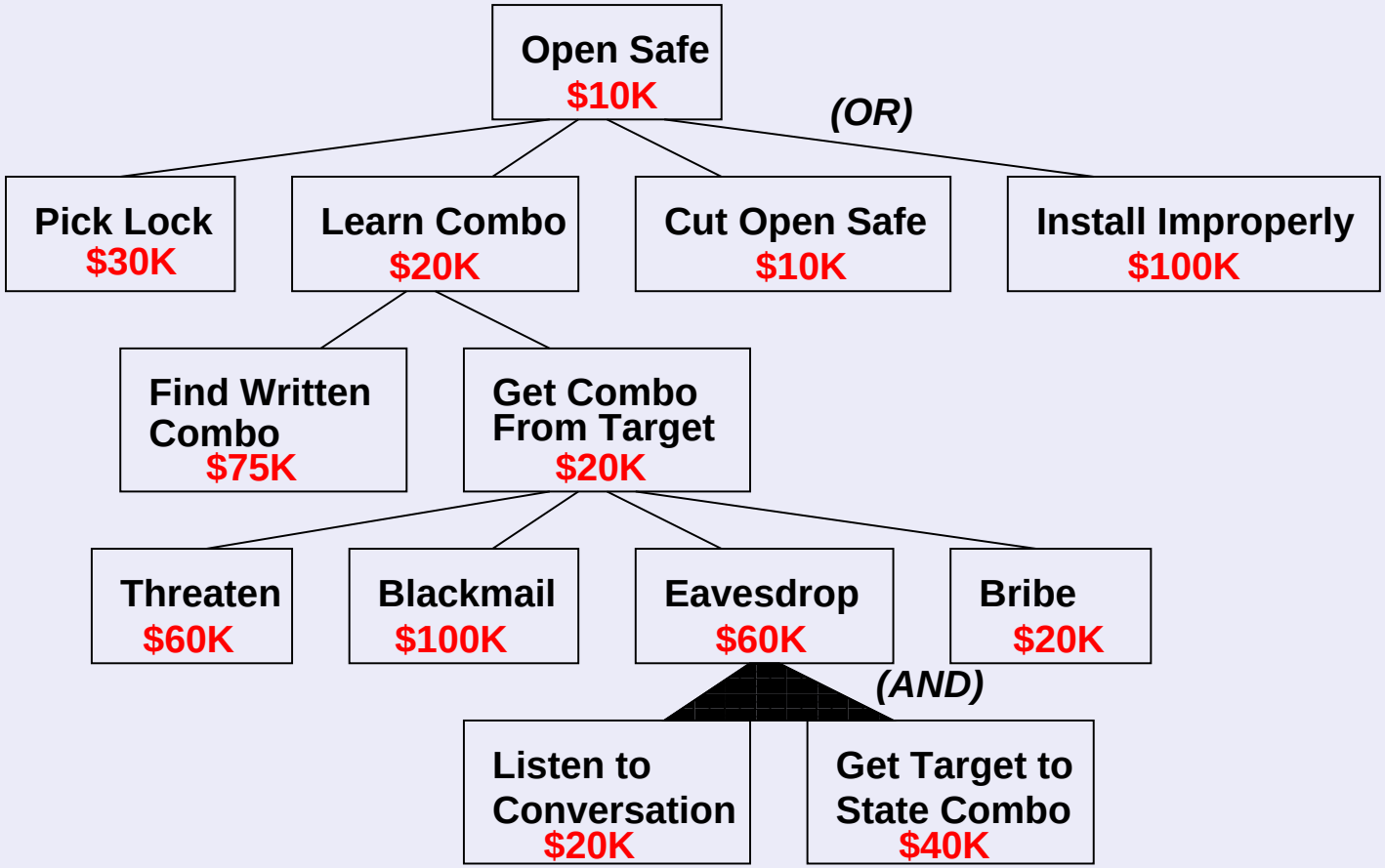
Attack Tree: Beispiel (Forts.)

Bewertung nach Kosten



Attack Tree: Beispiel (Forts.)

Bewertung nach Kosten (Forts.)



Attack Trees – Bewertung

Positiv

- übersichtliche Darstellung, Tool-Unterstützung
- erweiterbar
- wiederverwendbare Building-Blocks möglich
- gezieltes Bestimmen von Abwehrmaßnahmen

Problematisch

- Ist der Attack Tree vollständig?
- Ist die Baumstruktur ideal geeignet?
- Wie bewertet man realistisch die Knoten?
- Bewertungen abhängig vom Einsatzgebiet
  $\Rightarrow$ Bewertungen nicht direkt wiederverwendbar!

Abschließende Betrachtung

Wie ist Sicherheit zu erreichen?

- Technische Maßnahmen schützen nur vor bekannten Angriffen!

  $\Rightarrow$ Technik alleine ungenügend
- Zusätzliche organisatorische und juristische Maßnahmen sind daher unverzichtbar.

  $\Rightarrow$ *Wie wirken sich diese auf den Attack-Tree aus?*

  $\Rightarrow$ *Geht es nicht ganz ohne technische Maßnahmen?*

100-%ige Sicherheit?

„The only truly secure system is one that is powered off, cast in block of concrete and sealed in a lead-lined room with armed guards – and even then I have my doubts.“

(Gene Spafford, 1989)

Klassifikation von Systemen – Orange Book

- *Trusted Computer System Evaluation Criteria* (TCSEC); DoD, USA 1985
- Teil der *Rainbow-Series* (über Bände)
- Bewertung nach eingesetzten Verfahren **und** interner Realisierung

Kategorien

- Sicherheits-Policy
- Zugang zum System
- Zusicherungen (Vertrauen)
- Dokumentation

Sicherheitsstufen

A Überprüfter Schutz

B Erzwungener Schutz (B3 – B1)

C Wahlfreier Schutz (C2, C1)

D Minimale Sicherheit

Klassifikation von Systemen – Orange Book (Forts.)

Minimale Sicherheit (D)

- ausgewertete Systeme, keine festellbare Sicherheit

Wahlfreier Schutz (C1)

- disjunkte Adressräume
- einfache Zugangskontrolle
- freiwilliger Zugriffsschutz

Kontroll. Zugriffsschutz (C2)

- Access Control Lists
- physikalisches Löschen
- System-Protokollierung

Erzwungener Schutz (B3, B2, B1)

- vorgeschriebener Zugriffsschutz (MAC)
- modulare Systemarchitektur
- formales Modell der Sicherheit

Überprüfter Schutz (A1)

- formaler Beweis der gewährleisteten Sicherheit

Frage: *Wie würden jeweils Windows, Linux und Mac-OS eingestuft?*

Objektwiederverwendung

Um was geht es?

- Schutz gelöschter Objekte vor erneutem Zugriff (ab Klasse C2)
  - Dateien
  - Hauptspeicherinhalte
  - beliebige Ressourcen

1. Beispiel: Datei im Netzwerk drucken

- geheime Daten auf Festplatte
- gewählte Option: Löschen nach Druck
- kein physikalisches Löschen der Spooling-Datei

$\Rightarrow$ Daten u. U. restaurierbar

2. Beispiel: Benutzerkennung

- Benutzerkennung mit UID 007 wird gelöscht
- System vergibt UID 007 erneut

$\Rightarrow$ noch eingetragene Zugriffsrechte werden vererbt!

Objektwiederverwendung (Forts.)

Methoden

- Löschen von Hauptspeicherseiten sobald wie möglich;
  spätestens jedoch vor Zuteilung an anderen Prozess
- Sofortiges Löschen im Programm, z. B. Passwort-Puffer nach Verifikation
- Löschen des Festplattenspeichers, wenn eine Datei gelöscht wird
- Löschen des Swap-Bereichs, wenn ein Prozess beendet wird.
- Festspeicher physikalisch löschen (z.B. Linux: *wipe, shred, dd*, Mac: *rm –P*)
  $\Rightarrow$ mehrfach mit unterschiedlichen Bit-Mustern überschreiben???

  Nach heutigen Erkenntnissen genügt einfaches Überschreiben (Wright et al., *Overwriting Hard Drive Data: The Great Wiping Controversy*, ICISS 2008):
  `dd if=/dev/zero of=/dev/sda bs=1MB conv=noerror`
- Dateisystem mit starker Verschlüsselung verwenden (bei SSDs unbedingt!)
  auch Backups verschlüsseln!
- Entmagnetisieren bzw. Zerstören von nicht mehr benötigten Datenträgern
  (Bänder, Disketten, CDs, DVDs, USB-Disks, unverschlüsselte SSDs, ...)
- Löschen gepufferter Daten aus entf. Systemen (Druck-Server, ...)

Vertrauenswürdiger Pfad

Um was geht es?

Problem insb. bei Mehrbenutzersystemen:

- Auf dem Bildschirm wird die gewohnte Anmeldemaske angezeigt
- Aber Vorsicht:
  Stammt die wirklich vom Anmeldeprozess des Systems?

$\Rightarrow$ Eingabe von Kennung und Passwort ist kritisch!

Vertrauenswürdiger Pfad (Forts.)

Abhilfe

- Etablieren einer vertrauenswürdige Kommunikation zwischen Benutzer und System (ab Klasse B2):
  - Automatisches Zurücksetzen in einen wohldefinierten Zustand (nur Systemprozesse)
  - Starten des vertrauenswürdigen Anmeldeprozesses
- Unterstützung durch Hardware und/oder Betriebssystem erforderlich
  $\Rightarrow$ Wie denn??

Frage: *Geht das auch bei Ihrem PC?*

Überwachung (*Auditing*)

Um was geht es?

- Protokollieren der Systemereignisse:
  - Benutzeran- und -abmeldungen
  - Dateizugriffe (öffnen, schließen, umbenennen und löschen)
  - entfernte Systemzugriffe

Ziele der Protokollierung

- Rekonstruktion eines Einbruches
  - Wer ist der Einbrecher?
  - Was wurde verändert?
- Erstellen von Beweismaterial
- Input für Intrusion Detection Systeme (IDS)

Überwachung (Forts.)

Anforderungen des *Orange Book* (ab Klasse C2)

- Pro Ereignis Protokollierung von:
  - Datum und Zeit
  - UID des Benutzers
  - Art des Ereignisses
  - Erfolg oder Misserfolg der Aktion
  - Ort der Aktivität (z.B. Name des Terminals, IP-Adresse)
  - Name des Objekts (z.B. der geänderten Datei)
  - Beschreibung der Änderungen in der Sicherheitsdatenbank
  - Einstufung des Benutzers (ab Klasse B1)

Dabei beachten

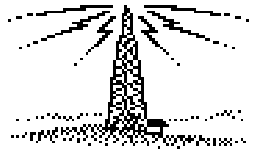
- zuverlässige und persistente Speicherung
- einfache und effiziente Auswertbarkeit der Protokolldaten

Systemarchitektur

Anforderungen des *Orange Book*

- Trennung von Benutzer und System
  (auch schon in der Hardware: User / System Mode der CPU)
- Adressraumtrennung (ab Klasse C1)
- Isolation der Sicherheitsfunktionen
- Modularer Aufbau der sicherheitsrelevanten Komponenten
- Prinzip des geringsten Privilegs (ab Klasse B2):
  Prozess hat immer nur soviele Rechte wie gerade benötigt
- Systemverwalterfunktion über mehrere Accounts / Personen aufteilen
  $\Rightarrow$ Ziel: Keiner hat alleine die komplette Kontrolle
- Hierarchische Schichtung des Systems (Klassen B3 und A1):
  Kommunikation nur über klar definierte Schnittstellen

Verdeckte Kanäle (*Covert Channels*)



Um was geht es?

- Informationsübermittlung findet im Verborgenen über einen unkonventionellen und daher unbeobachteten Kommunikationskanal statt

Verdeckter Timing-Kanal

- Beeinflussung der Systemleistung, gezielt wechselnde Auslastung von Systemressourcen
- *Orange Book*: verhindern oder erkennen ab Klasse B3
- Beispiele: wiederholtes Starten eines Programms, belegen / freigeben des Druckers, anlegen / löschen einer Datei, ...
- I. Allg. sehr geringe effektive Datenraten (nur wenige bps)

Verdeckte Kanäle (Forts.)

Verdeckter Speicherkanal

- Punktuelle Veränderung gespeicherter Daten
- *Orange Book*: verhindern oder erkennen ab Klasse B2
- Plumpes Beispiel: Dateiname einer Benutzerdatei:
  `README` $\Rightarrow$ `README.WASTI_XYZ123`
- Cleverer: Ändern von Metadaten wie z. B. Dateiattribute (Änderungsdatum oder Zugriffsrechte)
- Äußerst perfide: Steganographie
  - Verstecken von Informationen, meist in Multimedia-Dateien
    (auch: digitale Wasserzeichen $\Rightarrow$ DRM)
- I. Allg. eher geringe effektive Datenraten ($\leq$ 10 Kbps)

Verdeckte Kanäle (Forts.)

(Quelle: http://www.petitcolas.net/steganography/image_downgrading/, 2016*)*

Verdeckte Kanäle (Forts.)

(Quelle: http://www.petitcolas.net/steganography/image_downgrading/, 2016*)*

Verdeckte Kanäle (Forts.)

(Quelle: http://www.petitcolas.net/steganography/image_downgrading/, 2016*)*

Vertrauenswürdige Distribution

Um was geht es?

- sichere Auslieferung mittels vollständig geschütztem Transportweg
- *Orange Book*: Klasse A1
- Methoden:
  - Kommunikation: detaill. Vorabinformation, was genau geliefert werden soll
  - schützende Verpackung für Hard- und Software sowie Dokumentation
    $\Rightarrow$ Schlösser, Siegel, Schrumpfschlauch, elektron. Sicherung
  - verschlüsselte Auslieferung von Software
  - ggf. bewaffnete, vertrauenswürdige Kuriere

... und schließlich ...

Dokumentation

- äußerst umfangreich
  Umfang wächst rapide mit der Evaluationsklasse
  - Benutzerhandbuch der Sicherheitsfunktionen
  - Systemadminstratorhandbuch
  - Testdokumentation
  - Entwurfsdokumentation

Andere Klassifikationssysteme

Green Book

- seit 1988 in Deutschland

ITSEC (IT-Evaluationshandbuch)

- seit 1991 in Europa
- Sicherheitsklassen: E1 bis E6

Common Criteria (CC)

- seit 1996 international (USA, EU, Can, AUS, NZ, ISR)
- *Evaluation Assurance Levels*: EAL1 bis EAL7
  (in der Praxis meistens max. erreichbar: EAL4)
- derzeit wichtigstes Evaluationsschema
- vgl. http://www.commoncriteriaportal.org/

Schutzmechanismen im Betriebssystem

Benutzerverwaltung

- Zugangskontrolle sichert authentisierte Benutzer zu

Dateisystem

- logische Zugriffskontrolle: durch Rechteverwaltung
- direkte Zugriffskontrolle: durch Verschlüsselung
- physikalisches Löschen

Hauptspeicherverwaltung

- Adressraumtrennung (Hardware-abhängig)
- physikalisches Löschen aufgegebener Objekte

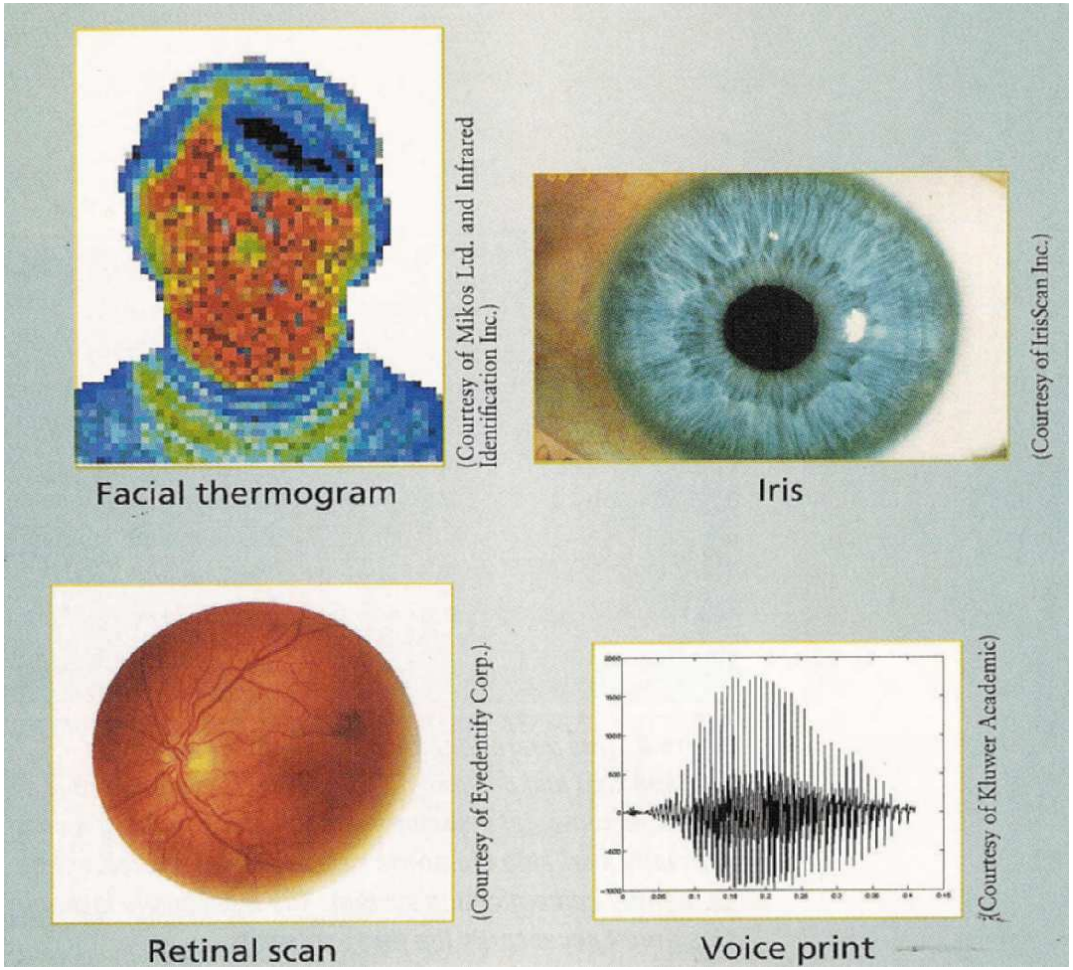
Benutzerauthentisierung

Möglichkeiten

- durch Besitz
  (Türschlüssel, Chipkarte, ...)
- durch biometrische Merkmale
- durch Wissen
  (PIN, Passwort, Algorithmus, ...)

$\Rightarrow$ **kombinieren!** (z.B. *two-factor authentication*)

Biometrische Authentisierung



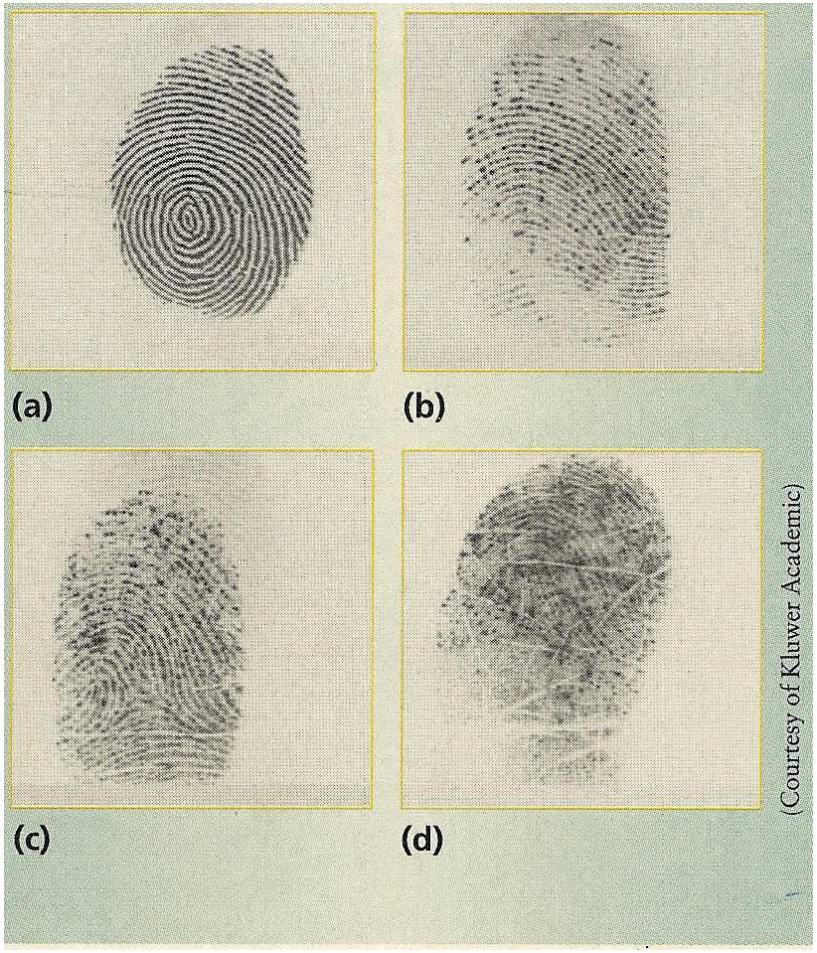
(Quelle: IEEE Spektrum)

Biometrische Authentisierung (Forts.)

Eindeutige körperliche Merkmale

- Fingerabdruck
- Stimme
- Handgeometrie
- Handschrift (Unterschrift)
- Bild des Gesichts
- Bild der Iris
- Bild der Netzhaut (Retina)
- DNS
- ...

Biometrische Authentisierung (Forts.)



(Quelle: IEEE Spektrum)

Biometrische Authentisierung (Forts.)

Phasen

- Enrollment (kann scheitern!)
- Authentisierung

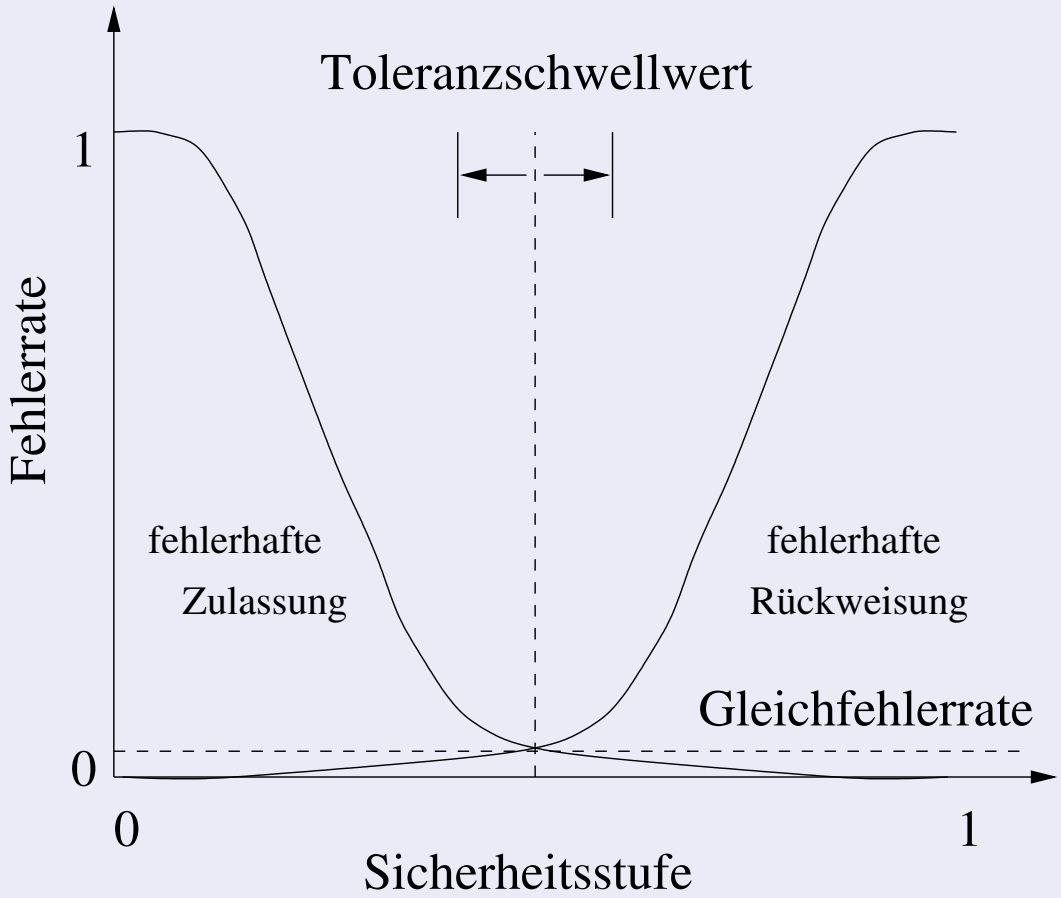
Wichtige Kennwerte

- Erkennungszeit
- Akzeptanz
- *False Rejection Rate* (FRR): Anteil fehlerhafter Rückweisungen
- *False Acceptance Rate* (FAR): Anteil fehlerhafter Zulassungen

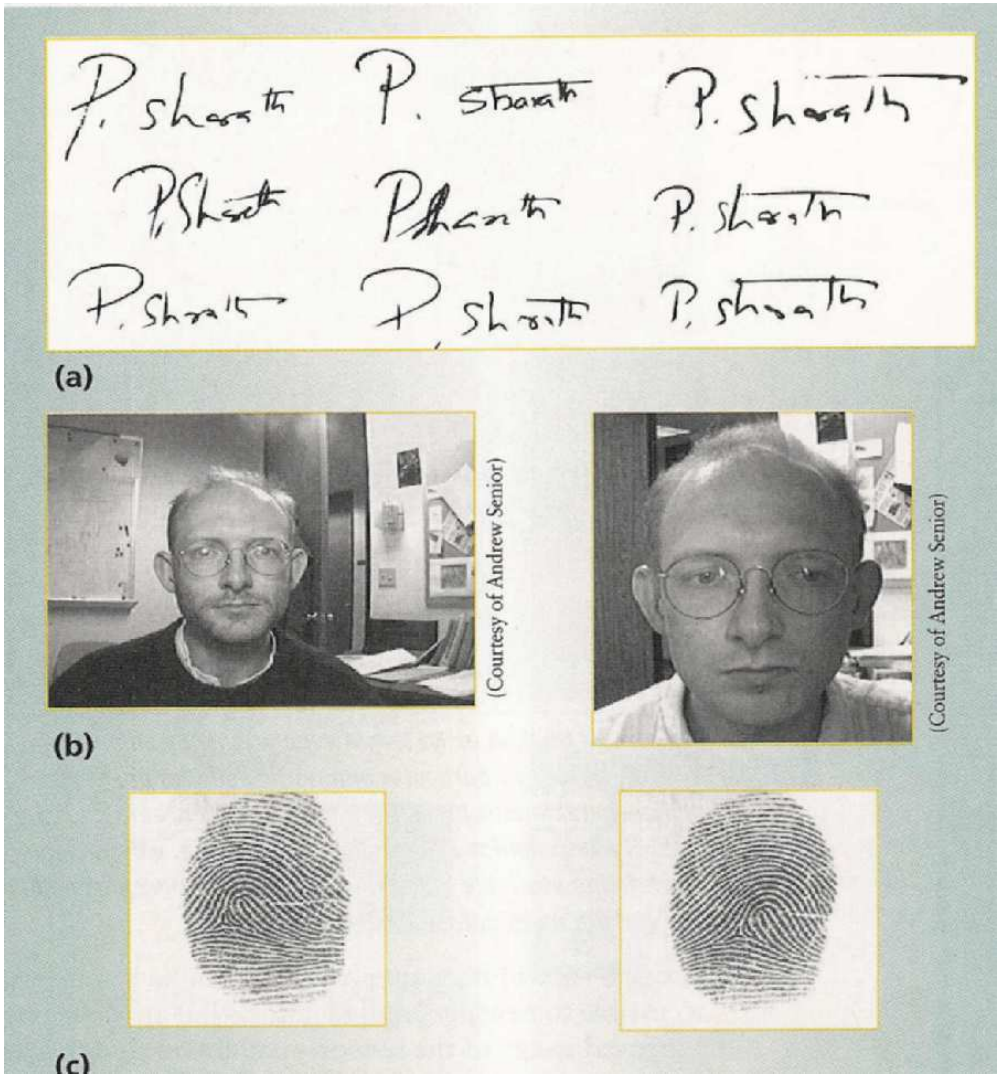
Biometrische Authentisierung (Forts.)

Zentrale Problematik

- Aufgenommene Muster bei jeder Erhebung anders!
  ⇒ Toleranzen einplanen:



Biometrische Authentisierung (Forts.)



(Quelle: IEEE Spektrum)

Biometrische Authentisierung (Forts.)

Angriffe

- Replay-Angriff auf den Sensor
  $\Rightarrow$ Tonband, Foto, ...
- Replay-Angriff auf die Entscheidungssoftware
  $\Rightarrow$ Einspielen eines alten Musters hinter dem Sensor

  z.B. mittels Angriff die Referenz-Datenbank oder die Übertragung
  Problem: i.Allg. nicht rückrufbare Authentisierungsmuster

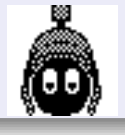
Fazit

- Biometrische Verfahren nur für Authentisierung „vor Ort“ geeignet!

Authentisierung mittels Passwort

Schlechte Passwörter

- Wort mit Bezug zur Person (Geburtsdatum, Telefonnr., ...)
- Namen (Vornamen, Nachnamen, Städtenamen, ...)
- Wort aus Wörterbuch (D, GB, F, ...)
  (auch rückwärts oder mit vorangestellter Ziffer)
- sind zu kurz
- enthalten z.B. nur Kleinbuchstaben oder nur Großbuchstaben
  (Achtung: Nicht immer sind alle Zeichen zulässig ...)
- muss man aufschreiben, um sie nicht zu vergessen.
- sind nicht flink einzutippen!



Passwortwahl

Pwds cracked from a sample set of 13 797 accounts			
Dictionary	Size	Matches	Total
/usr/dict/words	19 683	1 027	7.4%
Common names	2 239	548	4.0%
User/account name	130(P)	368	2.7%
Phrases and patterns	933	253	1.8%
Female names	4 280	161	1.2%
Male names	2 866	140	1.0%
Machine names	9 018	132	1.0%
Uncommon names	4 955	130	0.9%
King James bible	7 525	83	0.6%
Place names	628	82	0.6%
Myths & legends	1 246	66	0.5%
Science Fiction	691	59	0.4%
Chinese	392	56	0.4%
Famous people	290	55	0.4%
Sports terms	238	32	0.2%
Character sequences	866	22	0.2%
Asteroids	2 407	19	0.1%
Movies and actors	99	12	0.1%
Numbers	427	9	0.1%
Total			23.7%

Authentisierung mittels Passwort (Forts.)

Beliebteste Passwörter im Jahr 2010

1. 123456
2. password
3. 12345678
4. qwerty
5. abc123
6. 12345
7. monkey
8. 111111
9. consumer
10. letmein
11. 1234
12. dragon
13. trustno1
14. baseball
15. gizmodo
16. whatever
17. superman
18. 1234567
19. sunshine
20. iloveyou
21. fuckyou
22. starwars
23. shadow
24. princess
25. cheese

Quelle: https://duo.com/blog/brief-analysis-of-the-gawker-password-dump

Authentisierung mittels Passwort (Forts.)

Gibt es genug sichere Passwörter?

- PIN (vierstellig): 10.000
- 26 Kleinbuchstaben:
  $26^8 = 2 \cdot 10^{11}$ untersch. Passwörter mit 8 Zeichen
- $2 \cdot 26$ Buchstaben + 10 Ziffern + 30 Satzzeichen
  $\Rightarrow$ über 90 unterschiedliche Zeichen:
  $\geq 90^8 = 4.3 \cdot 10^{15}$ Passwörter mit 8 Zeichen
- Passwort-Entropie (in Bit) $\boldsymbol{H} = \boldsymbol{L} \cdot \log_2 \boldsymbol{N}$

  (mit Passwortlänge $\boldsymbol{L}$ und Größe des Zeichenvorrats $\boldsymbol{N}$)

  Voraussetzung: an jeder Stelle wird zufällig und gleichwahrscheinlich ein Zeichen aus dem Vorrat gewählt
- Sichere Passwörter haben H$\geq$100 Bit (BSI-Empfehlung von 2015).

  Unterschiedliche Empfehlungen je nach Verwendungszweck (WLAN, Chipkarte, ...)

Authentisierung mittels Passwort (Forts.)

Wie bilde ich ein gutes Passwort?

- Passphrase bzw. Verkettung kurzer Wörter mittels Zahl oder Satzzeichen
  $\Rightarrow$ `1SmartRobot4my_KidsIs100%expensive`
  (ausreichende Länge beachten!)
- Verändern eines Wortes in mehreren Stellen
  $\Rightarrow$ Ma2#Heim (aus Mannheim)
- Bilden von Akronymen
  $\Rightarrow$ `eEi1MmeB`
  (Von: „Ein Elefant ist eine Maus mit einem Betriebssystem“)
  Aber: Wie bekommt man hier Satzzeichen / Ziffern?

$\Rightarrow$ Ziel ist möglichst hohe Entropie $\Rightarrow$ Passwortgenerator ?!
(Aber man muss es sich merken!)

$\Rightarrow$ neue Passwörter mehrmals übungsweise tippen!
$\Rightarrow$ Passwort-Safe verwenden!

Unix-Passwörter

Die Datei `/etc/passwd` (":" ist Trennzeichen)

bastl:uHui19CB5aVu.:4806:2002:Bastl Mueller:/users/bastl:/bin/bash
ferdi:Pe/SsRCDX9YP2:4063:2002:Ferdi Mueller:/users/ferdi:/bin/bash
karl:Sd3rnzkdYqRbQ:2066:2005:Karl Mueller:/users/karl:/bin/csh
fred:PhP125um2mYMw:2280:2001:Fred Mueller:/users/fred:/bin/csh
walter:MMiemzpi7RjOs:8099:2003:Walter Mueller:/users/walter:/bin/bash
hubert:a1CqCAQP26iYM:5281:2002:Hubert Mueller:/users/hubert:/bin/bash
thomas:FREE;:8072:2008:Thomas Mueller:/tmp:/etc/logoff

Achtung: Diese Datei ist öffentlich lesbar!!!

Frage: *Dürfen die Passwörter hier denn stehen?*

Unix-Passwörter (Forts.)

Elemente einer Zeile

(1) Benutzerkennung (Login-Name)

(2) verschlüsseltes Passwort (**Frage:** mit welchem Schlüssel?)

(3) eindeutige Benutzernummer (UID)

(4) Benutzergruppennummer (GID)

(5) Benutzername (*finger*-Name)

(6) Login-Verzeichnis (*Home*)

(7) zu startendes Programm (i. Allg. Login-Shell)

Unix-Passwörter (Forts.)

Die *crypt()*-Einwegfunktion (ursprünglich in Unix verwendet)

- Passwort wird mit dem Programm *crypt()* verschlüsselt und abgespeichert ($\Rightarrow$ Passwort-Hash).
- *crypt()* ist eine Einwegfunktion basierend auf DES:
  Passwort wird als DES-Schlüssel eingesetzt:
  Block mit Null-Bits mit Passwort 25-mal DES-verschlüsseln.

- `login`-Programm verschlüsselt eingegebenes Passwort und vergleicht dies mit abgespeichertem Eintrag.

Frage: *Wie lang kann ein Unix-Passwort hier maximal sein?*

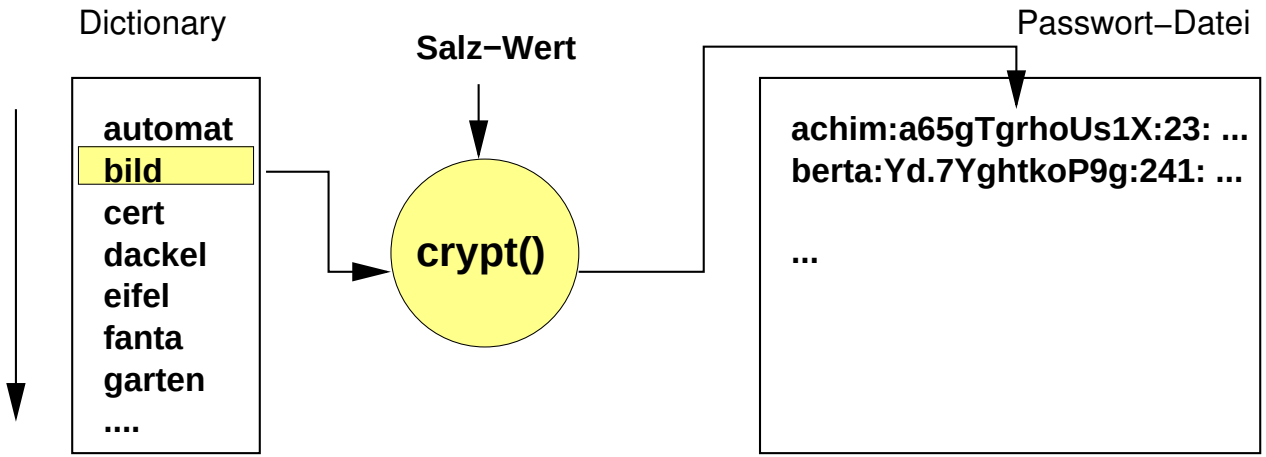
Wörterbuch-Angriff (*dictionary attack*)

Ein sehr effizienter Angriff

(1) Kopieren der Passwort-Datei auf lokale Maschine

(2) Durchführung des eigentlichen Angriffs Offline:

- (2a) Verschlüsseln des ersten Wörterbucheintrags
- (2b) Vergleichen des Ergebnisses mit allen (!) Passwörtern
- (2c) Testen des nächsten Wörterbucheintrags, usw.



Unix-Passwörter (Forts.)

Darstellung der Passwörter

- Resultat von *crypt* wird als ASCII-Zeichenkette abgespeichert
- *Base-64-Encoding*: (6 Bit pro Zeichen):

6-Bit-Wert	ASCII-Zeichen
0	.
1	/
2	0
⋮	⋮
11	9
12	A
⋮	⋮
37	Z
38	a
⋮	⋮
63	z

Frage: *Wieso stehen 13 Zeichen im Passwortfeld der* /etc/passwd*?*

Unix-Passwörter (Forts.)

Eine prima Idee: Salz (*Salt*)

- Beim Generieren des Passwortes zufällig gewählte 12-Bit Zahl
  $\Rightarrow$ jedes Passwort hat 4096 mögliche Verschlüsselungen
- ist Teil des Passworteintrags in /etc/passwd! (**Frage:** Wieso???)
- Wirkung:
  - Behinderung von schnellen DES-Hardware-Implementierungen
  - gleiche Passwörter haben unterschiedliche Darstellungen
  - Konstruktion einer „Verschlüsselungstabelle“ ist aufwändiger
    (400 GB statt 100 MB; seit einigen Jahren aber möglich)
  - Behinderung von Wörterbuch-Angriffen
  - Behinderung von sogenannten Rainbow-Table-Angriffen

Unix-Passwörter (Forts.)

Sicherheit von *crypt()*

- Maximale Dauer eines Brute-Force-Angriffs (bei L=8) (Annahme: 6000 Mio. Tests pro Sekunde).:
  - PWD mit 90 versch. Zeichen: $\approx$ 8,3 Tage
  - PWD mit 64 versch. Zeichen: $\approx$ 13 Stunden
  - PWD aus Wörterbuch: 0,016 Sekunden (!)

$\Rightarrow$ sehr geringe Sicherheit, inakzeptabel bei schwachen Passwörtern

- Passwortlisten für crypt im Internet verfügbar, sowie Einsatz von GPUs und FPGAs

  $\Rightarrow$ sichere Verschlüsselungsfunktion mit längeren Schlüsseln verwenden

Maßnahmen zur Erhöhung der Passwortsicherheit

Reaktion auf Authentisierungsfehler

- nach x-maliger falscher Eingabe die Benutzerkennung sperren
  $\Rightarrow$ **Gute Idee ???**
- nach jeder falschen Eingabe (wachsende) Wartezeit einlegen
- Passwort-Aging $\Rightarrow$ **Vor- / Nachteile ???**

Crack-Programme als Admin ausführen

- regelmäßiges Testen der Benutzer-Passworte
  $\Rightarrow$ aktuelle, umfangreiche Wörterbücher!
  $\Rightarrow$ Software aus vertrauenswürdiger Quelle!
  $\Rightarrow$ Problem: Hacker-Paragraph?!!

Maßnahmen zur Erhöhung der Passwortsicherheit (Forts.)

Shadow Passwords

- Passwörter getrennt von Benutzerinformationen speichern:
  Datei `/etc/shadow`
- Datei nur System-lesbar (`root`)
  $\Rightarrow$ Offline Dictionary-Attacke unmöglich!

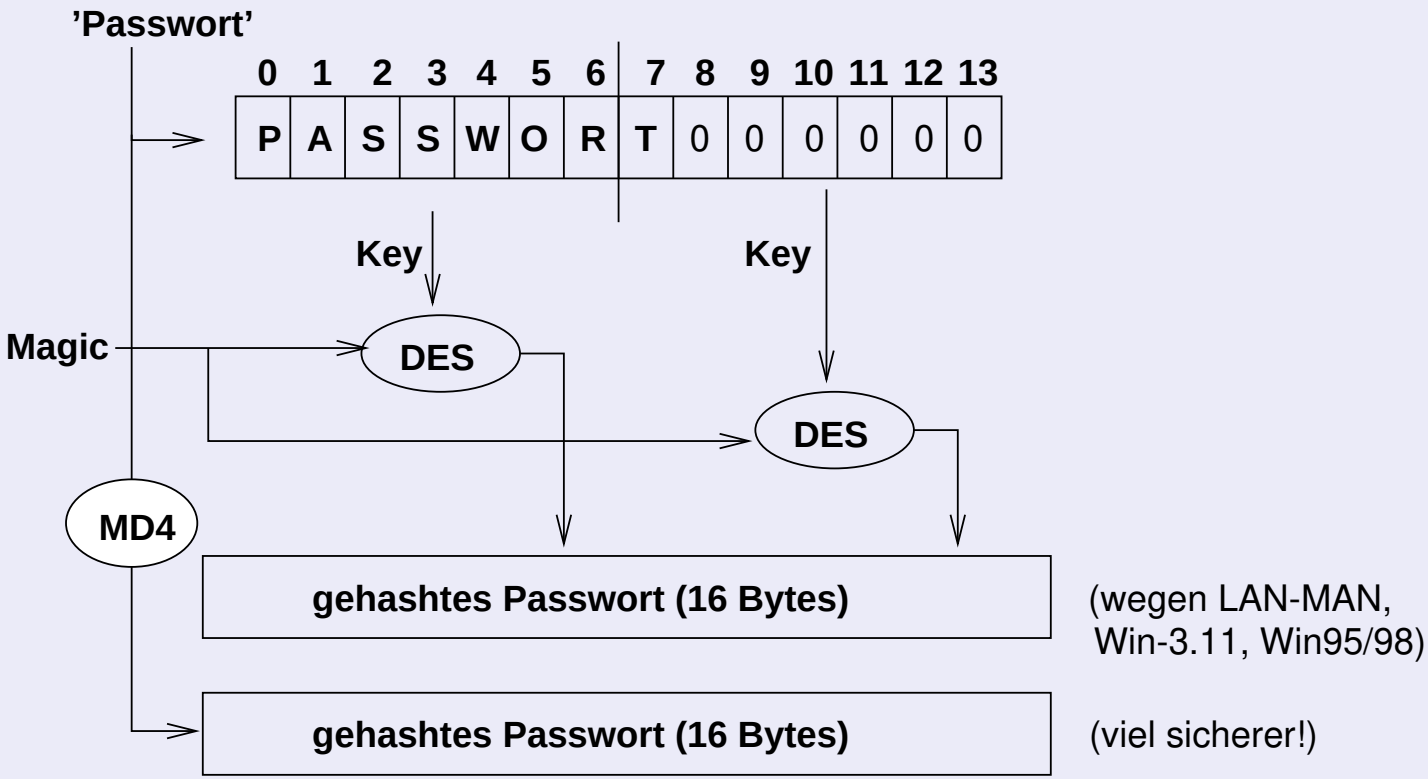
Längere Passwörter

- moderne Passwortverschlüsselungsfunktion nutzen:
  z. B. Argon2 (basiert auf AES), bcrypt (basiert auf Blowfish),
  scrypt, yescrypt (basiert auf SHA-256), Pufferfish, ...

Frage: *Wieviel sicherer ist das in der Praxis?*

Verschlüsselung d. Windows-Passwörter (ab Win-NT)

Einwegfunktion (LM-Hash bzw. NTLM-Hash)



Verschlüsselung d. Windows-Passwörter (ab NT) (Forts.)

Wo abgespeichert?

- Shadow Passwörter: nur lesbar von Systemkennung:
  `%SYSTEMROOT%\system32\config\sam`
- Im Netz an mehreren Stellen mit meist schwächerem Schutz gespeichert

Bewertung:

- LM-Hash (DES-Methode):
  Verfahren schwächer als *crypt()* ($\Rightarrow$ **Wieso???**)
  seit Vista standardmäßig deaktiviert
- NTLM-Hash (mit MD4 bzw. ab NTLMv2 MD5):
  + Unicode-Strings werden verarbeitet
  – ebenfalls kein individuelles Salt (KGS!@#$%)
  $\Rightarrow$ zahlreiche Tools: l0phtCrack, LC5, RainbowCrack, hashcat, Cain, ...

Benutzerverwaltung (Unix)

Rechte-Prüfung in Unix

- relevant: UID und GID (jeweils 16 Bit)
- Abbildung: Kennung zu UID (i. Allg. 1:1-Abbildung)
- definiert in `/etc/passwd`
- Abbildung: Kennung zu GID (i. Allg. n:m-Abbildung)

Wer bin ich eigentlich?

```
> login emueller
emueller> id
      uid=2008(emueller)
      gid=10(staf)
emueller> groups
      staf wheel
emueller>
```

Benutzerverwaltung (Forts.)

UNIX: Hauptgruppe

- definiert in `/etc/passwd`

Zusätzliche Gruppen definiert in: `/etc/group`

```
root:*:0:root
wheel:*:0:root,emueller
news:*:6:
staf:*:10:emueller
shadow:*:15:root
gast:*:2003:fmueller,gmueller
```

Auch: `/etc/gshadow`

Benutzerverwaltung (Forts.)

Besondere Benutzerkennungen in Unix

- `root` $\Rightarrow$ *Superuser* (System); `UID = 0`
- `nobody`, `www` $\Rightarrow$ Benutzer ohne Dateien; unprivilegierte Operationen, z.B. im WWW
- `ftp`, für anonyme ftp-Zugriffe
- `guest`, für Besucher (kritisch!)
- ...

für Passwörter manchmal noch Standard-Vorbelegungen
$\Rightarrow$ Unbedingt ändern!!!

Benutzerverwaltung (Forts.)

Wechseln zwischen Benutzerkennungen

- Substitute User (su-Befehl)

```
emueller> su fmueller
      Password:
      su: Sorry
emueller>
```

- $\Rightarrow$ meistens Wechsel zu `root`:

```
emueller> su
      Password:

root> su fmueller
fmueller>
```

- Simuliertes Login (inkl. Setzen der Umgebung):

```
> su -
```

Benutzerverwaltung (Forts.)

root darf alles

1. Signalisieren beliebiger Prozesse (z.B. `KILL`)
2. An- und Abschalten der Protokollierung
3. beliebiges Ändern der UID bzw. GID seiner Prozesse
4. Ausführen ALLER Befehle (z.B. shutdown)
5. Setzen der Systemuhr
6. Lesen und Verändern beliebiger Speicherzellen
7. Lesen, Ändern und Löschen beliebiger Dateien
8. Einrichten und Löschen von Benutzerkennungen
9. Untersuchen aller Datenpakete im Netzwerk
10. Neukompilieren **aller** Programme

Ein typischer Angriff

Ablauf

1. Einbruch in das System
2. Erzeugen eines Superuser-Prozesses
3. Beseitigen aller Spuren

4. Installation einer *Back Door* und eines *Rootkit*

⇒ **Frage:** Was will der Angreifer von meinem Rechner?

Mögliche Schutzvorkehrungen gegen Back Doors

- Unlöschbare Protokollierung (auf Drucker o.ä.)
- Integritätskontrolle der Systemdateien (z.B. *Tripwire*)
- Physikalisch schreibgeschütztes Dateisystem für Systemdateien
- Intrusion Detection (War ein Angriff? Welcher Angriff?)
- Intrusion Prevention (Firewall-Systeme)

Ein typischer Angriff (Forts.)

Rootkits

- Verstecken der Backdoor sowie anderer Spuren des erfolgreichen Angriffs mittels Veränderung wichtiger Systemdateien und / oder Teile des Kernels (insb. System Calls und Interrupt-Handler)

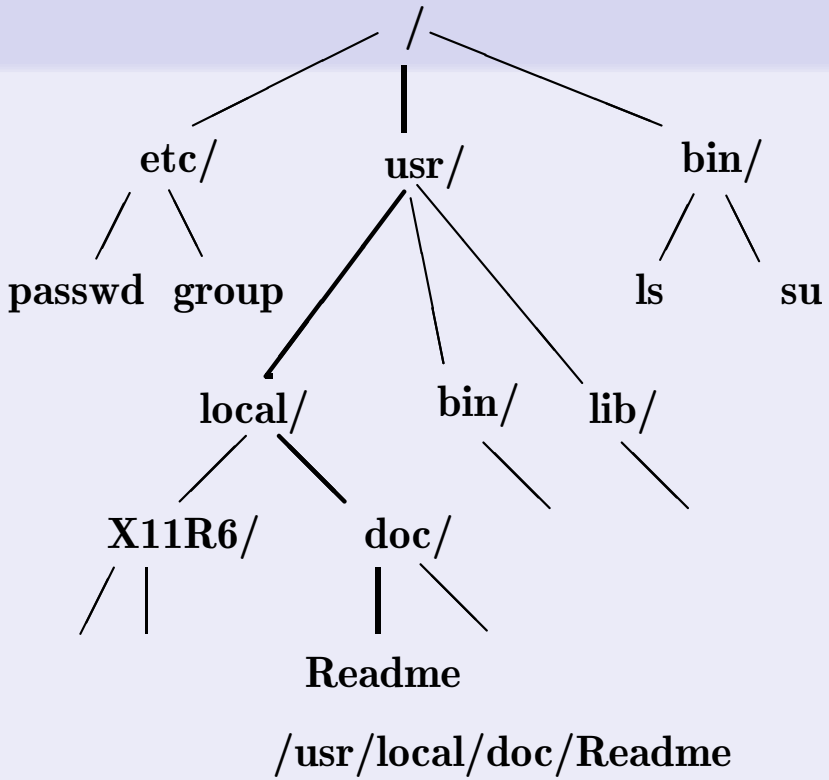
Varianten

- Application-level / User-level Rootkits
- Kernel-level Rootkits
- Hauptspeicher Rootkits
- VM-based:
  Installation einer virtuellen Maschine (z.B. Blue Pill, 2006)

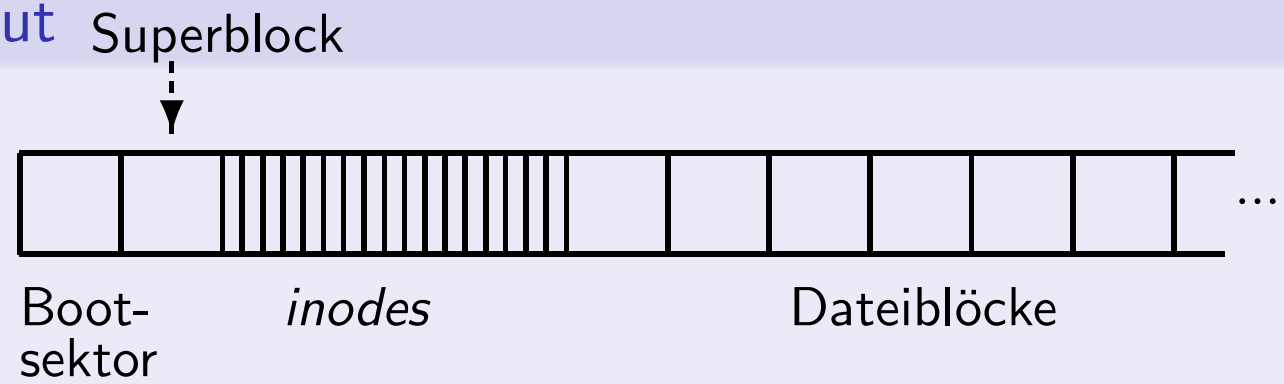
Frage: *Wie bemerke ich ein solches Rootkit?*

Das Unix-Dateisystem

Dateibaum

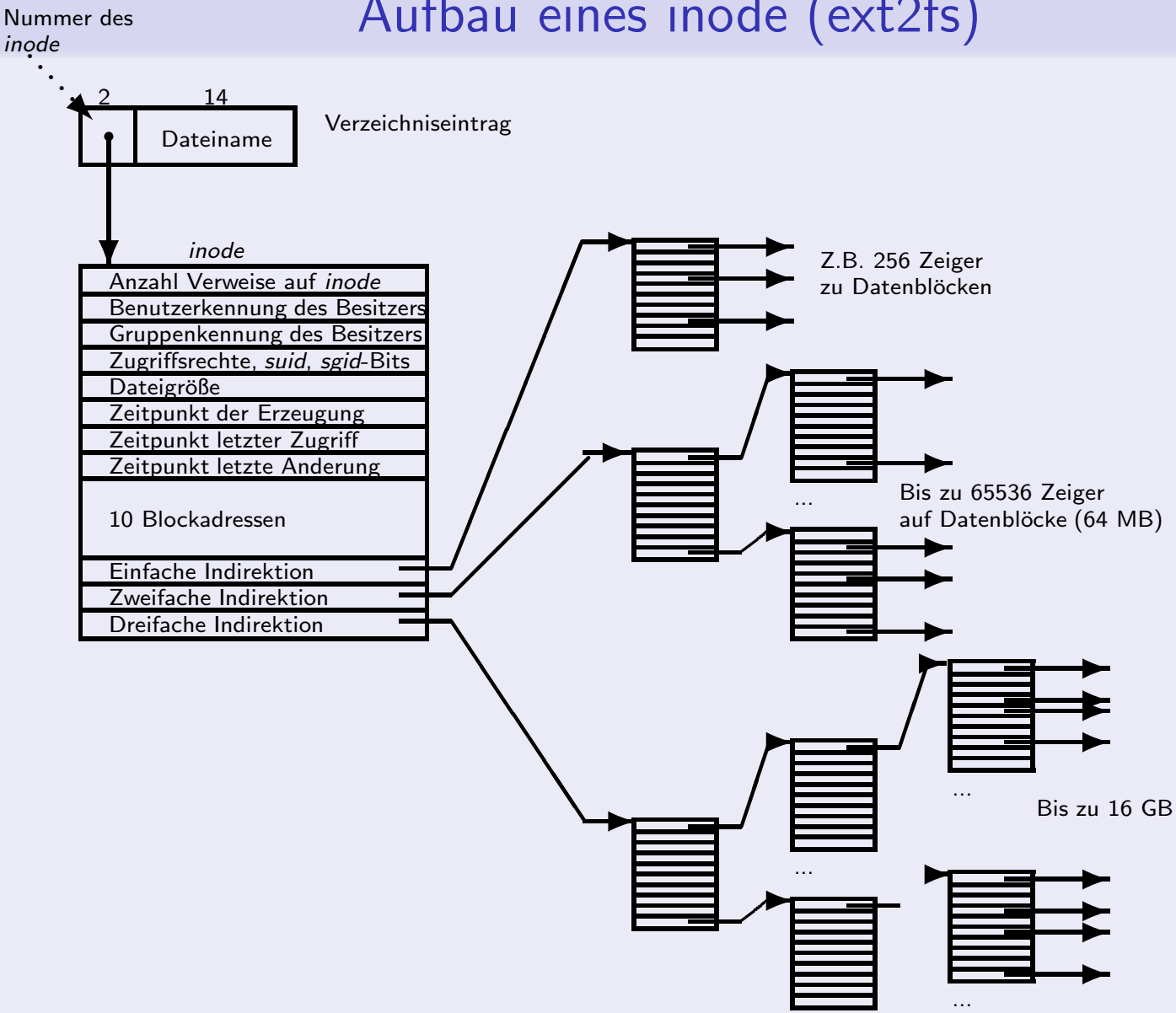


Plattenlayout



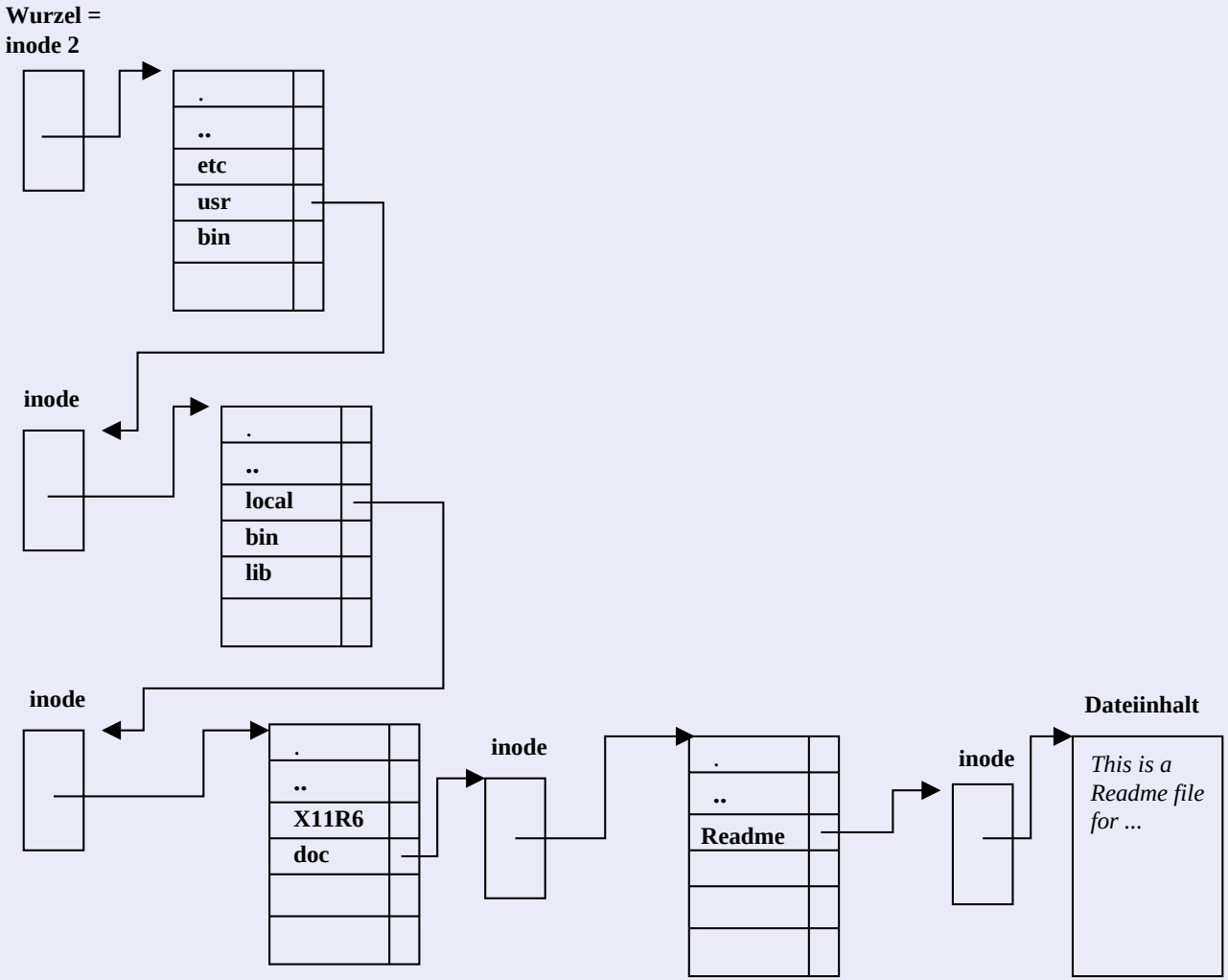
Das Unix-Dateisystem (Forts.)

Aufbau eines inode (ext2fs)



Das Unix-Dateisystem (Forts.)

Zugriff auf Datei /usr/local/doc/Readme



Unix-Zugriffsschutz: Dateischutzbits

Granularität

- Benutzer (`u`)
- Gruppe (`g`)
- alle anderen (`o`)

Rechte

- lesen (`r`)
- schreiben (`w`)
- ausführen (`x`)

Bedeutung bei Verzeichnissen

- `r` $\Rightarrow$ Verzeichnisinhalt auflisten (`ls`)
- `w` $\Rightarrow$ Dateien anlegen / löschen / umbenennen (`touch, rm, mv`)
- `x` $\Rightarrow$ Verzeichnis betreten (`cd`)

Unix-Zugriffsschutz (Forts.)

Beispiel

```
> ls -l
total 846
drwxr-xr-x   pagnia   mitarb   2560     May 14 14:36   Bilder
-rw-r--r--   pagnia   mitarb   76762    Jan 20 17:22   folien.tex
-r--------   pagnia   mitarb   820876   Jan 20 17:22   backup.tgz
-rwx------   pagnia   mitarb   46       Apr 23 11:40   publish
```

Frage: *Darf ich eigentlich mein eigenes Passwort ändern?*

Unix-Zugriffsschutz (Forts.)

SUID- und *SGID*-Bit

- Jeder Prozess hat *reale* UID und *effektive* UID.
- Das gesetzte *SUID*-Bit bei einer AUSFÜHRBAREN Datei ändert die effektive UID jedes ausführenden Prozesses!
  $\Rightarrow$ Prozess agiert im Folgenden mit den Rechten des Dateibesitzers
- *SGID*-Bit analog für effektive GID
- Beispiel:
  das Programm `/bin/passwd` (Verändern des Passworteintrags)

```
-rwsr-xr-x   root   root   /bin/passwd
-rw-r--r--   root   root   /etc/passwd
```

Prozess benötigt zum Schreiben *root*-Rechte!

Unix-Zugriffsschutz (Forts.)

Unix-Zugriffsberechtigungen (oktal)

Rechte werden addiert:

Oktalwert	Bedeutung
4000	*suid*-Bit setzen
2000	*sgid*-Bit setzen
1000	*sticky*-Bit setzen ($\Rightarrow$ auch Mandatory Locking)
0400	Leserecht für Besitzer
0200	Schreibrecht für Besitzer
0100	Ausführungsrecht für Besitzer
0040	Leserecht für Gruppe
0020	Schreibrecht für Gruppe
0010	Ausführungsrecht für Gruppe
0004	Leserecht für restliche Benutzer
0002	Schreibrecht für restliche Benutzer
0001	Ausführungsrecht für restliche Benutzer

Unix-Zugriffsschutz (Forts.)

Setzen der Zugriffsrechte

```
> ls -l it_klausur
     -rw-r----- pagnia mitarb 567 it_klausur


> chmod 646 it_klausur
> ls -l it_klausur
     -rw-r--rw- pagnia mitarb 567 it_klausur


> chmod go-rwx it_klausur
> ls -l it_klausur
     -rw------- pagnia mitarb 567 it_klausur
```

Unix-Zugriffsschutz (Forts.)

Das `umask`-Kommando

- automatische Vorbelegung der Zugriffsrechte für neue Dateien (Angabe oktal)
- jeder Prozess kann `umask` setzen
- *umask* spezifiziert, welche Zugriffsrechte **nicht** gesetzt werden
- Zugriffsrechte := Standardwert `AND NOT` *umask*
  (Standardwert = 666 bzw. für Programme 777)
- am besten: Setzen im Profile
- Beipiel:
  `umask 077` $\Rightarrow$ ausschließlich Dateibesitzer hat Zugriff

Allmächtige Systemverwalter

Ich mach' nur eben mal schnell ...

- Problem:
  Schreiben in fremdes Verzeichnis $\Rightarrow$ permission denied
- Rechte des Unterbaums auf 777 setzen
- `cd` *`directory`*
- dann `su` zu `root` mit: `su -`
- schließlich `chmod -R 777`
- ... dauert recht lange ... (nur 45 Dateien)
- Warum? `su -` wechselt in Homeverzeichnis!
- Hoffentlich gibt's ein Backup!

Allmächtige Systemverwalter (Forts.)

Einmal zu wenig nachgedacht!

- autom. Löschen alter Test-Accounts
- `su -`
  `rm -r ...`
- eine Kennung hatte / als Homeverzeichnis!
- $\Rightarrow$ vollständige Platte gelöscht

More systems have been wiped out by admins, than any hacker could do in a life time.

SUID- und *SGID*-Dateien

Sicherheitskritisch

- evtl. *Back Door*: Shell, Editor o.ä. mit SUID- oder SGID-Bit
- Schutz: Regelmäßiges Durchforsten des Dateisystems
- „Missbrauch“ von SUID- und SGID-Programmen möglich
  ($\Rightarrow$ Puffer-Überlauf)
- am besten *SUID root* vollständig vermeiden! (**Frage:** Wie??)
- zumindest *defensiv programmieren*!

SUID- und *SGID*-Dateien (Forts.)

Regeln zum sicheren Programmieren

1. sorgfältiges Design ($\Rightarrow$ Spezifikation, `man`-Pages)
2. überprüfen aller Argumente (Kommandozeilen-Parameter, Aufrufparameter für Unix-Systemfunktionen und Umgebungsvariablen)
3. Bereichsverletzungen vor dem Ausführen entdecken
4. Umgebungsvariablen löschen und neu setzen
5. keine Zeichenkettenfunktionen, die interne Puffergrenzen nicht checken `gets()`, `strcpy()` und `strcat()` in C
6. Rückgabewerte von Systemfunktionen überprüfen
7. Zusicherungen mittels `assert`-Makro
8. ausführliches Testen des neuen Programms
9. Fahndung nach *Race-Conditions* ($\Rightarrow$ keine SUID Kommandodateien)
10. keine Aufrufe von `system` und `popen` ($\Rightarrow$ erzeugen Shell)
11. keine öffentlich schreibbaren Verzeichnissen anlegen

Ein Insider-Angriff (*social attack*)

Wie werde ich *root*?

- Suchpfad des Systemverwalters:
  `PATH=.:/bin:/usr/bin:...`

- Angreifer erzeugt ausführbare Datei `ls`:

```
#!/bin/sh
cp /bin/sh ../misc/rootshell
chmod 4555 ../misc/rootshell
rm -f $0
exec /bin/ls $@
```

- Berechtigung des aktuellen Verzeichnis auf 700 setzen
- platzieren einer Datei mit `touch ./-f`
  (mit `rm` nicht ohne weiteres zu löschen)

Ein Insider-Angriff (*social attack*) (Forts.)

Systemverwalter will helfen

- keine Zugriffsrechte $\Rightarrow$ `su`
- `cd` *`directory`*
- `ls`
- `rm ./-f`
- $\Rightarrow$ Angreifer kann nun `root` werden

Frage: *Welche Fehler hat der Systemverwalter begangen?*

Ein Insider-Angriff (Forts.)

Moral

- Besser: `/bin/su -` verwenden
- Noch besser: unmittelbar anschließend `/bin/su -` *<newuser>* ausführen
- Vollständigen Dateinamen inkl. Pfad angeben (z.B. `/bin/ls`)
  $\Rightarrow$ vermeidet troj. Pferde
- `PATH=.:/bin/:...` möglichst nie verwenden
  keinesfalls jedoch für `root`!!!

Erweiterte Zugriffskontrolltechniken

Zugriffskontrollisten (ACLs)

- pro Datei: Liste mit Rechten der zugriffsberechtigten Benutzer

Beispiel:

Datei 'xlock':

Benutzer	Zugriffsrechte
fritz	R,W,X
chef	R,X
anna	R,W,D,X
⋮	⋮

('D' = Löschrecht)

Erweiterte Zugriffskontrolltechniken: ACLs (Forts.)

ACLs in Unix (hier: HP-UX)

- bis zu 16 Einträge
- Ableitungsreihenfolge nach Genauigkeit
  (1. Benutzer & Gruppe; 2. nur Benutzer; ...)
- Kommandos: `lsacl`, `chacl` (auch Kopieren von ACLs)

```
> lsacl -l notenliste
```

```
notenliste:
rw-   pagnia.%
r--   %.kurs
r--   albertini.mitarb
rw-   nagler.direktion
---   %.%
```

Zugriffskontrolle nach Bell-LaPadula

Bell-LaPadula Sicherheitsmodell

- regelbasierte Informationsflusskontrolle
- im Multics Betriebssystem, ca. 1968
- Definitionen:
  - Datei ($\Rightarrow$ Freigabe)
  - Benutzer ($\Rightarrow$ Freigabe)
  - Projekt (Menge von Benutzern + Dateien)

Beispiel für eine Klassenhierarchie:

```
unclassified   (0)
  < confidential (1)
    < secret       (2)
      < top secret (3)
```

Zugriffskontrolle nach Bell-LaPadula (Forts.)

Mandatory Access Control:

- Benutzer darf Datei lesen, falls
  (1) er Mitglied aller Projekte der Datei ist:
  $$\text{Projekte(user)} \supseteq \text{Projekte(file)}$$
  $\wedge$ (2) $\text{Freigabe(user)} \geq \text{Freigabe(file)}$

- Benutzer darf Datei schreiben, falls
  (1) sie in **allen** Projekten des Benutzers ist:
  $$\text{Projekte(user)} \subseteq \text{Projekte(file)}$$
  $\wedge$ (2) $\text{Freigabe(user)} \leq \text{Freigabe(file)}$

Zugriffskontrolle nach Bell-LaPadula (Forts.)

Beispiel

- Benutzer: A, B
  Dateien: u, v, w, x, y, z
- Freigabedefinitionen:

$$F(A) = F(u) = 0$$
$$F(B) = F(v) = F(y) = 1$$
$$F(w) = F(z) = 2$$
$$F(x) = 3$$

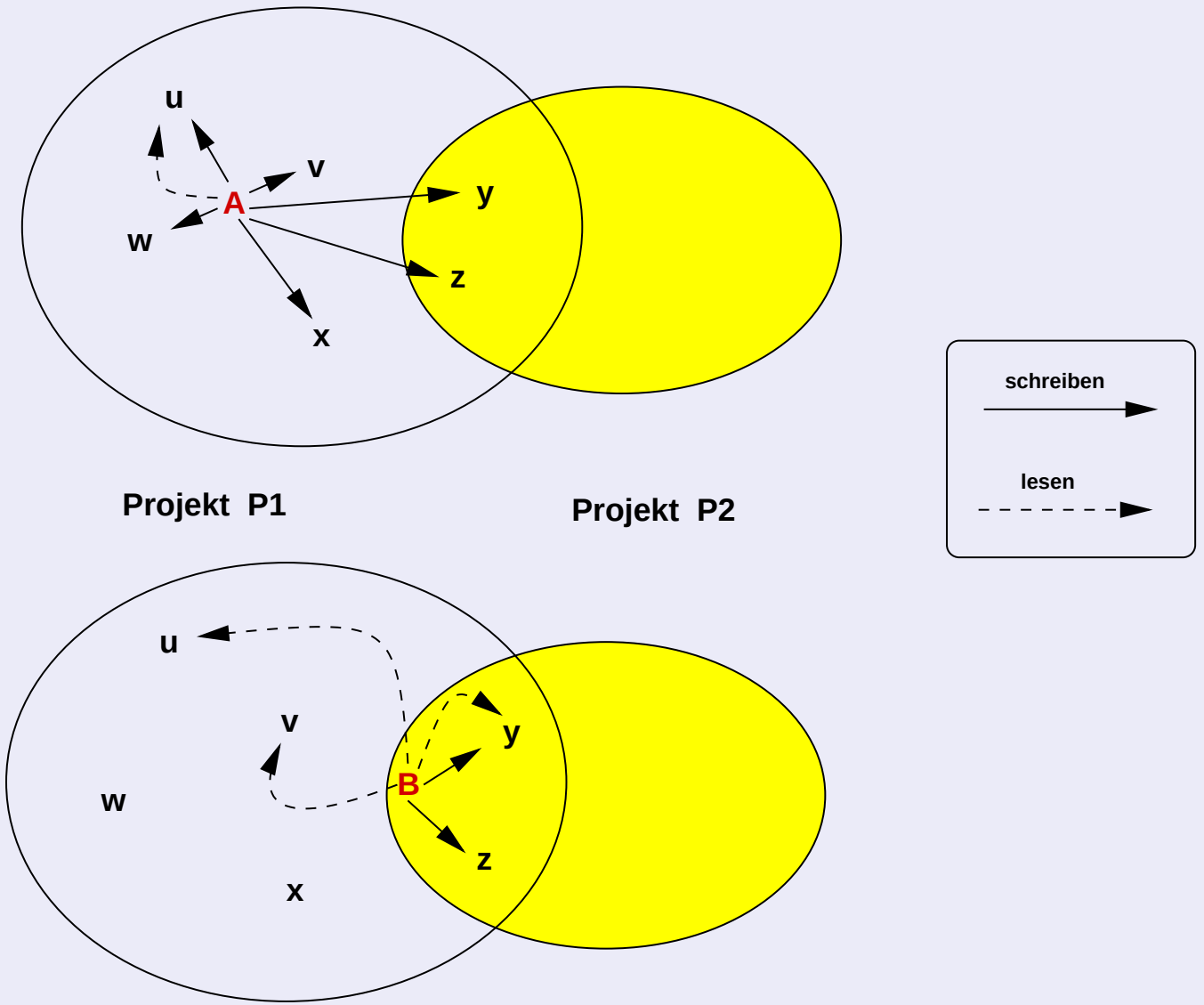
- Projektdefinitionen:

$$P(A) = P(u) = P(v) = P(w) = P(x) = \{P1\}$$
$$P(B) = P(y) = P(z) = \{P1, P2\}$$

Frage: *Welche Zugriffe sind erlaubt?*

Zugriffskontrolle nach Bell-LaPadula (Forts.)

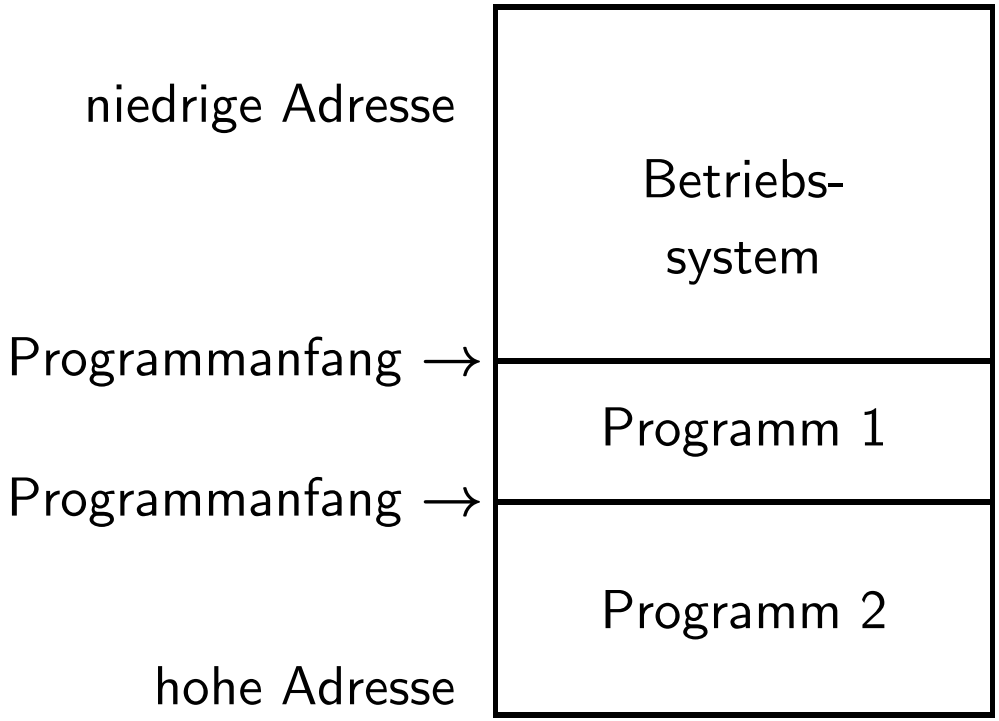
Erlaubte Zugriffe



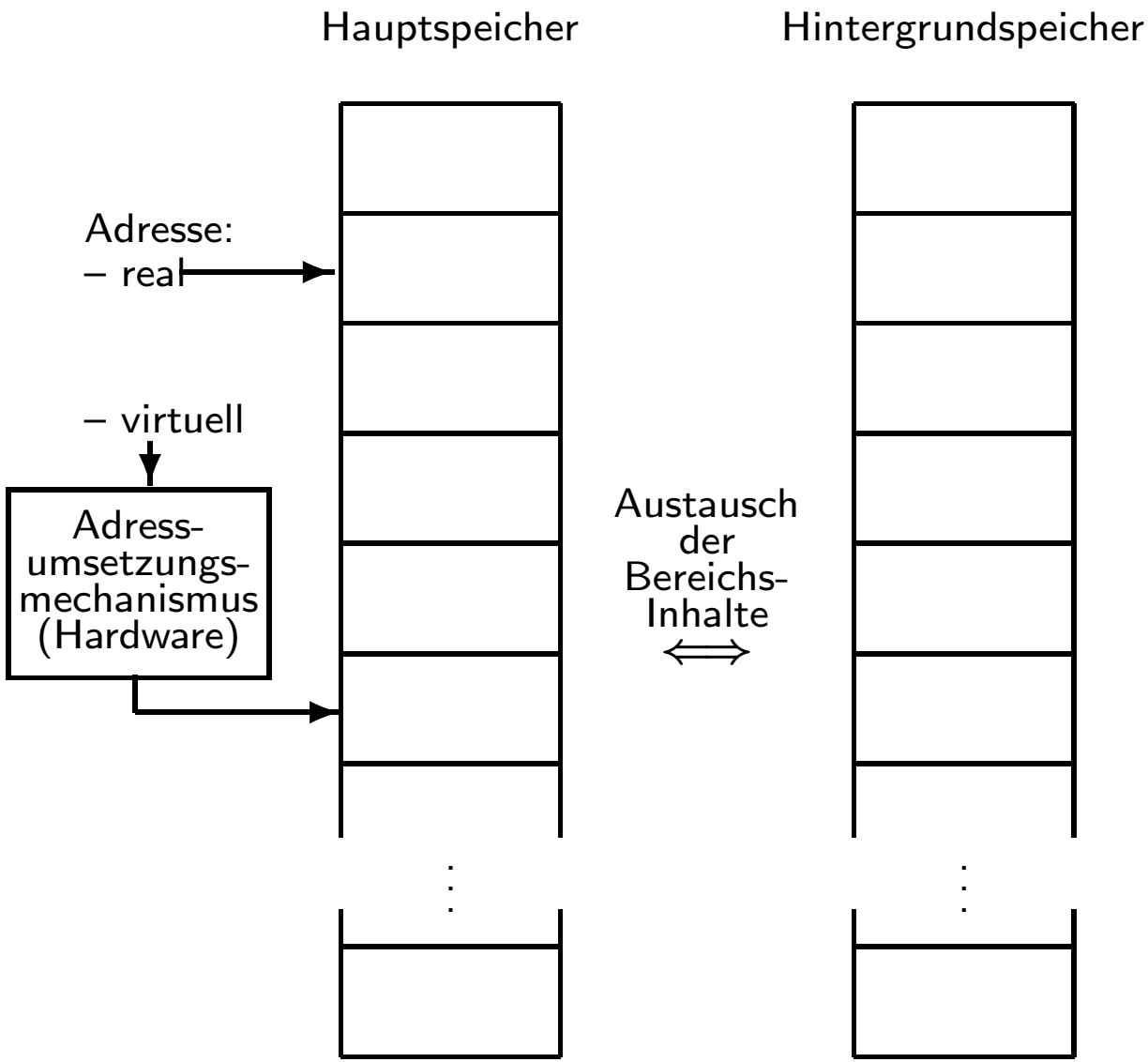
Speicherverwaltung

Multi-Programming

- gleichzeitig mehrere Programme im Speicher
- Adressierung relativ zum Programmamfang
- Schutz **aller** Programme untereinander
- Schutz des Betriebssystems (liegt im unteren Hauptspeicherbeich)



Speicherverwaltung: Virtueller Speicher



Speicherverwaltung: Virtueller Speicher (Forts.)

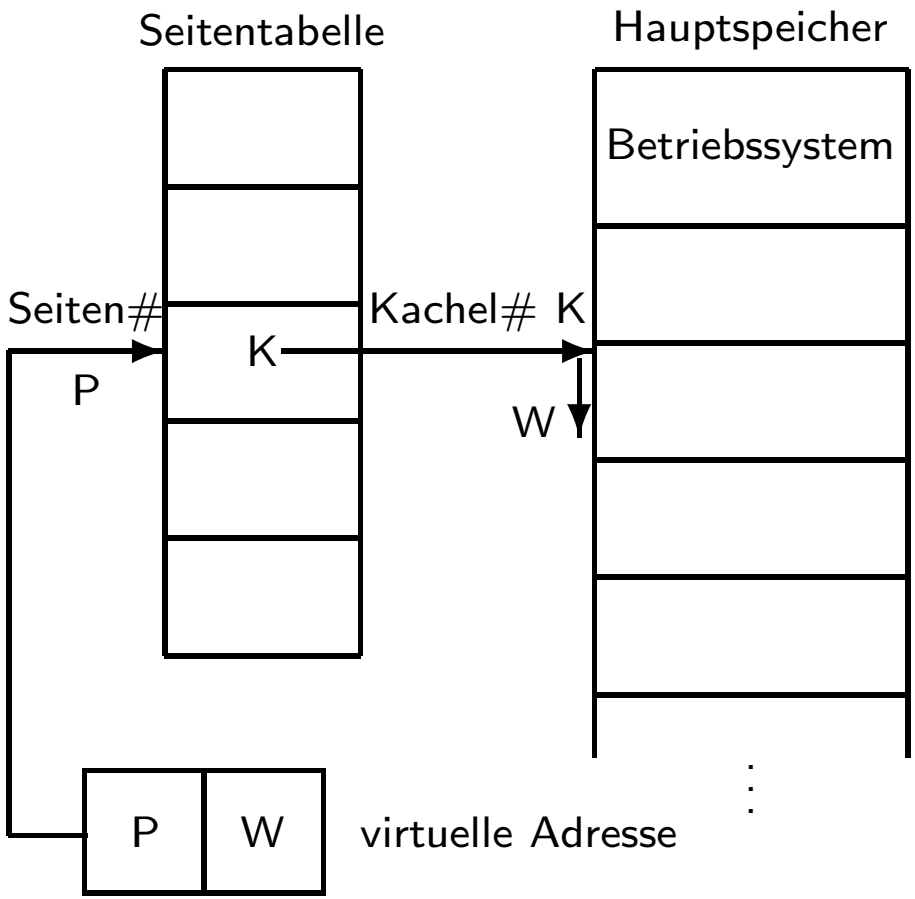
Paging

- Hintergrundspeicher ergänzt Hauptspeicher
- größerer Adressraum
- Aufteilen des Speichers in *Seiten*
  $\Rightarrow$ einige Seiten abwesend
- ggf. wird Unterbrechung erzeugt
- Unterbrechungsbehandlung vom Betriebssystem

Speicherverwaltung: Virtueller Speicher (Forts.)

Adressumsetzung

- virtuelle Adresse $\Rightarrow$ reale Hauptspeicheradresse
- Seitentabelle

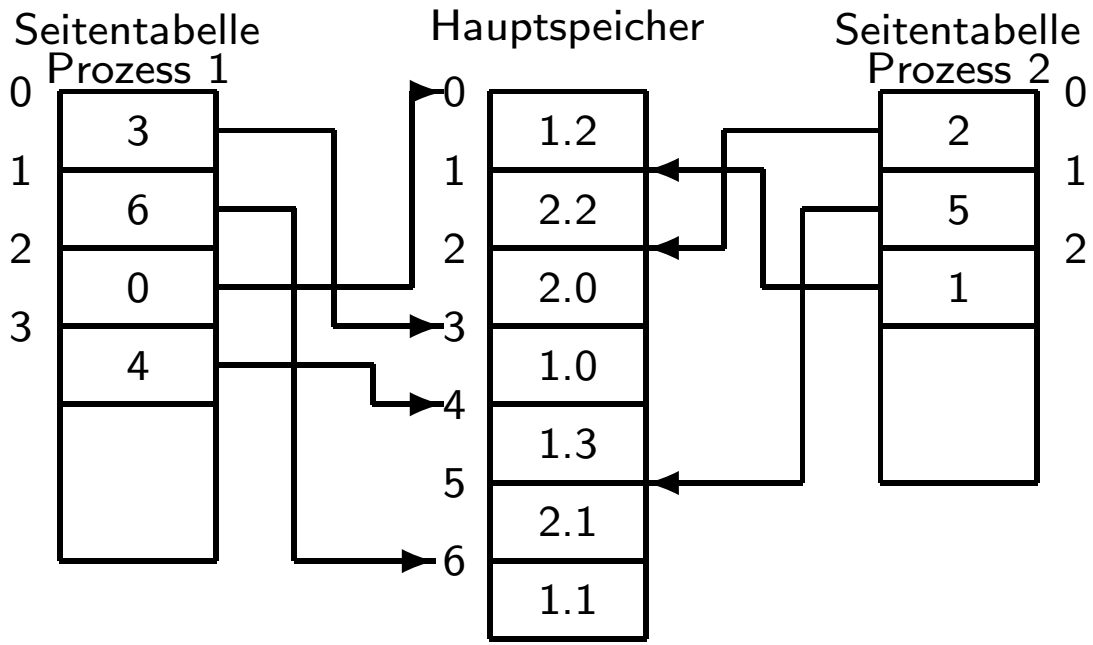


Speicherverwaltung: Virtueller Speicher (Forts.)

Disjunkte Adressräume

- eigene Seitentabellen (für Benutzer und System) pro Prozess
- Seitentabellenregister bei jeder Prozessumschaltung laden

$\Rightarrow$ nur Adressierung im eigenen Adressraum möglich



Speicherverwaltung: Virtueller Speicher (Forts.)

Probleme

- Auffinden von Seiten (liegen verstreut)

- Seite präsent oder abwesend?
- ...

  $\Rightarrow$ Seitentabelleneintrag:

K	P-Bit	C-Bit	R-Bit	W-Bit	X-Bit	···

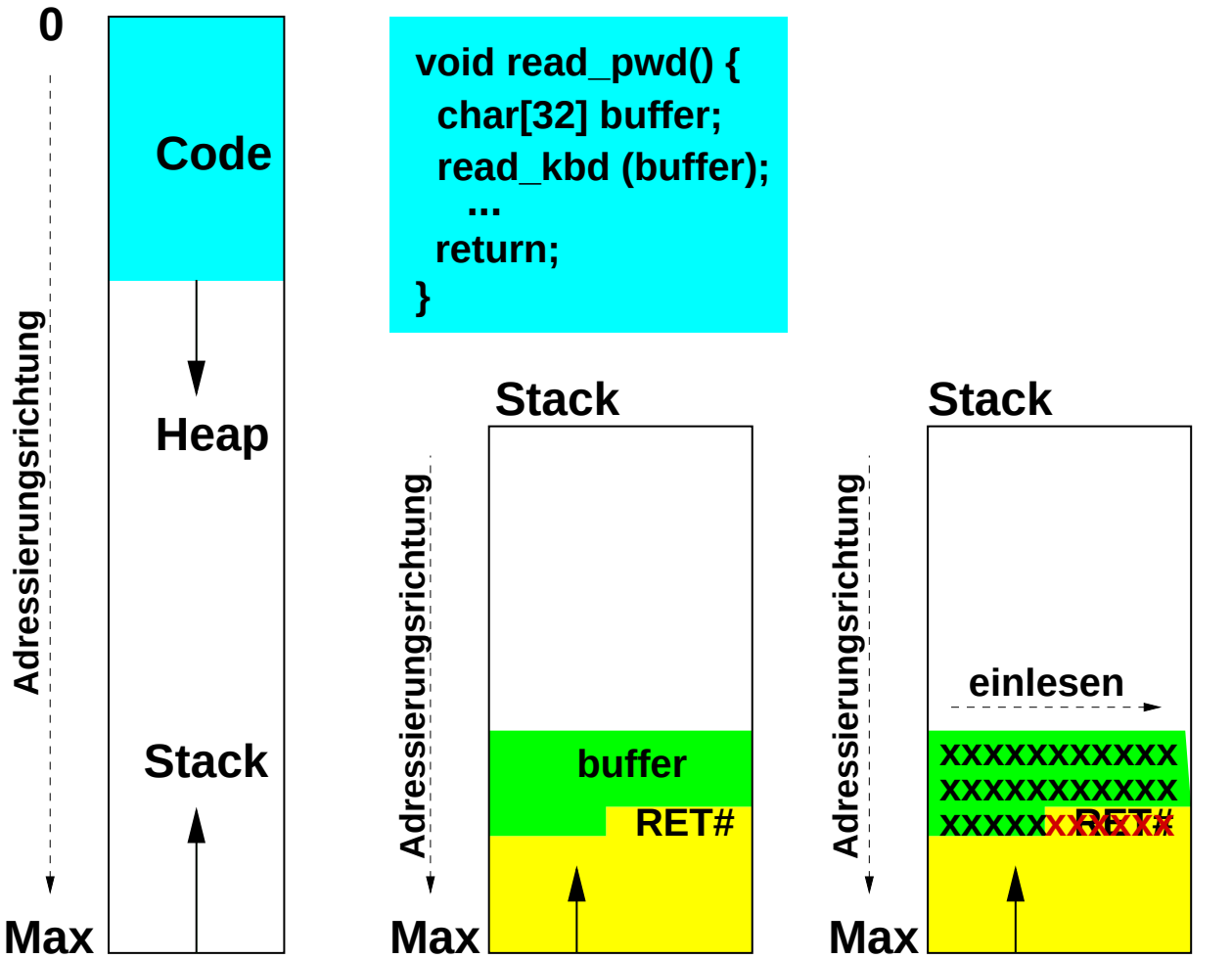
K: Kachelnummer
P-Bit: *Presence*-Bit, Präsenzbit
C-Bit: *Change*-Bit, Änderungsbit, *modified*-Bit
R-Bit: *Read*-Bit, Lesebit
W-Bit: *Write*-Bit, Schreibbit
X-Bit: *Execute*-Bit, Ausführbit

Speicherverwaltung: Virtueller Speicher (Forts.)

Schutzbits der Seitentabelleneinträge

- können verhindern, dass
  - Code zur Laufzeit überschrieben wird
  - Konstanten nachträglich geändert werden
  - Daten als Code ausgeführt werden
- Insbesondere ältere Betriebssysteme verwenden noch kein X-Bit
- Zum Teil müssen Schutzmechanismen von der Applikation explizit genutzt werden
  $\Rightarrow$ *Buffer-Overflow*-Angriff immer noch möglich!

Buffer-Overflow-Angriff (schematisch)



(a) Speicheraufteilung (b) Prozeduraufruf (c) Buffer Overflow

return → Wohin???

Buffer-Overflow-Angriff (Forts.)

Kritisch

- Beschreiben von Stack-Bereichen durch gefährliche Befehlssequenzen, die sodann angesprungen und ausgeführt werden.
  (**Frage:** Mit welchen Rechten ???)
- Realisierung des X-Bit in modernen Prozessoren
  (z. B. Intel: XD-Bit, AMD: NX-Bit, Sun SPARC, PowerPC)
- Berücksichtigung in modernen Betriebssystemen:
  u. a. Linux, Windows (seit XP SP-2: Data Execution Prevention DEP)

- Generische Abwehr im Betriebssystem auch ohne X-Bit möglich:
  u. a. OpenWall-Patch bzw. exec-Shield bei Linux, Software-DEP in Windows

Buffer-Overflow-Angriff (Forts.)

Variante

- Alternativ kann der Angreifer die Rücksprungadresse so verändern, dass Code-Stücke im Kernel oder in Bibliotheken angesprungen werden.
- Abwehr mittels ASLR (Address Space Layout Randomization)
  - zufällige Adresswahl für wichtige Strukturen:
    Bibliotheken, Stack, Heap, Prozess- / Thread-Kontrollblöcken
  - u. a. Windows (ab Vista), Linux, OpenBSD

  $\Rightarrow$ **Wieder problematisch:**
  Viele Sicherheitsfunktionen müssen durch Anwendungen explizit unterstützt werden!

Entwicklung des Internet

ARPANET

- Konzeption nach militärischen Aspekten
- u.a. fehlertolerantes Netzwerk:
  Ausfall beliebiger Rechner verkraften
- sehr abgeschlossener Personenkreis
- vertrauenswürdige Benutzer
- kompetente Benutzer

Entwicklung des Internet (Forts.)

Zivile Nutzung

- Zu Beginn der 80er Jahre auch beschränkter Zugang für Universitäten und Forschungsinstitute
- i. Allg. abgeschlossener, bekannter Personenkreis
- meist kompetente Benutzer
- heterogene Hardware- und Betriebssystem-Architekturen
- Regulierung durch Einhalten der Netiquette ...

Entwicklung des Internet (Forts.)

NSFNET $\Rightarrow$ Internet

- Ende 80er Jahre Nachfolger des ARPANET
- offener Personenkreis
- Kompetenz der Benutzer fragwürdig
- enorme Zuwachsraten
  $\Rightarrow$ TCP/IP „de-facto Standard“ für Netzwerke
- immer stärkere kommerzielle Nutzung
- enorme Sicherheitsprobleme
- Anonymität der Benutzer $\Rightarrow$ erwünscht oder unerwünscht?
- rechtliche Regelungen?

Entwicklung des Internet (Forts.)

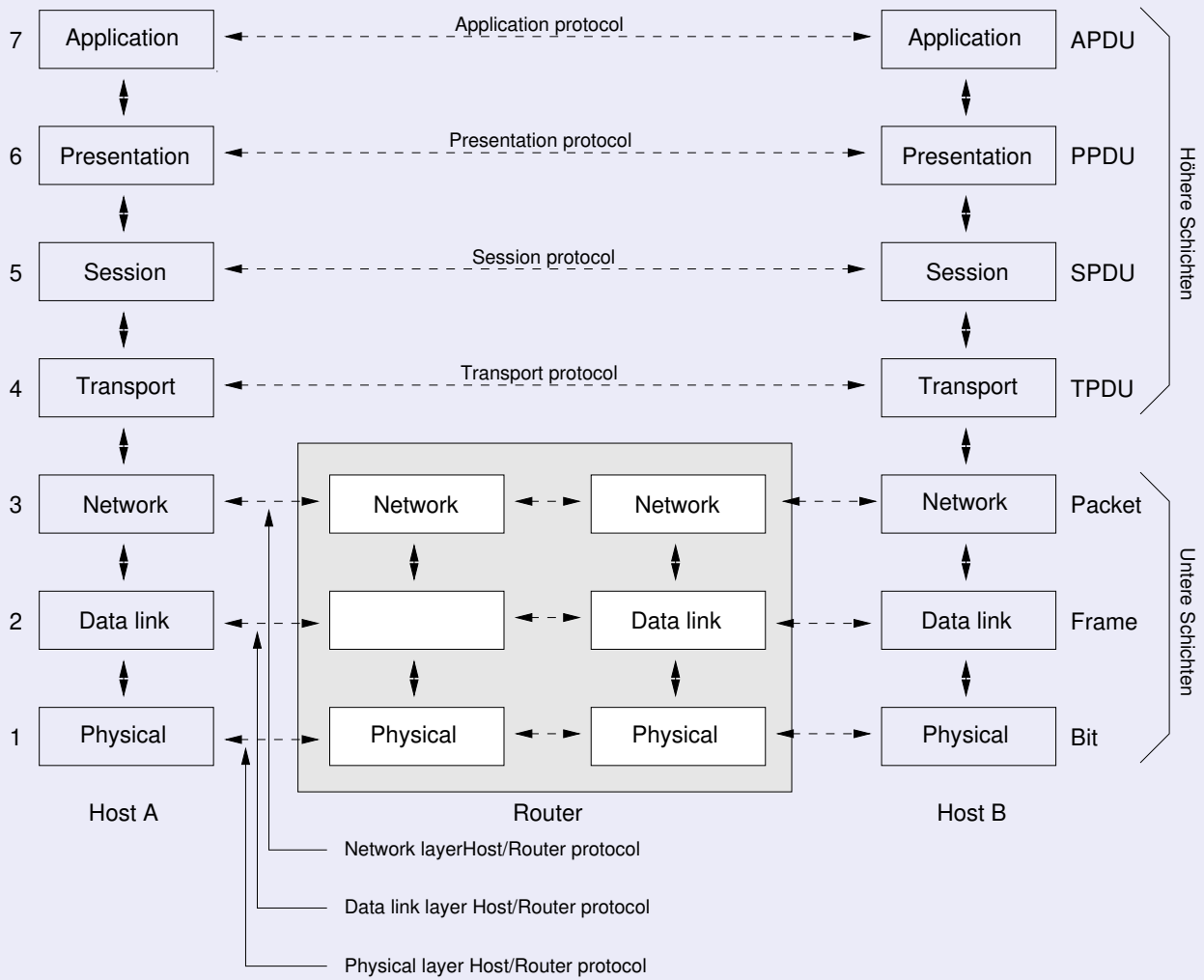
Lösung der Sicherheitsprobleme

- Lösung am Besten auf Protokollebene und nicht erst in den Applikationen ($\Rightarrow$ **Frage:** Wieso?)
- Vollständiges Neudesign der Protokolle: IPv6 (vgl. RFC 1550)
  - Verschlüsselung und Authentisierung
  - Unterstützung zeitkritischer Nachrichten durch Priorisierung
  - 128 Bit-Adressen, ...

  $\Rightarrow$ noch immer in Europa und in den USA wenig verbreitet!
- Deshalb:
  - sichere Protokolle für IPv4
  - möglichst transparente Integration

Computernetze

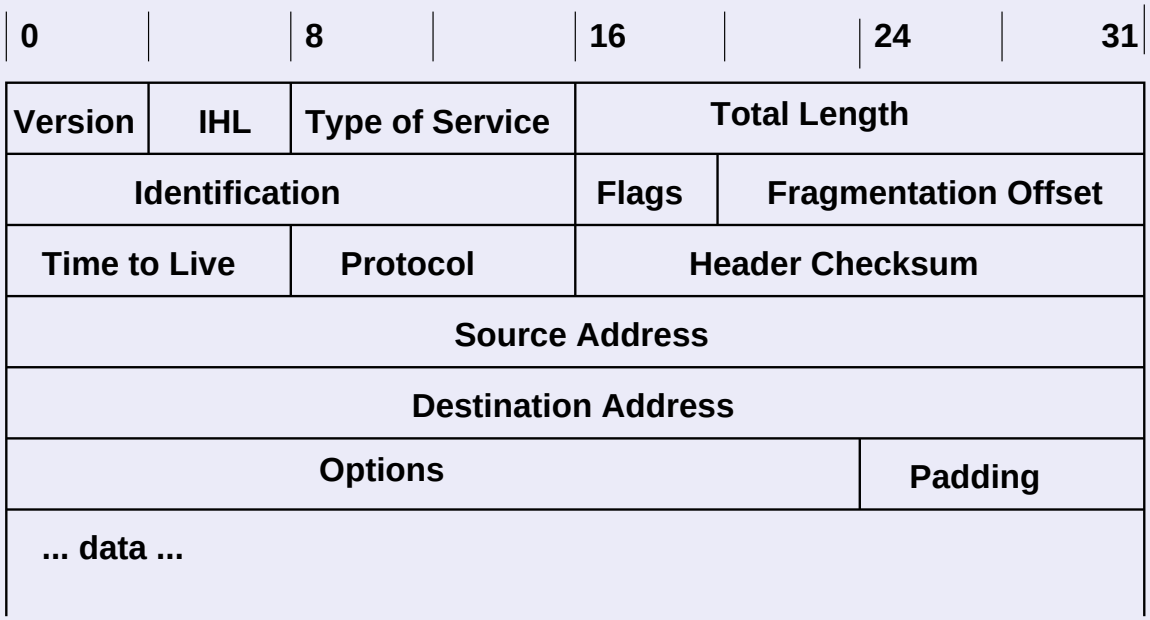
Das ISO/OSI Referenzmodell



Das Internet Protocol (IP)

IP (Version 4)

- Basisprotokoll zur Kommunikation im Internet (1982)
- Protokoll der Schicht 3 (Network Layer)
- IP-Paket mit Header:



Das Internet Protocol (IP) (Forts.)

Funktionalität von IP

- Versenden von *Datagrammen*:
  - Nachrichten ggf. aufteilen
  - im Klartext verpacken in *IP-Pakete*
  - keine Zustellungsgarantie!
  - Paket zu groß für Empfänger? $\Rightarrow$ Sender fragmentiert
- Aufbau einer scheinbar direkten Kommunikation zum Empfänger
  - Zustellen an angegebene Empfängeradresse
  - durch Vermittlungsrechner mittels Routing
  - unter Verwendung von Schicht 2-Protokoll (z.B. Ethernet)

Frage: *Wozu dient die Angabe der Absenderadresse?*

Das Internet Protocol (IP) (Forts.)

Angriff: *IP-Spoofing:*

- Angreifer trägt absichtlich falsche Absenderadresse ein
- Wirkung:
  - Empfänger protokolliert Zugriffsversuch falsch
    $\Rightarrow$ Absender verschleiert seine Identität
  - Empfänger vertraut möglicherweise dem scheinbaren Absender
    $\Rightarrow$ Angreifer kann evtl. ihm nicht zustehende Rechte erhalten!

Frage: *Macht das ein Proxy nicht auch?*

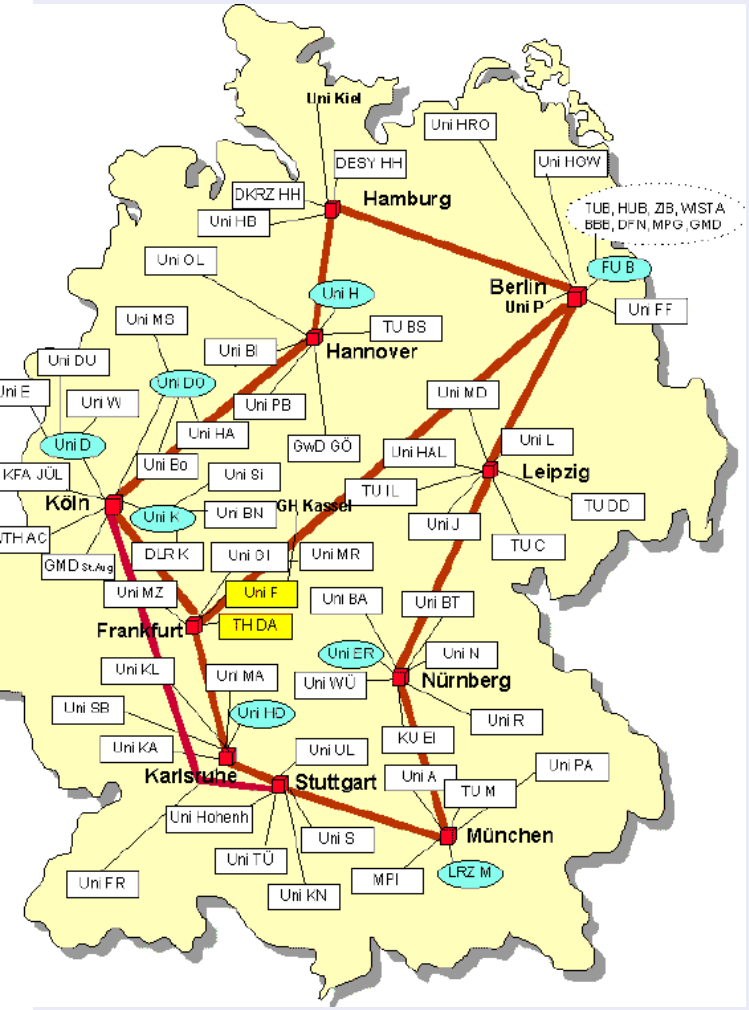
Routing

Grundlagen

- *Router*: Vermittlungsrechner mit mehreren Netzverbindungen
- Vermitteln der IP-Pakete über die derzeitig bestmögliche Route
  $\Rightarrow$ Aufbau der momentan bestmöglichen Routingtabelle!
  (Format: <Zieladresse::nächster Router>)
- Dazu:
  Austausch von Statistik- und Zielinformationen über den momentanen Netzzustand zwischen den Routern
  (Verfahren: Link State Routing, Distance Vector Routing)

Routing (Forts.)

DFN WIN-Netz (1995)



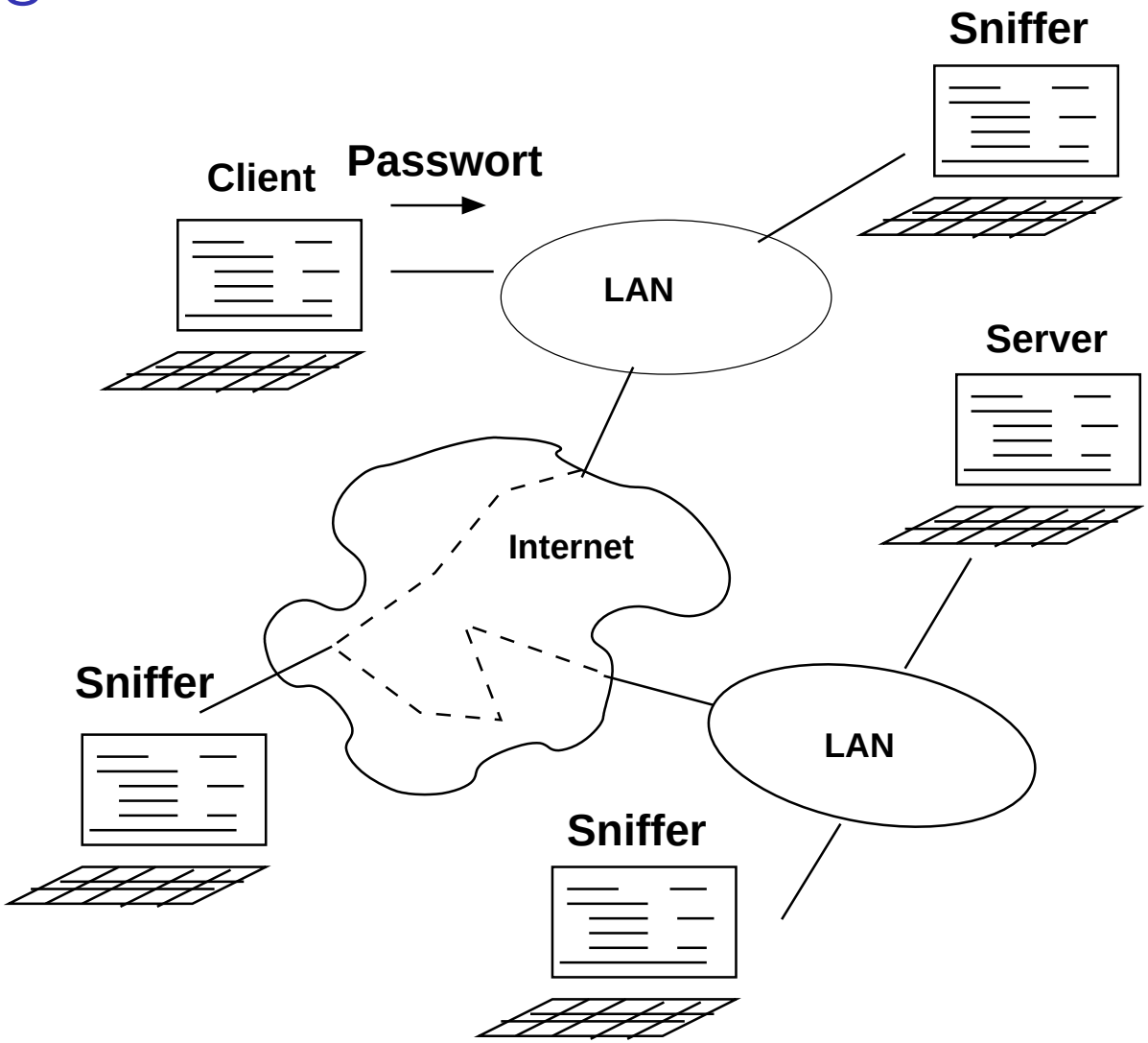

```
> traceroute www2.rrz.uni-hamburg.de
1 cis23-129.hrz.th-darmstadt.de (130.83.129.254) 13 ms
2 cis12.hrz.th-darmstadt.de (130.83.127.3) 7 ms ...
3 TH-Darmstadt1.WiN-IP.DFN.DE (188.1.11.17) 6 ms ...
4 ZR-Frankfurt1.WiN-IP.DFN.DE (188.1.11.13) 8 ms ...
5 ZR-Koeln1.WiN-IP.DFN.DE (188.1.144.33) 11 ms ...
6 ZR-Hannover1.WiN-IP.DFN.DE (188.1.144.25) 24 ms ...
7 ZR-Hamburg1.WiN-IP.DFN.DE (188.1.144.21) 33 ms ...
8 Uni-Hamburg1.WiN-IP.DFN.DE (188.1.3.38) 24 ms ...
9 DKRZgate.HHR.uni-hamburg.de (136.172.251.2) 25 ms ...
10 OVSt44router.HHR.uni-hamburg.de (192.41.147.34) 31
ms
11 UNIRRZgate.HHR.uni-hamburg.de (192.41.147.22) 32 ms
12 www2.rrz.uni-hamburg.de (134.100.249.17) 35 ms ...
```

Angriffe auf das Routing

Spoofing-Angriff

- Router tauschen Routing-Informationen aus
  $\Rightarrow$ Angreifer kann falsche Routing-Information einschleusen
- Wirkung:
  Angreifer kann IP-Pakete über die eigene Maschine umleiten
- weiterer Angriff: Manipulation beim *Source Routing*
  (aktive Wegewahl durch den Sender: angegeben der zu verwendenen Route)
- Realisierung von *Man in the Middle*-Angriffen (MitM)
- Angreifer kann bei IP-Spoofing auch Antworten erhalten

Sniffing-Angriff



Frage: *Wie arbeitet ein Sniffer?*

Sniffing-Angriff (Forts.)

Arbeitsweise der Netzwerkkarte

- Es werden nur Nachrichten an die eigene (Schicht 2-)Adresse angenommen (Normalmodus)
  (**Frage:** Wieso ist das sinnvoll?)
- **Aber:** Superuser kann diese Filterfunktion abschalten
  $\Rightarrow$ *Promiscuous*-Modus: System verarbeitet **alle** Nachrichten
- **Fazit:**

  Bedrohung der Vertraulichkeit über gesamten Nachrichtenweg!

Frage: *Lokale Netze haben heute meist Switches. Schützt das hier?*

Das *Address Resolution Protocol* (ARP)

Wozu ist das gut?

- Dynamisches Protokoll im lokalen Netz zwischen Schicht 2 und Schicht 3
- Umsetzung der IP-Adresse in die (weltweit eindeutige) MAC-Adresse der Netzwerkkarte
- interne Adressstruktur abhängig von Netzwerkarchitektur
  (z.B. Ethernet: MAC-Adressen der Länge 48 Bit)
- Sender schickt ARP-Anfrage per Broadcast ins lokale Netz:
  "Wem gehört diese IP# ?"
- gesuchte Maschine antwortet mit ihrer MAC-Adresse

Das *Address Resolution Protocol* (ARP) (Forts.)

Wissenswert

- MAC-Adressen werden manchmal als Zugangsicherung verwendet (z.B. WLAN)
- begrenzter Schutz, da Betriebssystem die MAC überdecken kann (hierzu ist Betrieb im Promiscous-Modus notwendig $\Rightarrow$ Wieso??)

Angriff: ARP-Spoofing

- Angreifer spielt bei ARP-Anfrage falsch: IP-Adresse $\Leftrightarrow$ falsche MAC-Adresse
- Wirkung: Angreifer übernimmt die Rolle des Zielrechners (MitM)
- Angriff nur im lokalen Netz möglich

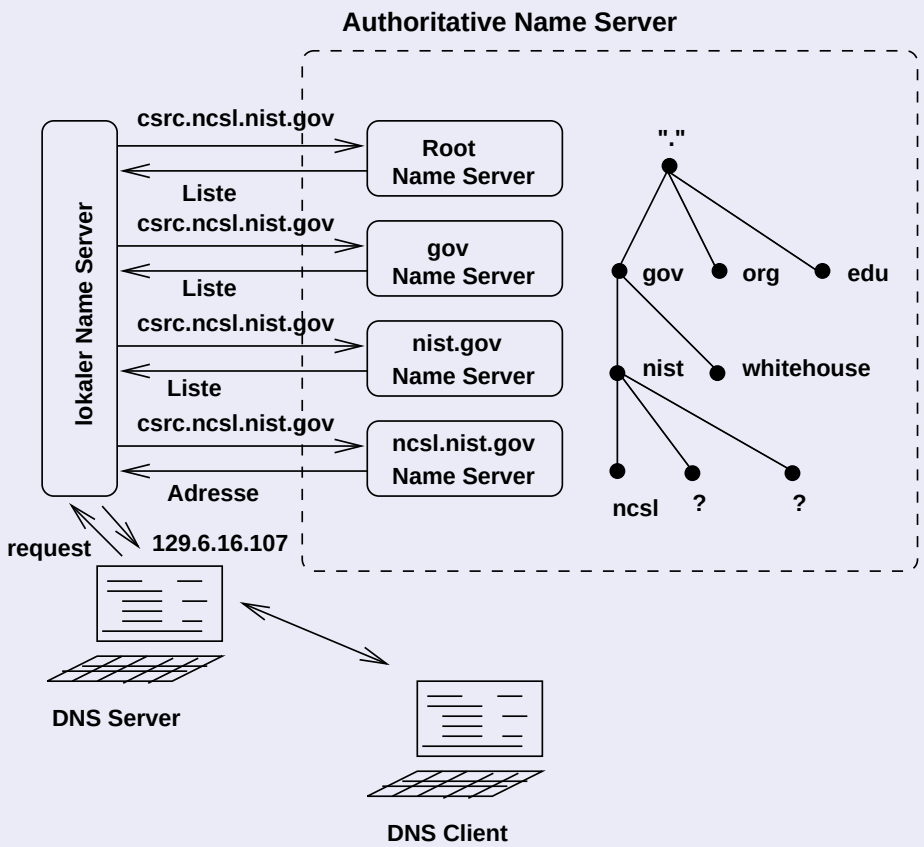
Das *Domain Name System* (DNS)

Wozu ist das gut?

- Rechner werden meist über ihren DNS-Namen benannt:
  $\Rightarrow$ Domainname & Rechnername
- Zum Erstellen von IP-Paketen: Umsetzung von DNS-Namen $\Leftrightarrow$ IP Adresse
- lokale Tabelle, z.B. von Unix in `/etc/hosts`
  $\Rightarrow$ Lösung skaliert nicht!
- daher Verwendung eines verteilten Namensdienstes: DNS

Das *Domain Name System* (DNS) (Forts.)

Namensauflösung



Über DNS

- nur wenige Root-Nameserver
- mehrere Domain-Nameserver
- rekursive oder iterative Namensauflösung
- DNS kann auch zusätzliche Infos über Rechner bzw. Dienste speichern
- Ergebnisse werden für einige Zeit im DNS-Cache zwischengespeichert

Angriffe auf DNS

DNS-Spoofing

- Angreifer antwortet vor echtem DNS-Server
  $\Rightarrow$ Sender akzeptiert die erste erhaltene Antwort
- Problem: DNS authentisiert nicht den Absender
- Variante: *Cache Poisoning*-Angriff
  - DNS-Server tauschen sich untereinander aus
  - DNS-Server speichern auch Adressauflösungen, ohne dass sie eine Anfrage gestellt haben!

  $\Rightarrow$ korrekte und **falsche** Informationen verbreiten sich automatisch!
- Weitere Variante: Manipulieren der lokalen `hosts`-Datei
- **Wirkung der Angriffe:**
  Zugriffe auf Server können Internet-weit auf falsche Hosts umgeleitet werden (MitM), z.B. moderne Phishing-Angriffe

- sehr gefährlich auch in Verbindung mit DHCP Spoofing
  (Frage: Wie geht das?)

Transmission Control Protocol (TCP)

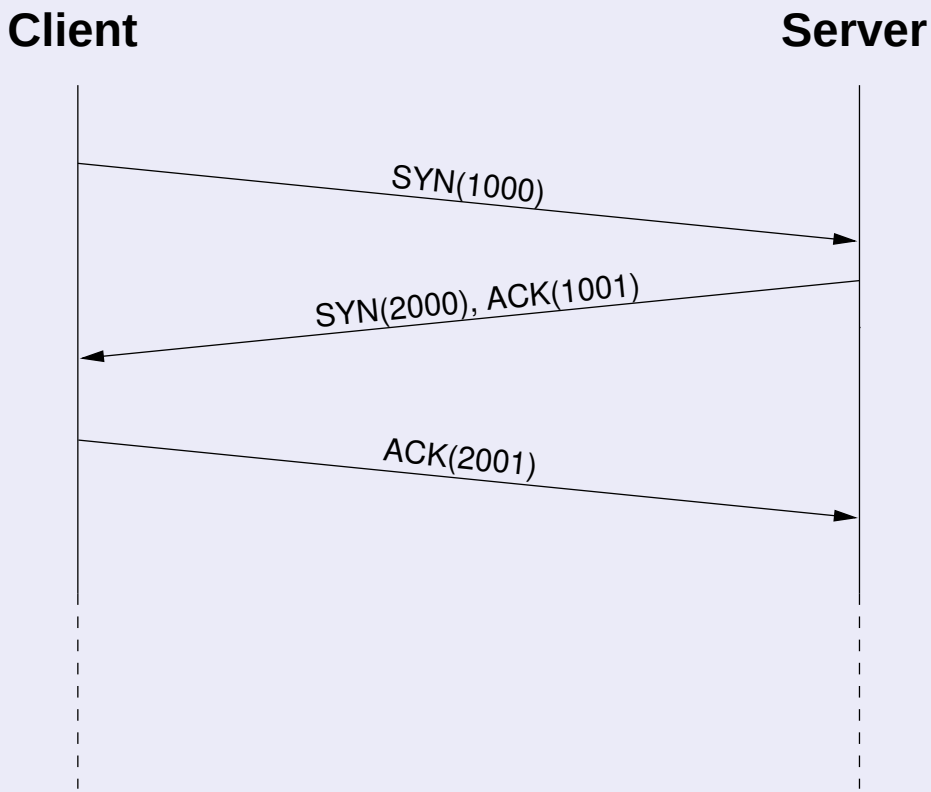
Grundlagen

- Protokoll der Schicht 4 (Transport Layer)
- basiert auf IP
- verbindungsorientierte Kommunikation zweier Rechner im Internet
- zuverlässig und geordnet
  - Verwerfen von Duplikaten und fehlerhaft übertragenen Paketen
  - automatisches Wiederversenden fehlender Paketen
  - Nachrichtenpuffer: Daten werden in korrekter Reihenfolge an Applikation zugestellt
- Verbindungsaufbau immer zwischen zwei *Sockets*
  (Socket-Adresse: IP Adresse und 16 Bit-Port-Nummer)

Transmission Control Protocol (Forts.)

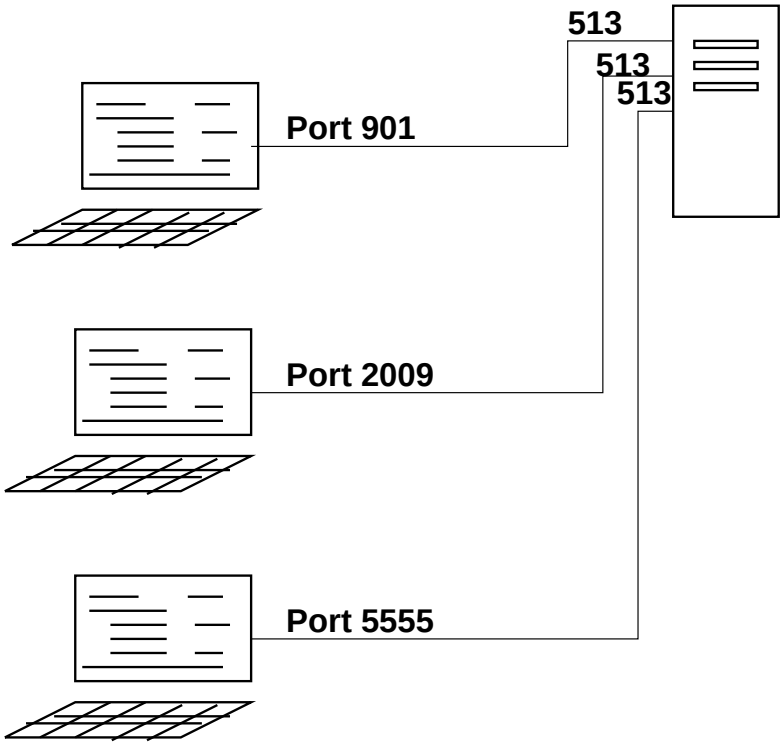
Aufbau einer TCP Verbindung

- „Dreifacher Handshake“



- Das Betriebssystem sollte die initialen Sequenznummern zufällig wählen, so dass ein Angreifer diese nicht leicht vorhersagen kann.

Transmission Control Protocol (TCP) (Forts.)



Ports

- rechnerinterne, eindeutige Adressen für Dienste / Prozesse
- Konzept der *privilegierten Ports:* (Unix)
  An Ports < 1024 dürfen sich nur Systemprozesse binden
- einige Port-Nummern sind Standarddiensten fest zugeordnet

Transmission Control Protocol (TCP) (Forts.)

Port-Nummern einiger TCP-Dienste (Unix:/etc/services)

Protokoll	Dienst	Portnummer
ftp	Dateitransfer	21
ssh	Secure Shell	22
telnet	Virtuelles Terminal	23
smtp	Mailtransport	25
dns	Namensverzeichnis	53
finger	Benutzerinformation	79
http	World Wide Web	80
pop3	Mailabruf	110
https	HTTP über Secure Socket Layer	443
login	Login auf entfernte Rechner	513
spop3	POP3 über Secure Socket Layer	995

Angriffe auf TCP

Warum TCP?

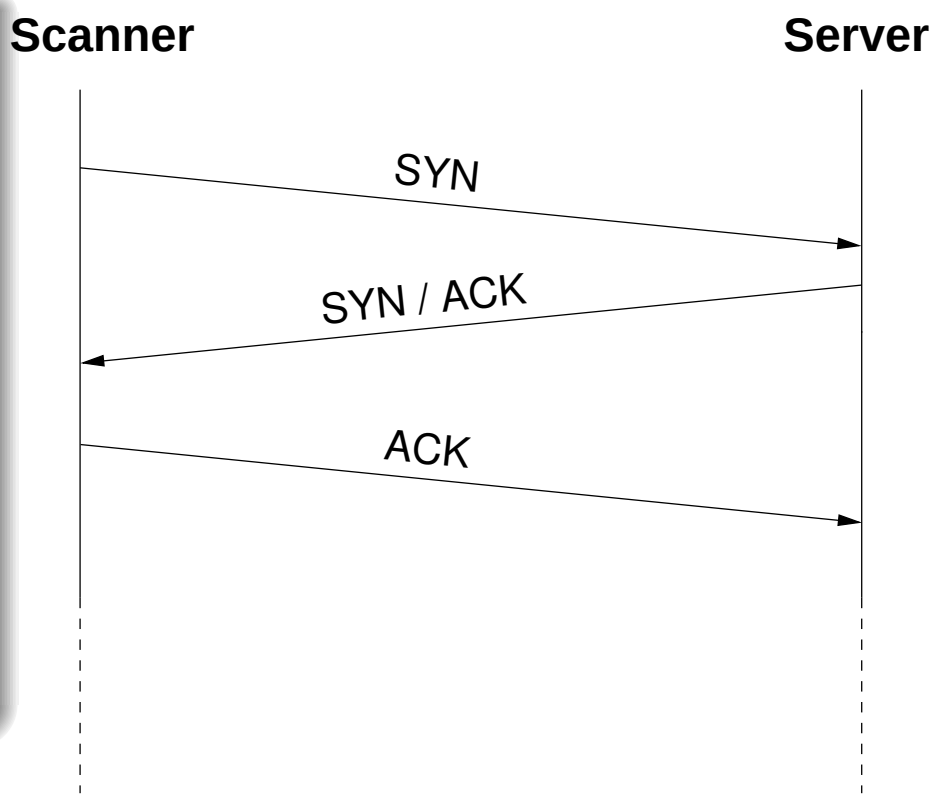
- Netzwerkprogrammierung mit TCP ist relativ komfortabel.
- Die meisten (wichtigen) Dienste sind mit TCP implementiert.
- Angreifer nutzen Schwachstellen ($\Rightarrow$ *Vulnerabilities*) insbesondere in TCP Diensten aus.
- Server haben heutzutage i. Allg. alle nicht verwendeten Dienste geschlossen.
- Angreifer muss verwundbare Dienste finden

  $\Rightarrow$ Port Scans

Port Scans

TCP Connect Scan

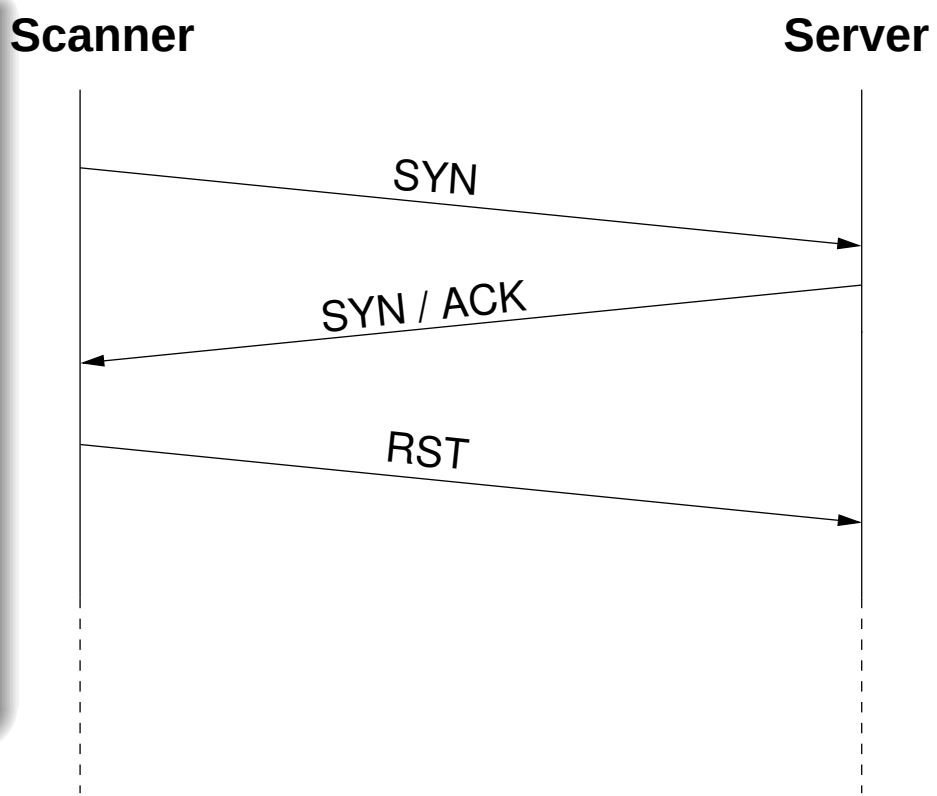
- vollständiger Verbindungsaufbau zu allen bzw. zu ausgewählten Ports
- simpelster Port Scan
- große Entdeckungsgefahr
  (Scan selbst ist kein Angriff)
- Verbesserung:
  zwischen dem Scannen mehrerer Ports Pausen einstreuen
  (Wie lange?)



Port Scans (Forts.)

TCP Syn Scan

- Senden eines TCP-Segments mit gesetztem SYN-Flag an einen Port
  - falls Port offen, kommt SYN/ACK zurück
    $\Rightarrow$ danach RST senden
  - anderenfalls kommt RST (oder gar nichts) zurück
- Verbindung wird nicht geöffnet
  $\Rightarrow$ meist nicht protokolliert
  $\Rightarrow$ Scan bleibt unbemerkt.



Port Scans (Forts.)

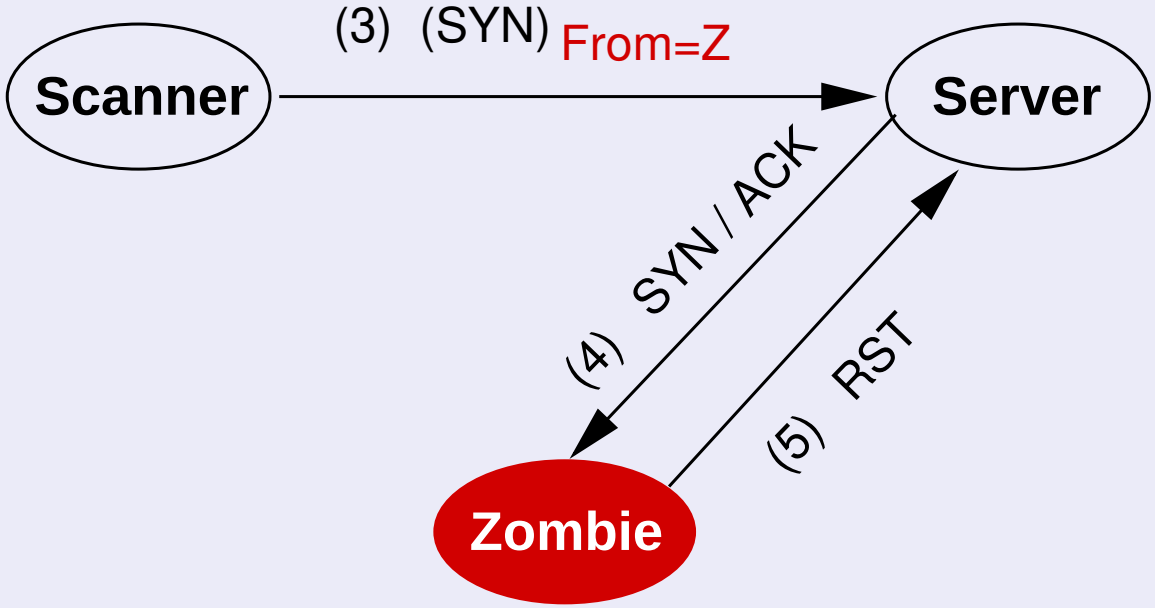
Stealth Scans

- Versenden eines für den Verbindungsaufbau ungültigen TCP-Segments an einen Port.
- Varianten
  - NULL-Scan (keine Flags)
  - ACK-Scan (ACK-Flag)
  - FIN-Scan (FIN-Flag)
  - XMAS-Scan (alle Flags)
- Laut RFC kommt RST zurück, falls Port offen.
  (Reaktion aber abhängig vom Betriebssystem)
- Zugriff wird meist nicht protokolliert
  $\Rightarrow$ Scan bleibt unbemerkt.

Port Scans (Forts.)

Idle Scan

- Bei allen bisher betrachteten Scans kann der Scanner prinzipiell identifiziert werden
- Unter Verwendung eines sog. *Zombies* geht's auch anders

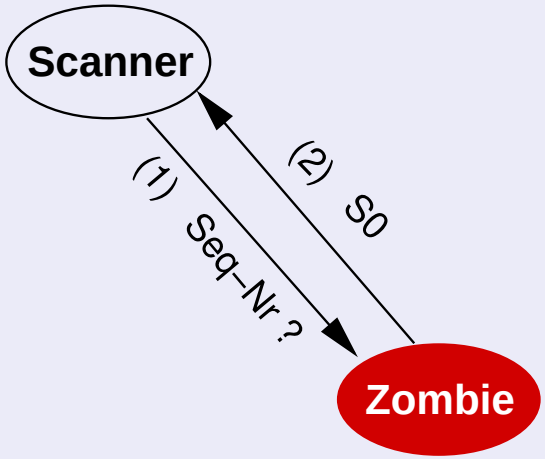


- Wahl des Zombies: *in Vergessenheit geratener* Rechner im Internet möglichst ohne eigenen Netzverkehr und mit altem Betriebssystem

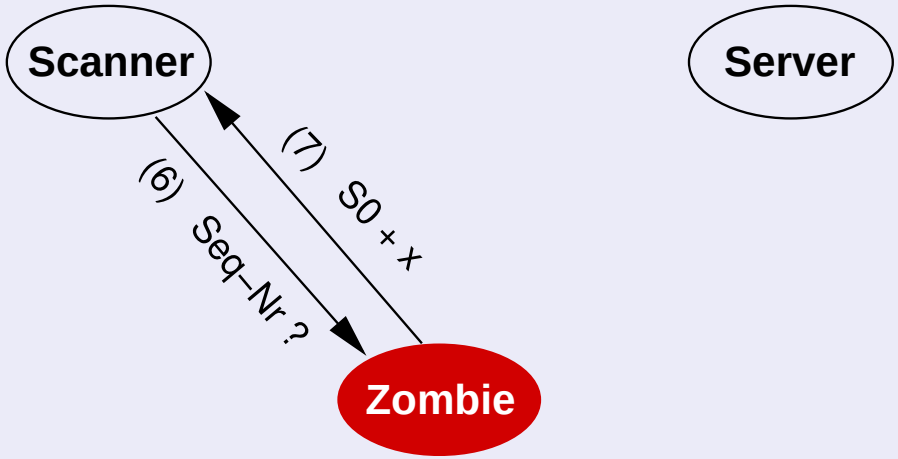
Port Scans (Forts.)

Idle Scan (Forts.)

Zuvor:



Danach:



Frage: *Welchen Wert hat* ***x*** *?*

Port Scans (Forts.)

Tool: nmap

- alle Arten von Port-Scans möglich
- auch *OS fingerprinting*
- u. U. sogar Ermittlung der Versionsnummern von Diensten

Beispiel

```
# nmap -O -sV localhost
Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2007-08-23 15:20 CEST
Interesting ports on localhost (127.0.0.1):
(The 1658 ports scanned but not shown below are in state: closed)
PORT STATE SERVICE
22/tcp open ssh OpenSSH 3.9p1 (protocol 1.99)
25/tcp open smtp Postfix smtpd
80/tcp open http Apache httpd 2.0.53 ((Linux/SUSE))
111/tcp open rpc
631/tcp open ipp CUPS 1.1
Device type: general purpose
Running: Linux 2.4.X—2.5.X—2.6.X
OS details: Linux 2.5.25 - 2.6.3 or Gentoo 1.2 Linux 2.4.19 rc1-rc7)
Uptime 0.223 days (since Thu Aug 23 10:00:08 2007)
Nmap finished: 1 IP address (1 host up) scanned in 2.419 seconds
```

Port Knocking

Schutz vor Port Scans

- Ein *Knock-Daemon* versteckt offenen Ports auf dem Server.
- Zugriffe auf alle Ports werden im Log-File protokolliert.
- Knock-Daemon beobachtet das Log-File.
- Erst nach Erkennen einer vordefinierten *(Einmal-)Klopfsequenz* öffnet der Knock-Daemon den gewünschten Port für diesen Client.
- Client kann nun die Verbindung aufbauen.
- Vgl. M. Krzywinski:
  Port Knocking: Network Authentication Across Closed Ports
  in SysAdmin Magazine 12: 12-17. (2003)

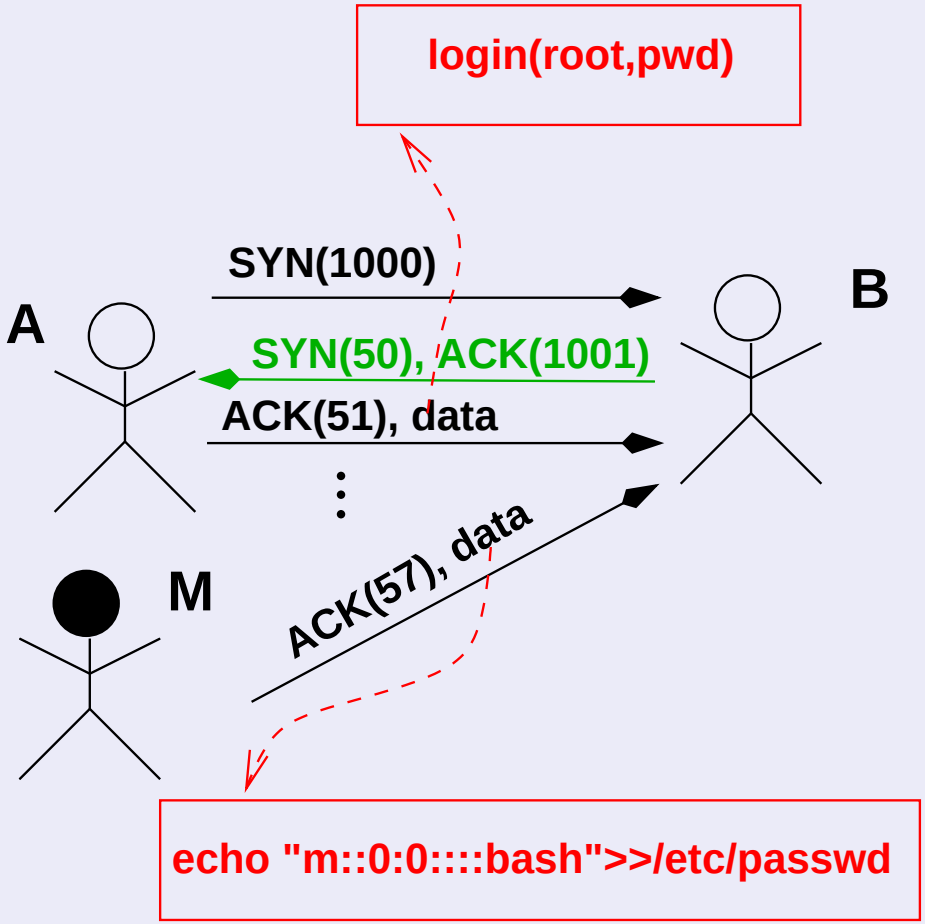
- Weitergedacht von C. Grothoff und J. Kirsch:
  Unsichtbare Server mit TCP Stealth
  URL: https://heise.de/-2399788 in iX 10/2014

Frage: *Wie kann das eingesetzt werden?*

Connection Hijacking: ein anderer Angriff auf TCP

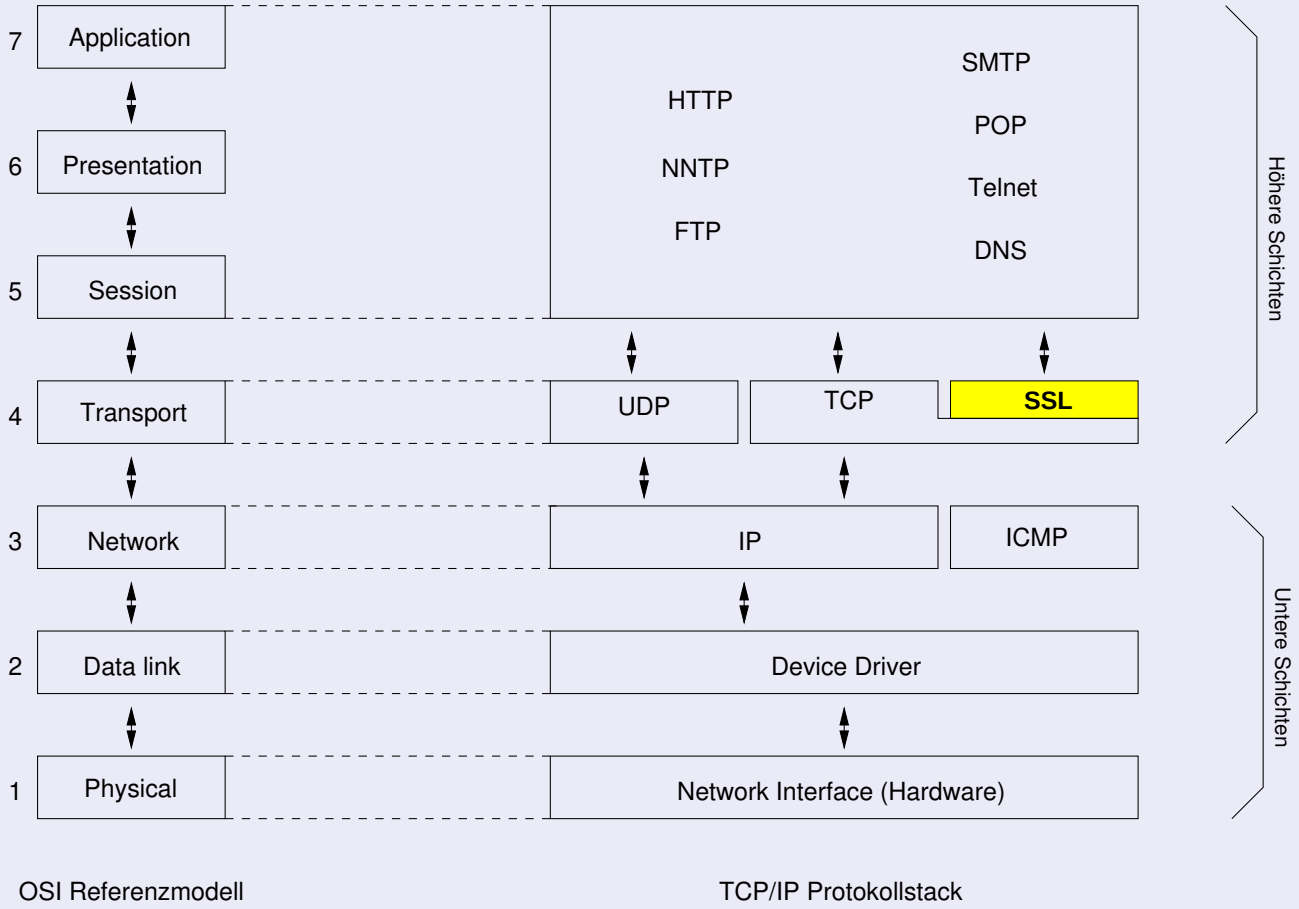
Idee

- Angreifer übernimmt eine bestehende, bereits durch (Einmal-)Passwort authentisierte Verbindung



Secure Socket Layer (SSL)

Wie kann man Sicherheit nachrüsten?



Im Browser: `https://www. ...`

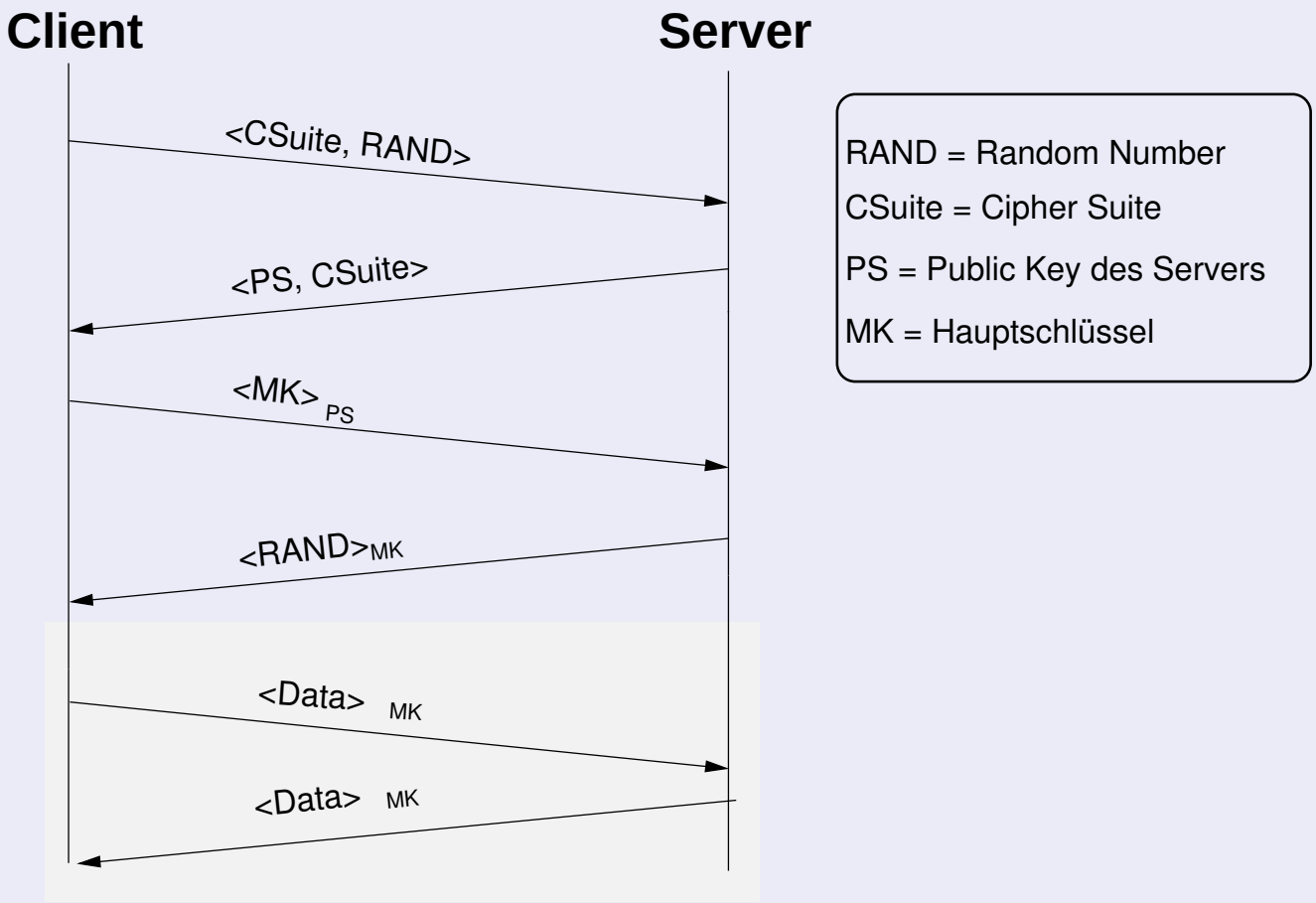
Secure Socket Layer (SSL) (Forts.)

Entwurfsziele (Firma *Netscape*)

- generische Lösung des Sicherheitsproblems (nicht nur HTTP)
- abgesicherte Verbindung für beliebige TCP-basierte Dienste
- flexibel: Sicherheitsniveau je nach Bedarf wählbar
- offen: leichte Integrierbarkeit neuer Verfahren
- transparente Verschlüsselung der Nutzdaten (Schnittstellen eines Transportschichtprotokolls)
- schnell:
  Vermeiden langsamer asymmetrischer Verschlüsselung durch Wiederverwendung alter Authentisierungsinformation
- ab SSL v3.0: *Transport Layer Security* (TLS)
- aktuell: TLS v1.3

Secure Socket Layer (SSL) (Forts.)

Prinzip des SSL-Verbindungsaufbaus (schematisch)



Frage: *Welche Schutzziele werden wie umgesetzt? Klappt das?*

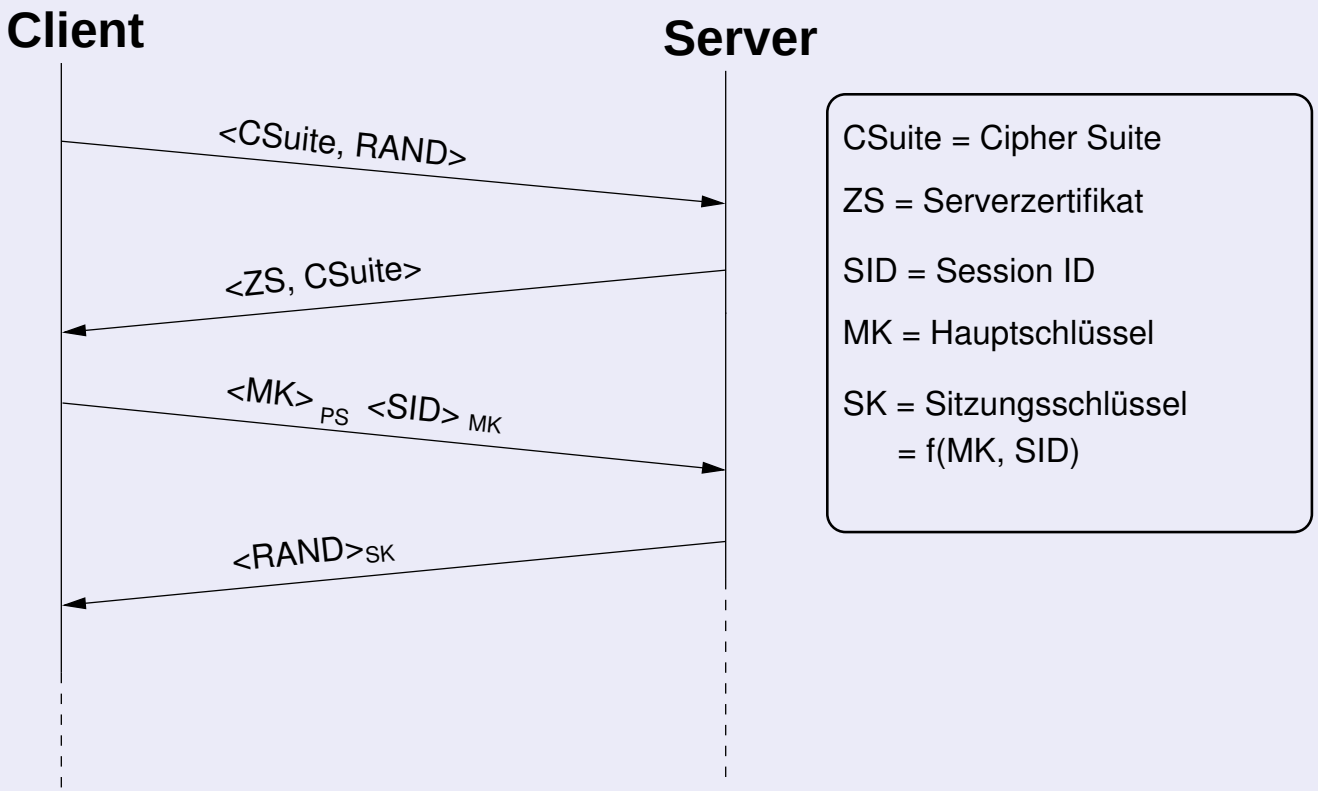
Secure Socket Layer (SSL) (Forts.)

Zum Verbindungsaufbau

- Server muss sich gegenüber dem Client immer authentisieren
- Client muss dies nur auf Anforderung des Servers
- Kommunikationspartner einigen sich auf eine *Cipher Suite*:
  (1) Verfahren für Authentisierung / Schlüsseltausch (i. d. R. asymm.)
  (2) Verfahren zur Nutzdatenverschlüsselung (symm.)
  (3) Verfahren zur Integritätssicherung (MAC, krypt. Hashfunktion)
- Bilden eines gemeinsamen Geheimnisses:
  - einfaches Diffie-Hellman Verfahren ohne Authentisierung
  - RSA zur verschlüsselten Übertragung des Geheimnisses
  - Diffie-Hellman mit Signierung der übertragenen Parameter

Secure Socket Layer (SSL) (Forts.)

Verbesserter Verbindungsaufbau (schematisch)



Frage: *Welche Informationen muss das Zertifikat enthalten?*
Welche Angriffe sind noch möglich?

Secure Socket Layer (SSL) (Forts.)

Schlüsselaustausch und Authentisierung

- übergeordnete Zertifizierungsautoritäten
- einseitige oder gegenseitige Authentisierung mittels (X.509)Zertifikaten
- Zertifikat des Servers enthält DNS Namen des Rechners
- verkürztes Verfahren für häufige Verbindungen zum selben Server:
  - Client schickt beim Verbindungsaufbau SID einer früheren Verbindung mit
  - falls Session ID beim Server noch bekannt
    $\Rightarrow$ einfach neuen Sitzungsschlüssel (aus MK + SID) generieren
- Einsparung der langsamen asymmetrischen Verschlüsselung
  $\Rightarrow$ viel schneller! (insb. beim Abruf von WWW-Dokumenten)

Secure Socket Layer (SSL) (Forts.)

Übertragung der Nutzdaten

- Voraussetzungen:
  - spezifischer Sitzungsschlüssel für den vereinbarten symmetrischen Verschlüsselungsalgorithmus
  - geheimer Wert für Message Authentication Code (MAC)
- Bildung des MAC (z.B. mit SHA-1) aus geheimen Wert, Daten, Padding-Daten und Sequenznummer
  - Sequenznummer verhindert Replay-Attacke
  - Voranstellen des geheimen Wertes verhindert Known-Plaintext Angriffe (bei HTTP Abfrage z.B. immer `GET`)

Denial-of-Service-Angriffe (DoS)

Ziel des Angreifers

- Lahmlegen eines Dienstes oder des ganzen Systems
  - durch Ausnutzen von Schwachstellen (*vulnerabilities*, z.B. Buffer Overflow)
  - durch Generierung von Überlast

Exemplarisch:Ping-of-Death (historisch aus dem Jahr 1997)

- ping verwendet Internet Control Message Protocol (ICMP)
- üblicherweise kleine Nachrichten, verwendete Länge aber einstellbar
- falls zu groß $\Rightarrow$ Buffer Overflow $\Rightarrow$ Systemabsturz !

- Variante: mittels Fragmentierung ließen sich generell übergroße IP-Pakete (>65,536 Byte) erstellen.

Denial-of-Service-Angriffe (DoS) (Forts.)

SYN-flooding Angriff

- Angriff auf Design
- Angreifer sendet eine Verbindungsaufbauanforderung (gesetztes SYN-Flag) an Zielmaschine
- Server generiert eine halboffene TCP-Verbindung
- Angreifer wiederholt in schneller Folge dieses erste Paket zum Verbindungsaufbau
  $\Rightarrow$ vollständiges Füllen der internen Systemtabelle
  $\Rightarrow$ Anfragen normaler Benutzer werden zurückgewiesen
- Angreifer verwendet i. Allg. IP-Spoofing
  $\Rightarrow$ Firewall wirkungslos
- mögliche Abwehr: SYN-Cookies
  D J. Bernstein: *SYN cookies* in URL http://cr.yp.to/syncookies.html

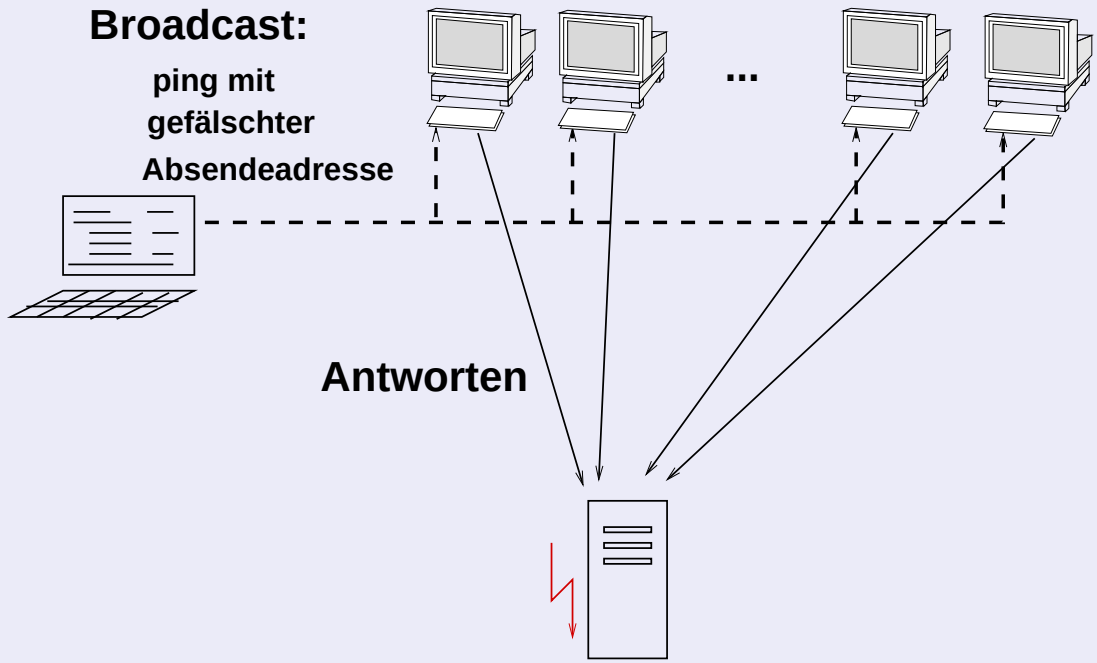
Frage: *Wie funktionieren SYN Cookies – ohne TCP zu ändern?*

Denial-of-Service-Angriffe (DoS) (Forts.)

Verteiltes DoS (DDoS)

- Opfer wird von sehr vielen Angreifern mit Nachrichten überflutet

Beispiel: Smurf-Angriff



Frage: *Welche Varianten? Wie lässt sich ein DDoS-Angriff abwehren?*

Schutz gegen *Password Sniffing*

Verwendung von `rlogin` bzw. `telnet`

- Passwort muss eingegeben und übertragen werden
- Übertragung in IP-Paketen als Klartext
- durch Sniffing beliebig abhörbar
- Sicherheit?? (Passwort wiederverwendbar)

Frage: *Wie findet ein Angreifer in dieser Datenflut die Passwörter?*

Angriff: *Password Sniffing* (Forts.)

Gescheiterter Versuch der Abhilfe

- Unix: Eintrag in `.rhosts` Datei (Zugriff nur für Besitzer!)
  - Inhalt:

```
pagnia priv-pc
pagnia 130.83.24.10
```

  - kaum Sicherheit!

Wirkungsvoller

- Einmal-Passwörter
- Passwort nur verschlüsselt übertragen (RSA + IDEA, u.a.)
  z.B. `ssh`, `scp`
  - für TCP/IP-Verbindungen
  - für X11-Verbindungen

Einmal-Passwörter

Idee

- Passwort nur genau **einmal** gültig und daher nicht wiederverwendbar

Methoden

- **Token Card:**
  Spezielle Hardware-Karten mit Display
  zum Ermitteln des gerade gültigen Passworts
- **Codebuch:**
  Passwort-Liste als gemeinsames Geheimnis;
  elektronisch oder auf Papier gespeichert

Einmal-Passwörter (Forts.)

RSA-SecurID-Card (früher: SecurID)

- Token Card, *tamper proof*
- *Two-Factor Authentication*:
  Karte und PIN
- pseudo-zufällige Zahlenfolge
- wechselt alle 60 Sekunden
- Server-Passwort:
  6- oder 8-stellige Zahl
  ggf. + PIN
- kommerziell:
  Algorithmen nicht offengelegt

Das S/Key-Verfahren

Prinzip

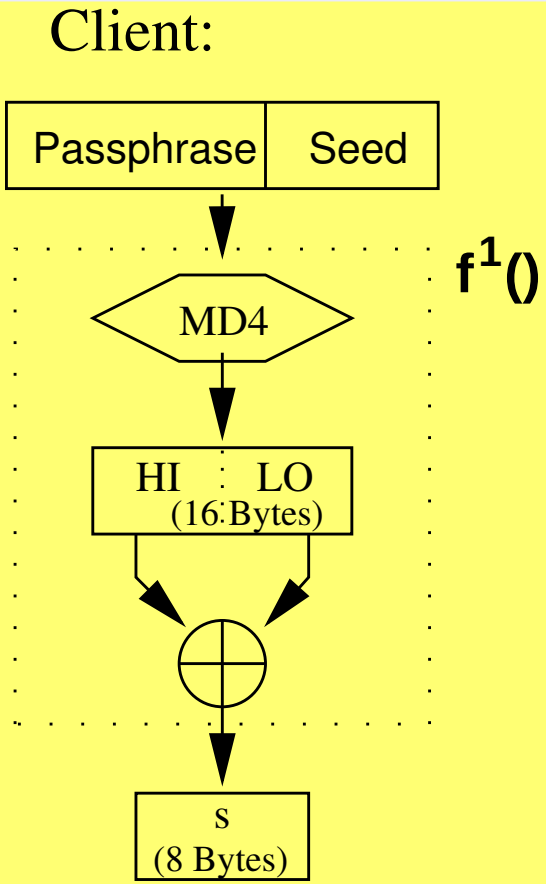
- Einmal-Passwort-System nach Codebuch-Verfahren
- basiert auf kryptographischer Hashfunktion MD4

Anmeldevorgang bei Server

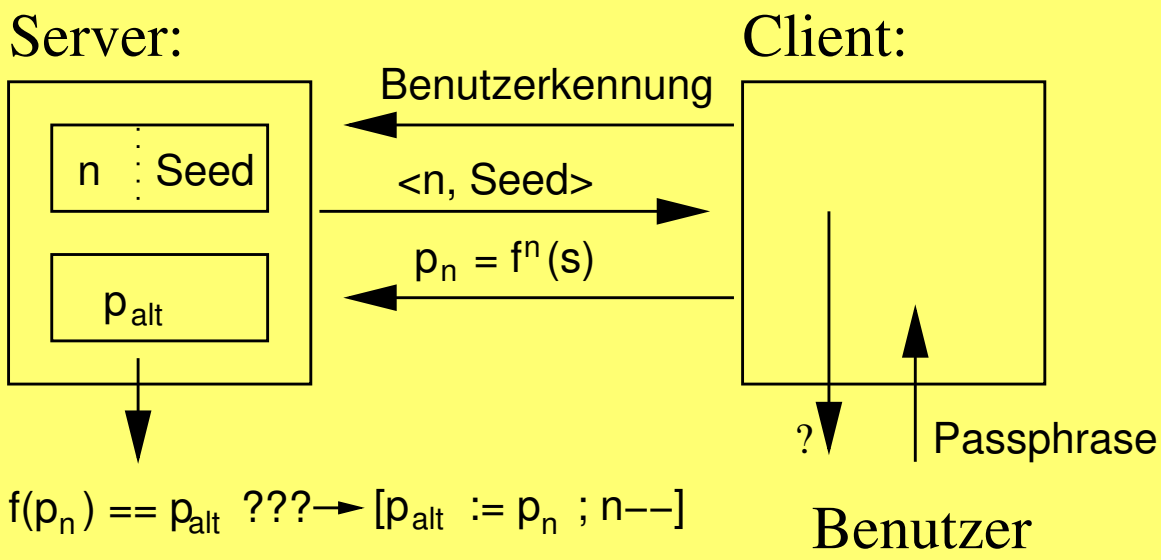
- Initialisierung: Hinterlegen eines S/Key Init-Passwortes
  $\Rightarrow p_N = f^N(s)$
- erlaubt N-maliges Ausführen von f
  (also N-maliges Anmelden beim Server)
- Berechnung von s aus *Passphrase* des Benutzers (Tool:key-Programm)
- Initialisierung p_N liegt im Klartext auf dem Server
  $\Rightarrow$ kein Passwort!!!

Das S/Key-Verfahren (Forts.)

Schematischer Ablauf



Die Funktion f()



Anmeldevorgang

Das S/Key-Verfahren (Forts.)

Beispiel zum Ermitteln des S/Key-Passwortes

```
$ key 23 unix2
   Reminder - Do not use this program while logged in via telnet
or rlogin.
   Enter secret password: <Passphrase>
  > GRAB CUFF MERT GANG TIE ADEN
```

(Die 64-Bit Passwörter werden nach einer fest vorgegebenen Liste als kurze englische Wörter dargestellt.)

Secure Shell (SSH)

Verschlüsselte Verbindung

- asymmetrisch / symmetrisch kombiniert
- Schutz auch für X11-Verbindungen

Ablauf

(1) Authentisierung des Server-Rechners

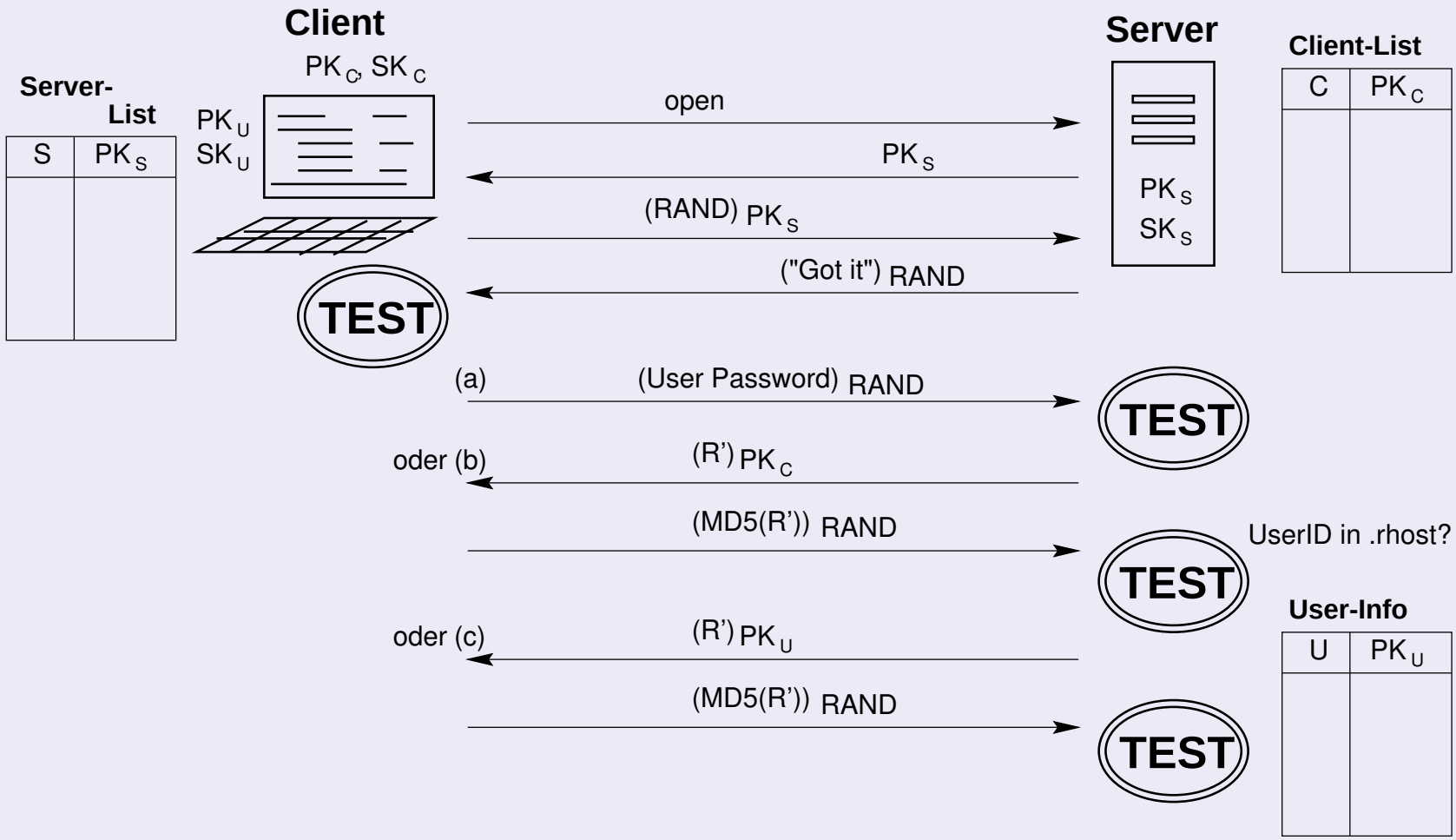
(2) Authentisierung des Benutzers (bzw. des Clients) mittels

  (a) Passwort
  (b) `.rhosts`-Eintrag
  (c) privatem RSA-Key

(3) Kommunikation über symmetrisch verschlüsselte Verbindung

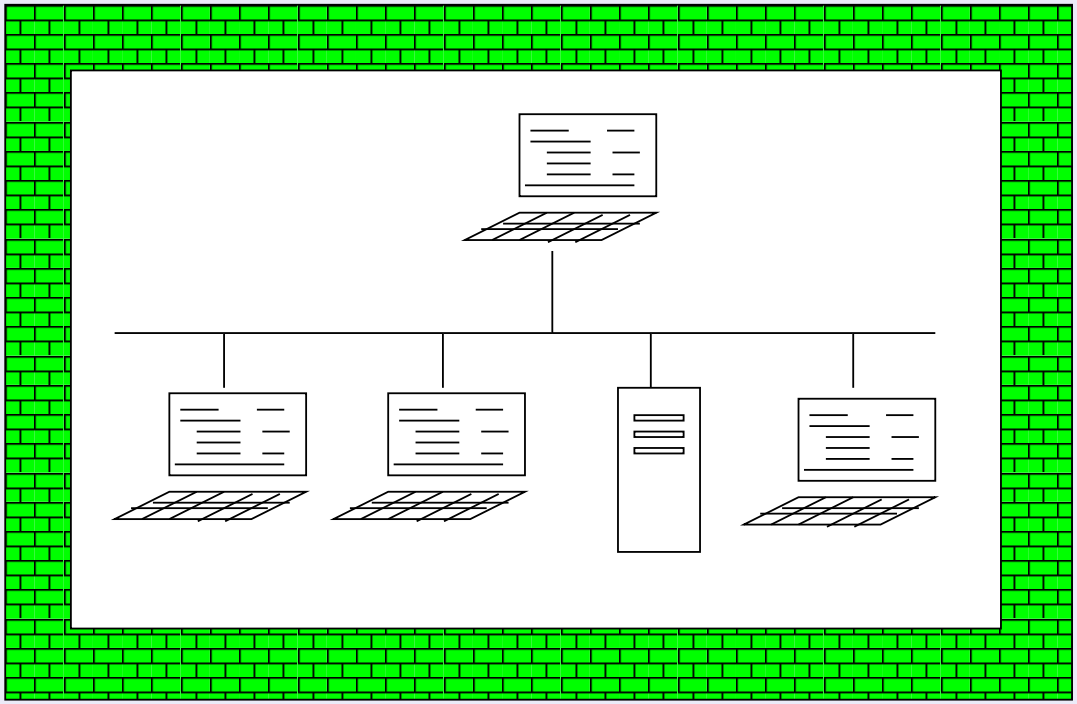
Secure Shell (SSH) (Forts.)

Schematischer Verbindungsaufbau



Firewalls

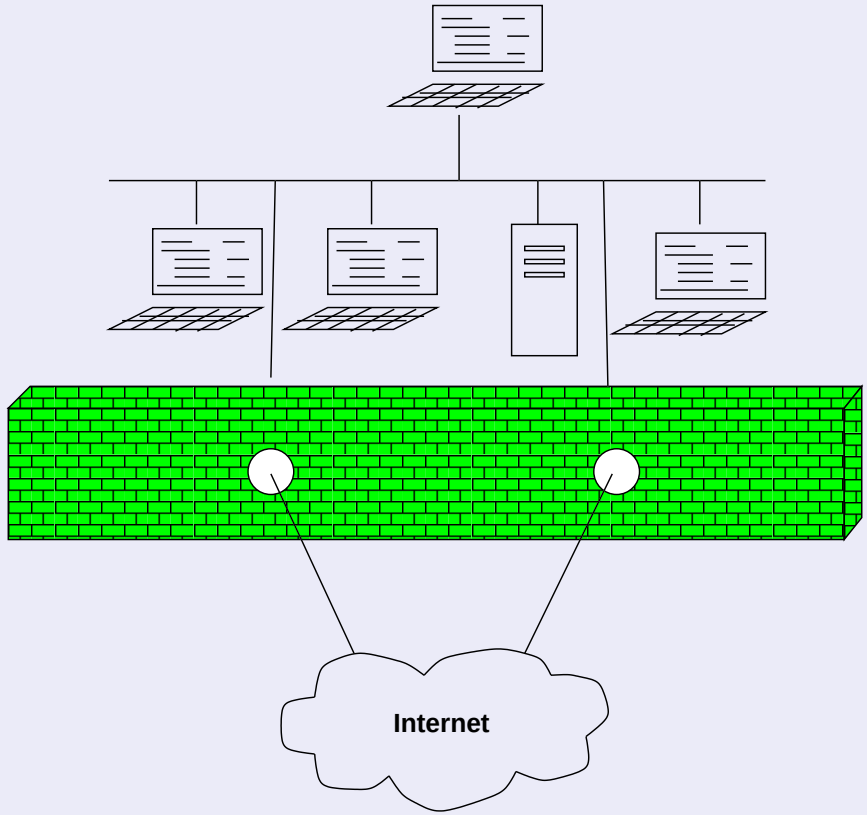
Schutz interner Netze – ideale Situation für den Admin



- Vorteil: keinerlei Angriffsmöglichkeiten von außen
- Nachteile:
  - kein Schutz gegen Insider
  - kein Zugang zum World-Wide Web
  - kein E-Mail-Verkehr von / nach außen

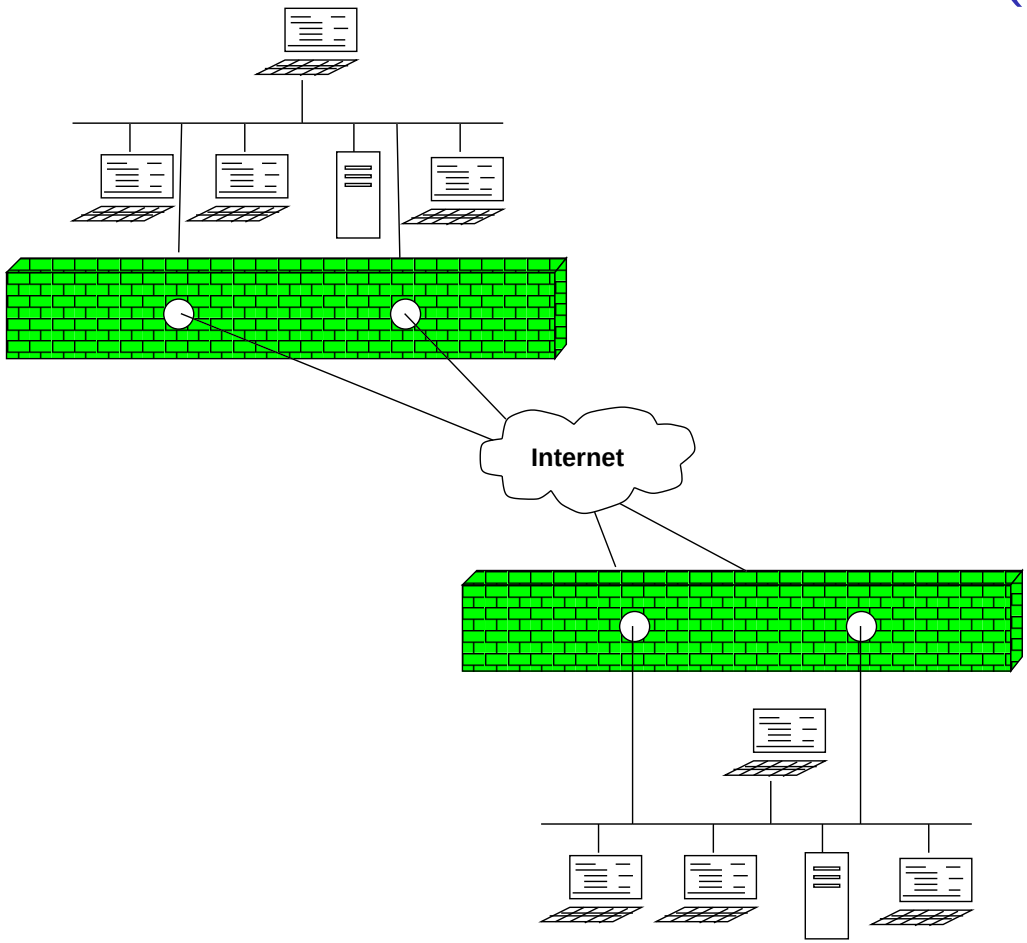
Firewalls (Forts.)

Schutzschicht zwischen internem und externem Netz



- Kontrolle des Nachrichtenverkehrs durch Filterung
- begrenzte Isolation $\Rightarrow$ begrenzter Schutz

Einsatzmöglichkeit: Virtual Private Network (VPN)



- Aufbau einer scheinbar privaten Verbindung von Firmenteilnetzen über das (öffentliche) Internet
- Zusätzliche Verbindungsverschlüsselung zwischen den Firewalls!

Firewalls (Forts.)

Zentraler Schutz des gesamten internen Netzwerks durch

- *Packet Filter* (Schicht 3 und 4)
  - Blockieren bestimmter IP-Empfänger-Adressen (extern / intern)
  - Blockieren bestimmter IP-Absender-Adressen (extern / intern)
    * z.B. aus dem Internet mit internen IP-Absender-Adressen
  - Blockieren bestimmter Dienste; ggf. nur für bestimmte IP-Adressen
- *Application-Level Filter* (Schicht 7)
  - inhaltsbezogene Filterung der Verkehrsdaten eines Dienstes
  - z.B. Virenfilter
  - wirkungslos bei verschlüsselten Verkehrsdaten
- Protokollierungsmöglichkeit der Kommunikation von / nach extern

Firewalls (Forts.)

Realisierungsmöglichkeiten

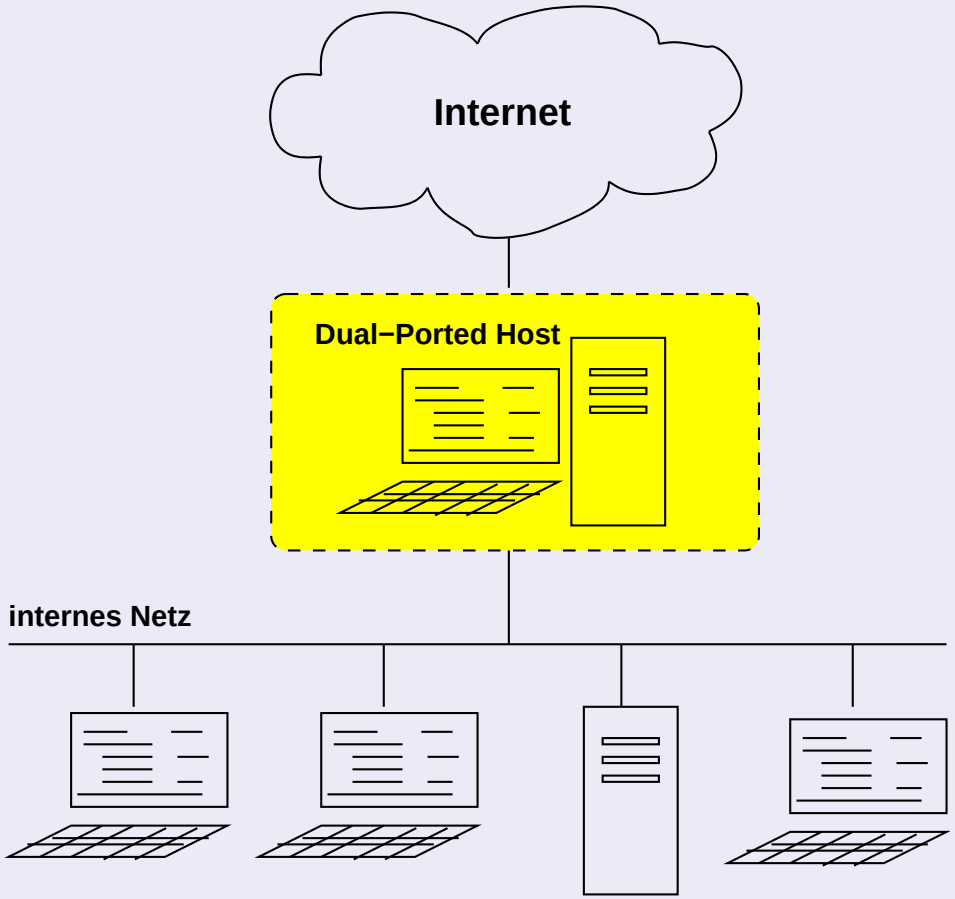
- Hardware-Firewall
- Software-Firewall (*Personal Firewall*)

Hardware-Firewall: Bausteine

- Screening Router (auch *Choke*)
- Gate (auch *Bastion Host*)
  - Proxy-Server für bestimmte Dienste
  - Client-Software (HTTP-Browser, `telnet`, `ftp`, `...`)
  - Server-Software (aber nicht HTTP-Server o.ä. !)

Architekturen von Firewall-Systemen

Dual-Ported-Host

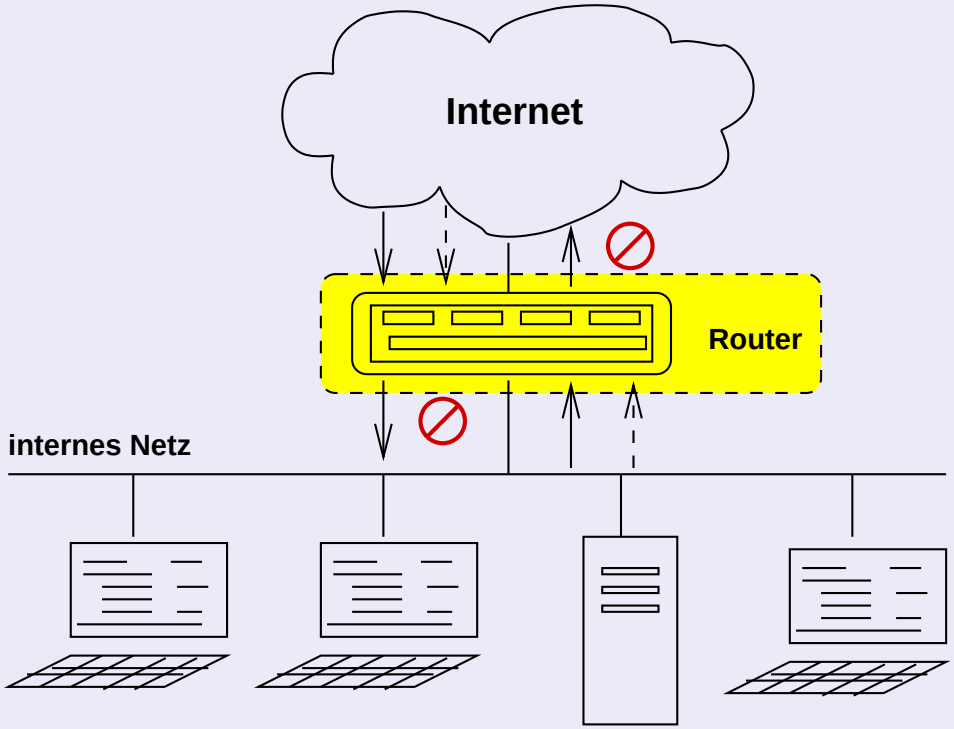


Aufbau

- zwei Netzwerkkarten: ggf. private interne Adressen
- Screening Router & Gate: Packet Filter und Application-Level Filter
- Proxy-Dienste installieren
- Benutzer-Logins von extern
- Konf. der Netzwerkkarten: IP-Pakete nicht automat. Weiterleiten!

Architekturen von Firewall-Systemen (Forts.)

Screening Router



Aufbau

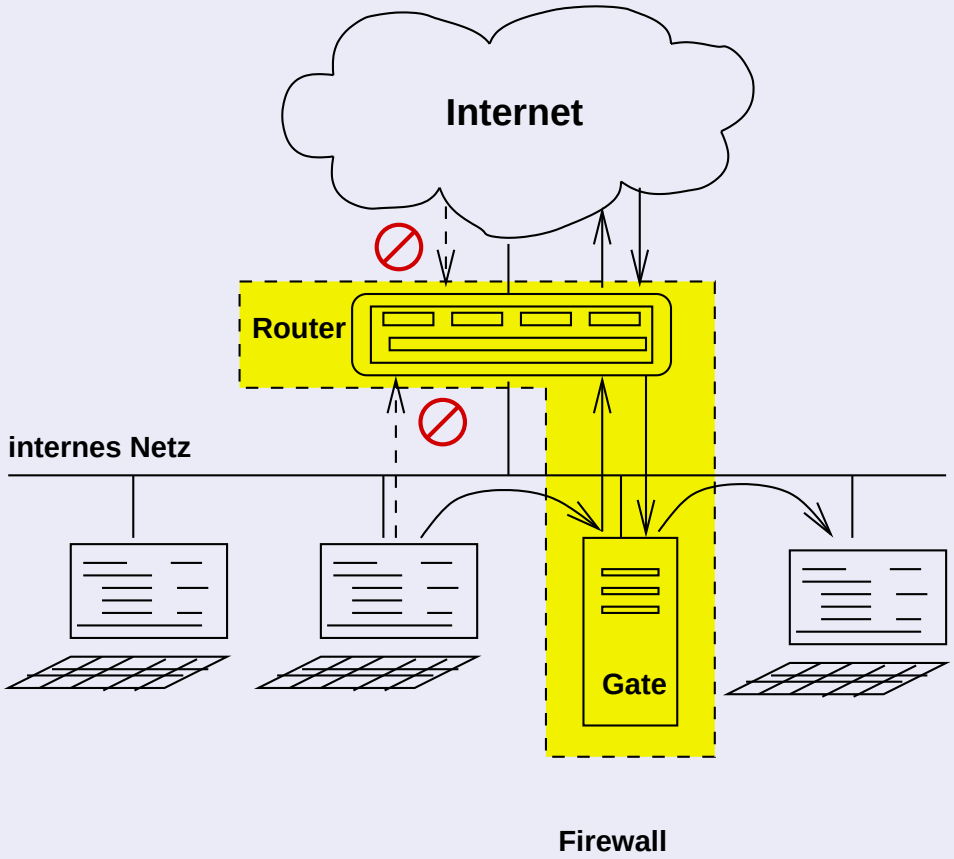
- programmierbarer HW-Router
- simple Filterfunktionen:
  - nur Paket-Header prüfen
  - schnelle Auswertung $\Rightarrow$ hoher Durchsatz
- Realisierung eines *Packet Filter*

Bewertung

- $\oplus$ einfach und billig
- $\oplus$ flexibel
- $\ominus$ schwer zu testen
- $\ominus$ Protokollierung
- $\ominus$ Fernwartung
- $\ominus$ Router ist Angriffspunkt
- $\ominus$ keine Inhaltsfilterung

Architekturen von Firewall-Systemen (Forts.)

Screened Host



Aufbau

- Screening Router blockiert:
  - Pakete von / an interne Rechner (nicht Gate)
  - Source-Routed Pakete
- von extern nur Gate sichtbar
- Pakete von intern nur via Gate
- Gate bietet Proxy-Server (z.B. für E-Mail)

Architektur von Firewall-Systemen (Forts.)

Sichere Gate-Konfiguration: minimale angreifbare Oberfläche

- Abschalten aller nicht-benötigten Netzdienste;
  Löschen aller nicht benötigter Programme;
  Rechte von `/bin/sh` auf 500 setzen;
  Rechte aller Systemverzeichnisse auf 711 setzen
- keine regulären Benutzerkennungen
- `root`-Login mit Einmal-Passwortsystem
- keine Vertrauensbeziehungen (`/etc/hosts.equiv /etc/hosts.lpd`)
- setzen von Platten- und Prozess-Quotas
- volle Protokollierung, möglichst auf *Hardcopy*-Gerät
- möglichst sichere, stabile Betriebssystemversion einsetzen;
  Updates regelmäßig einspielen

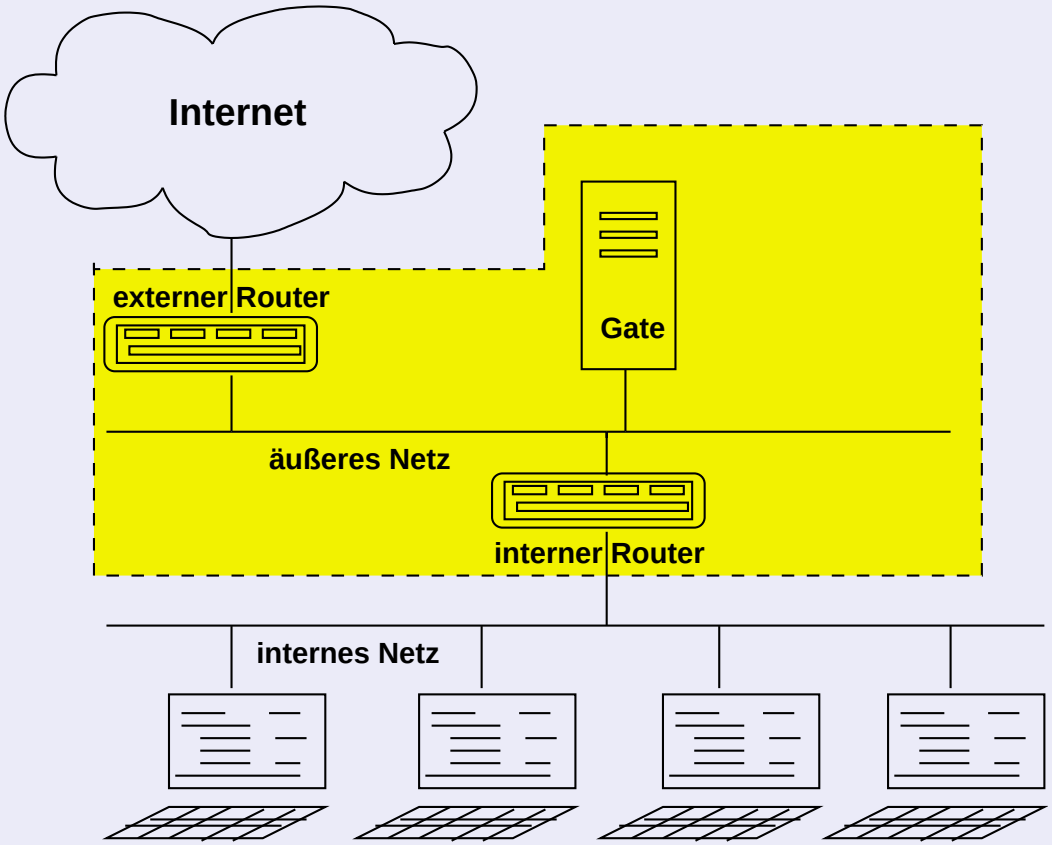
Architektur von Firewall-Systemen (Forts.)

Konfiguration für den Zugriff von extern

- Vergabe von Benutzerkonten mit zufälligen Namen für das Gate
- Verwenden von Einmal-Passwortsystemen oder Zufalls-Passworten mit PWD-Aging
- keine Vertrauensbeziehungen(`.rhosts` Dateien)
- anschließend unmittelbares Anmelden auf Arbeitsplatzrechner mit „normaler“ Benutzerkennung

Architekturen von Firewall-Systemen (Forts.)

Screened-Subnet



Aufbau

- interner Screening Router als dritter Schutzwall
  - blockiert Dienste, die nicht einmal bis zum Gate gelangen sollen
  - lässt nur Pakete zum / vom Gate durch
- äußeres Netz realisiert *Demilitarisierte Zone* (DMZ)
  - guter Platz für HTTP-Server, Mail-Server, ...

Intrusion Detection Systeme (IDS)

Wozu?

- Firewall alleine ist zu statisch
- bessere Aufzeichnung und flexiblere Erkennung notwendig
- angepasste Reaktion notwendig

  $\Rightarrow$ an verschiedenen Stellen spezielle *Sensoren* platzieren

Sensor

- *Definition:* Gerät (meist speziell konfigurierter Rechner), das vielfältige Techniken zur Erkennung von Angriffen anwendet und Angriffe meldet
- Rückkopplung mit Firewall möglich
  ($\Rightarrow$ automatische Umkonfiguration der Firewall)

Intrusion Detection Systeme (Forts.)

Sprechweisen

- Intrusion Detection (IDS)
- Intrusion Response (IRS)
- Intrusion Prevention (IPS)

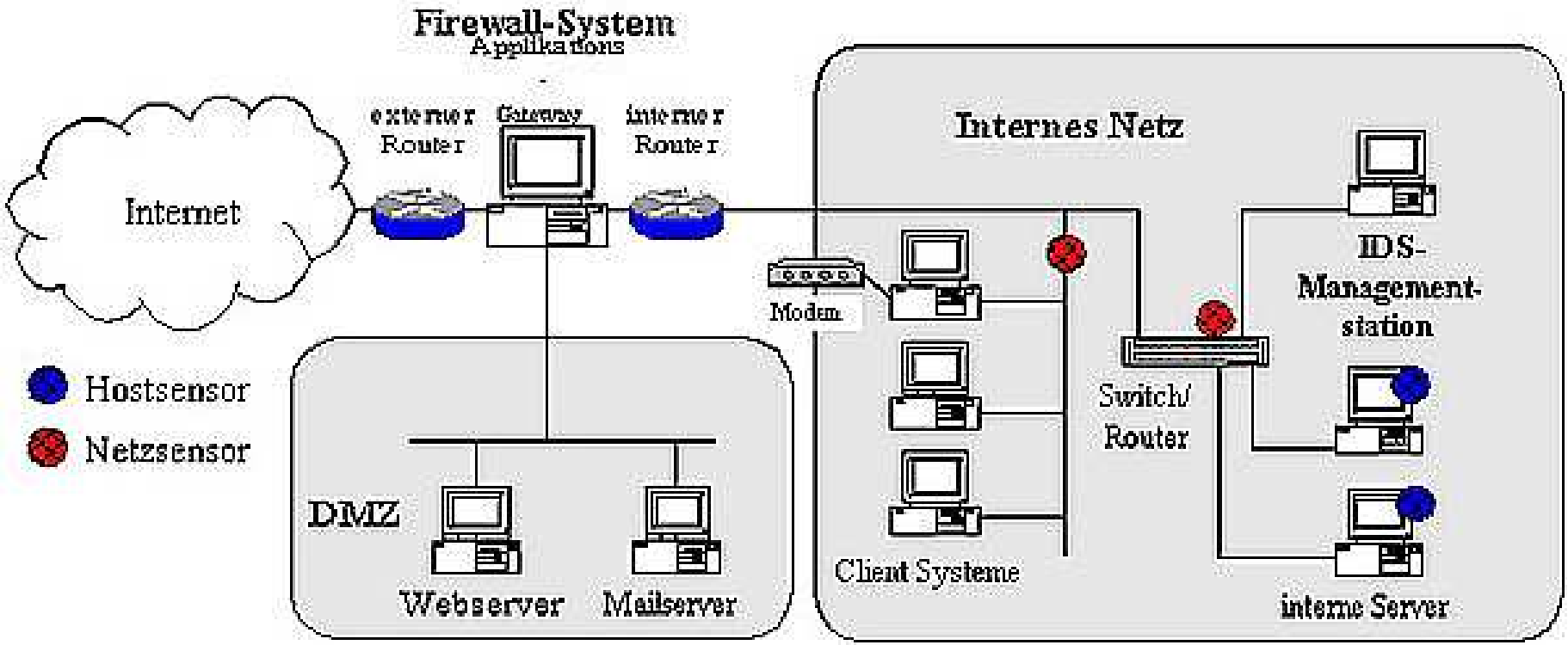
IDS-Erkennungstechniken

- Signaturerkennung
- statistische Analyse / Anomalieerkennung

Intrusion Detection Systeme (Forts.)

IDS-Architekturen

- Host-basiert
- Netzwerkbasiert
- Hybrid



(Quelle: https://www.bsi.bund.de/ContentBSI/Publikationen/Studien/ids02/gr4_htm.html)

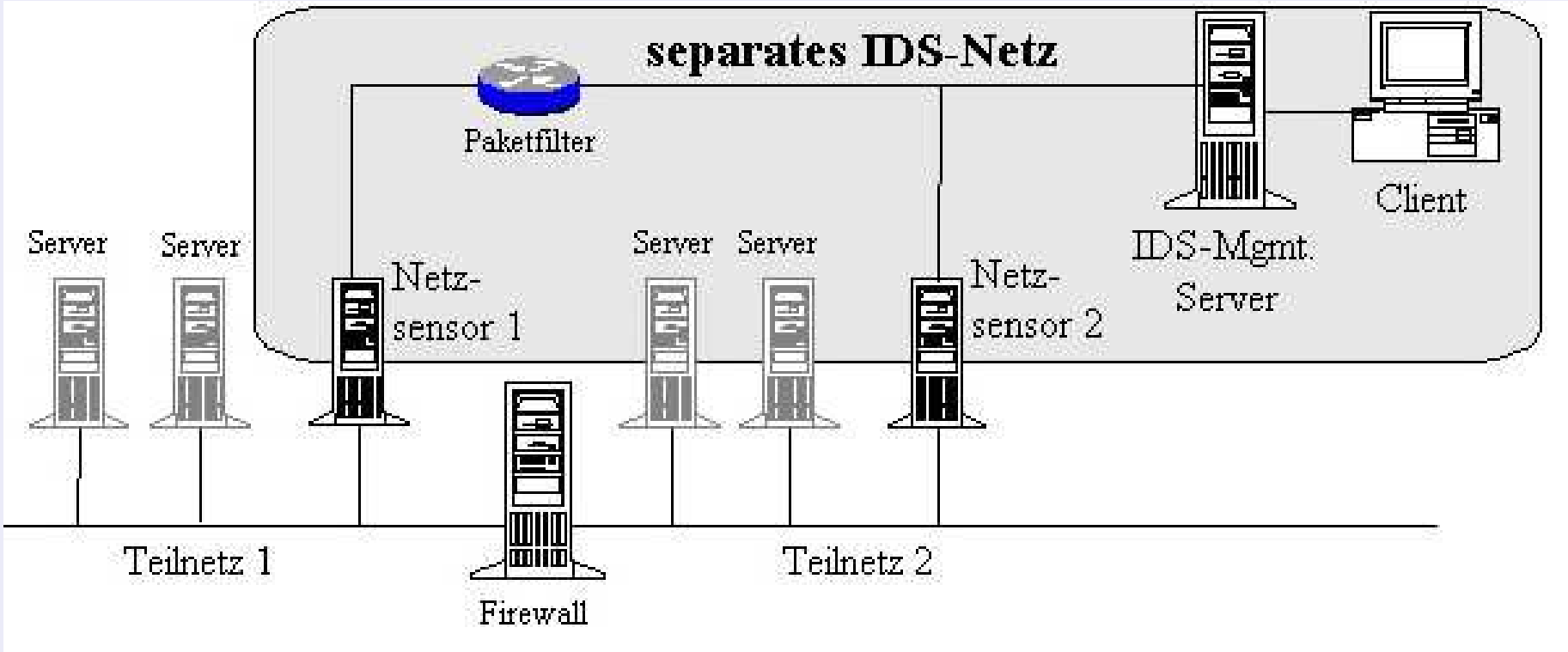
Intrusion Detection Systeme (Forts.)

Probleme

- fälschlicherweise gemeldete Angriffe (*false positives*)
- nicht gemeldete Angriffe (*false negatives*)
  (insb. bei neuartigen Angriffen)
- Echtzeitanforderung, insb. bei Hochgeschwindigkeitsnetzen
- Aufzeichnung bei Netzwerken mit Switches ($\Rightarrow$ spez. SPAN Port)
- Sensoren sollen unbeobachtbar sein (*stealth*)

Intrusion Detection Systeme (Forts.)

IDS Sensornetzwerk



(Quelle: https://www.bsi.bund.de/ContentBSI/Publikationen/Studien/ids02/gr4_htm.html)

Frage: *Welche Vorteile hat das?*