

osiv – open source initiative vorarlberg

Seminar 2: IT Security

DI Patrick Ritschel
Lukas Ruetz

Motivation: Sicherheit und Open Source

- „*Qualitätskriterium Geheimhaltung*“ ist kein Qualitätskriterium
- Begutachtung von Code durch unabhängige Dritte ist wünschenswert
- Das frühzeitige Finden von Fehlern vor der Produkteinführung erspart hohe Zusatzkosten
- Die Entwicklung von Produkten mit offenem Quellcode entspricht diesem Sicherheitsdenken
 - ◆ Offener Quellcode bringt nicht automatisch eine lizenz(kosten)freie Nutzung mit sich
- Negatives Beispiel
 - ◆ Geknackter GSM-Algorithmus A3/A8(Comp128-1)
- Open Source ist nicht nur Linux und OpenOffice!

Motivation: Veröffentlichung – Segen oder Fluch?

- Veröffentlichung ist nicht nur nützlich
- Exploits werden in Boards zur Verfügung gestellt
 - ◆ Kurz nachdem eine Sicherheitslücke bekannt wird
 - ◆ Mit Quellcode, meist ohne bösartige Funktion (*Proof-of-Concept*)
- Es darf nur kurze Zeit zwischen Veröffentlichung des Sicherheitsloches und Einspielen eines *Patches* vergehen
- Auch *Kiddiots* können so recht einfach gefährliche Software schreiben bzw. bedrohliche Angriffe ausführen
- Geheimhaltung der Systemarchitektur und der aktuell eingesetzten Softwareversion macht es für den Angreifer schwieriger
 - ◆ Er muss das Zielsystem (und somit dessen Fehler) erraten

- Einführung, Überblick
 - ◆ Dieser Teil
- Umgebung, Gebäudeschutz, Soziale Angriffe
 - ◆ Sicherheit bezieht sich nicht nur auf die Software eines Computers!
- Angriffe auf die Hardware, Attraktivität des Ziels, Honeypot-Ablenkung
 - ◆ Wer greift an – und warum?
- Viren, Würmer, Trojaner; Outlook und der dumme User, Buffer Under/Overrun, Stringformat, ...
 - ◆ Wie bösartige Software im System ausgeführt wird

- Kryptologie
 - ◆ Kryptographie und Kryptanalyse
- Betrachtung bekannter symmetrischer und asymmetrischer kryptographischer Algorithmen
 - ◆ Verfügbarkeit von Algorithmen: DES, AES, Blowfish, RSA, Ell. Kurven, Algorithmen (Authentifizierung, Schlüsseltausch, ...)
- Attacken auf Systeme mit Verschlüsselung
 - ◆ Brute-Force Attacke, Logische Attacken
- Organisatorisches zum Thema Kryptographie
 - ◆ PKI (PKCS#1-15), Zertifikate (X.509), CA-Server(-Software); Protokolle: PGP, SSL(TLS), SSH

- Grundlagen festverdrahteter Netzwerke
 - ◆ Der IP-Protokollstack
 - ◆ Switches, Hubs, Router
- Grundlagen von Netzwerksicherheit
 - ◆ Ports
 - ◆ Firewalls
- Angriffe auf das Netzwerk
 - ◆ Spoofing
 - ◆ DNS-Attacken
 - ◆ Sniffing
 - ◆ Session Hijacking

- Drahtlose Kommunikationsnetzwerke
 - ◆ WLAN
 - ◆ Bluetooth
- Bibliotheken zum Programmieren
 - ◆ Entwicklung eigener Sicherheitssoftware unter Zuhilfenahme von Open Source Bibliotheken
- Praktische Demonstration
 - ◆ Demonstration verschiedener Open Source Software aus dem Sicherheitsbereich
- Abschluss
 - ◆ Mit Diskussion

Einleitung: Sicherheit beim Haus

- Erkennen von Schwachstellen
 - ◆ Türen/Fenster/Dach/Briefkasten/...
- Schwachstellen ausbessern
 - ◆ Schloss/Zaun/Tresor/...
 - ◆ Das Prinzip des Schwächsten Glieds der Kette gilt!
- Einbruchserkennung
 - ◆ Wachhund/Alarmanlage/...



Einleitung: Motivation des Angreifers

- Angreifer möchten
 - ◆ Fremde Computer fernsteuern bzw. deren Daten ausspionieren („herumschnüffeln“)
 - ◆ Fremde Computer lahmlegen
 - ◆ Die Kommunikation (E-Mails, Verträge, Online-Banking, ...) anderer abhören oder ändern
 - ◆ Von fremden Computern aus andere Computer (z.B. NASA) angreifen, um nicht erkannt zu werden
 - ◆ Von fremden Computern aus Massenwerbemails (*Spam*) verschicken
- Und oft macht es den Angreifern einfach nur Spaß, andere zu ärgern oder mit Erfolgen anzugeben
 - ◆ Der Angreifer ist sich dann meist über die Folgen nicht im Klaren



Einleitung: Motivation des Verteidigers

- Der Verteidiger besitzt schützenswerte Güter
 - ◆ Daten und Informationen
 - ◆ Geräte und Einrichtungen
 - ◆ Gebäude und Anlagen
 - ◆ Menschenleben und Privatsphäre
 - ◆ Umwelt
- Das Internet ist besonders gefährlich
 - ◆ Der Zugriff ist nicht physisch
 - ◆ Die Verteidiger kennen die Angreifer oft nicht
 - ◆ Der Verteidiger kann auch ohne Absicht des Angreifers großen Schaden erleiden

Einleitung: Die „guten“ Hacker

- Ein *Hacker*
 - ◆ Umgeht die Sicherheitsmaßnahmen eines Computers bzw. Computersystems
 - ◆ Bewegt sich an der Grenze der Legalität!
- Ein *guter Hacker*
 - ◆ Hat gute Absichten
 - ◆ Ändert keine Daten
 - ◆ Macht die Betreiber des unsicheren Systems auf die Sicherheitslücken aufmerksam
 - ◆ Kann trotzdem angezeigt und bestraft werden!
 - Der *gute Wille* rechtfertigt keine Gesetzesübertretung.

Einleitung: Definition IT-System

- Geschlossenes oder offenes, dynamisches und technisches System mit der Fähigkeit zur Speicherung und automatisierten Verarbeitung von Informationen
 - ◆ Geschlossen
 - Bestimmter Hersteller- und Teilnehmerkreis
 - Beschränkte räumliche Ausdehnung
 - Zentrale Verwaltung
 - Inkompatibilitäten
 - ◆ Offen
 - Vernetzt
 - Physisch verteilt
 - Heterogen
 - An Standards orientiert

Einleitung: Definition *Schutz und Sicherheit*

- *Safety*, Funktionssicherheit
 - ◆ Schutz vor Fehlfunktionen
 - ◆ Der Übergang von Zustand A \rightarrow B ist definiert und wird eingehalten
- *Security*, Informationssicherheit
 - ◆ Schutz vor ungültigen Systemzuständen
 - ◆ Jeder Systemzustand X[i] ist definiert
- *Protection*, Datensicherheit
 - ◆ Schutz vor Datenverlust
 - ◆ Daten gehen nicht verloren
- *Privacy*, Datenschutz
 - ◆ Informationelles Selbstbestimmungsrecht
 - ◆ Eine Person weiß und bestimmt, wer welche Daten über sie besitzt

Einleitung: Sicherheitsziele

- *Authenticity*, Authentizität
 - ◆ Prüfung der Echtheit eines Subjektes oder Objektes
- *Integrity*, Integrität
 - ◆ Gewährleistung unmanipulierter Datenobjekte
- *Confidentiality*, Vertraulichkeit
 - ◆ Keine unautorisierte Informationsgewinnung
- *Non-Repudiation*, Nicht-Leugbarkeit, Verbindlichkeit
 - ◆ Kein Abstreiten getätigter/unterlassener Handlungen möglich

Einleitung: Weitere Sicherheitsziele

- *Availability*, Verfügbarkeit
 - ◆ Das System ist zum Betreiben der Anwendung verfügbar (Wartungszeiten fallen den Anwendern nicht auf)
- *Accountability*, Abrechenbarkeit
 - ◆ Überwachung und Protokollierung sämtlicher Handlungen im System
- *Access Control*, Zugriffssteuerung
 - ◆ Der Zugriff auf Objekte kann auf definierte Subjekte (bzw. Gruppen von Subjekten/Rollen) eingeschränkt werden

Einleitung: Gefahrenbereiche

- Menschliches Versagen
 - ◆ Irrtum, Fahrlässigkeit, Nachlässigkeit, Handhabungsfehler
- Technisches Versagen
 - ◆ Hardware, Software, Leitungen, Hilfsaggregate
- Physische Bedrohungen
 - ◆ Feuer, Blitz, Einbruch, Ausschreitungen, Stromversorgung
- Höhere Gewalt
 - ◆ Hochwasser, Erdbeben, Streik, Flugzeugabsturz, Kriegerische Ereignisse
- **Computer-Anomalien**
 - ◆ **Viren, Trojanische Pferde, Logische Bomben, Würmer**
- **Computerkriminalität**

Einleitung: Computerkriminalität

- Diebstahl
 - ◆ Rechenzeit
 - ◆ Daten
 - ◆ Programme
 - ◆ Geräte
- Betrug, Unterschlagung
- Spionage
- Sabotage, Vandalismus
- Hacking
- Phishing
 - ◆ Ausspionieren von Bankinformationen über *Social Engineering*
- Verstöße gegen das Datenschutzgesetz

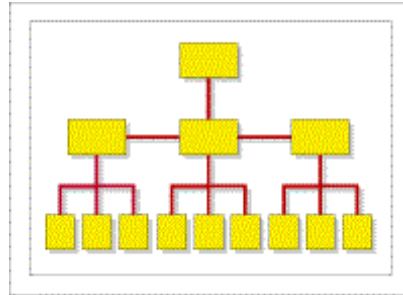
Einleitung: Bedrohungen und Angriffe

- Aktionen, die die Sicherheitseigenschaften eines Systems in Frage stellen
 - Unterschiedliche Gewichtung der Sicherheitseigenschaften
 - Durchführen von Bedrohungs- und Risikoanalysen
 - Formulieren einer Sicherheitsstrategie
-
- Angriffe nutzen oft bestehende Sicherheitslücken aus (*Exploits*)

Sicherheitsmanagement: Übergeordnete Komponenten



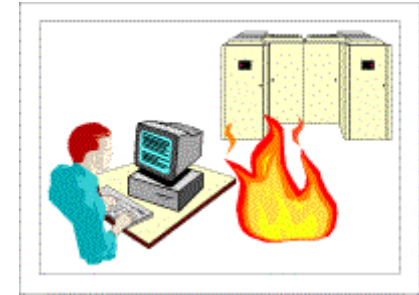
IT-Sicherheitsmanagement



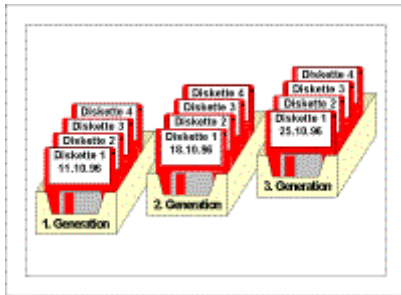
Organisation



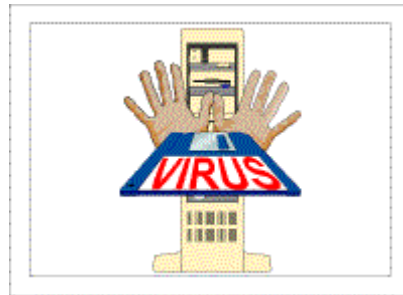
Personal



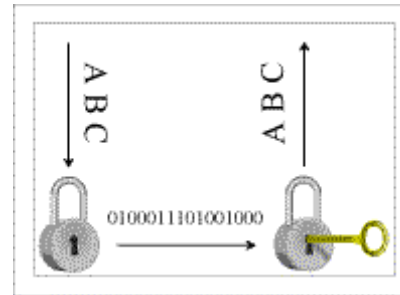
Notfallvorsorge-Konzept



Datensicherungs-Konzept



Computer-Virenschutzkonzept

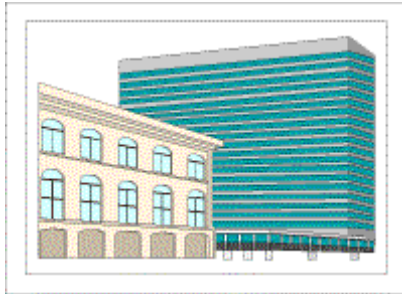


Kryptokonzept

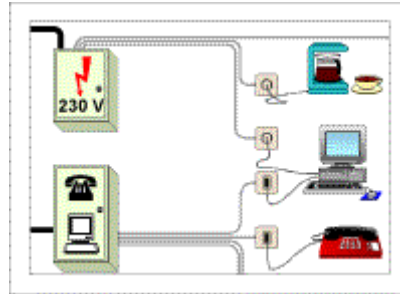


Behandlung von Sicherheitsvorfällen

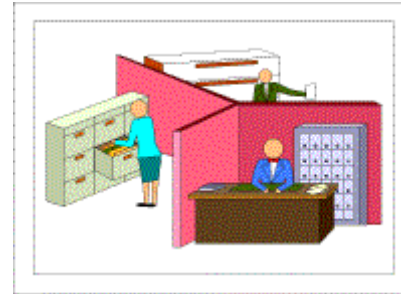
Sicherheitsmanagement: Infrastruktur



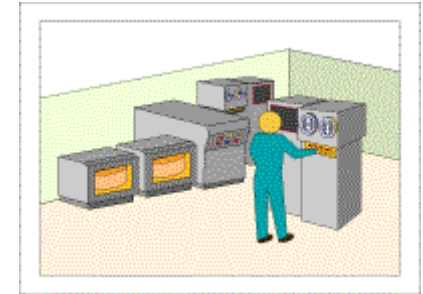
Gebäude



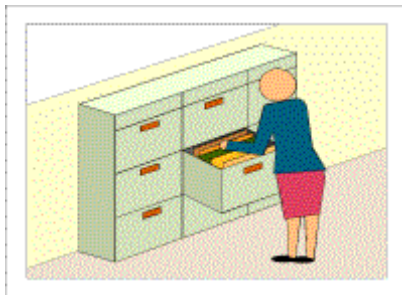
Verkabelung



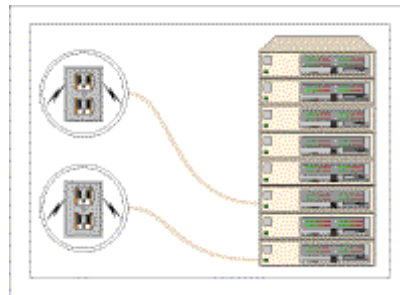
Bürraum



Serverraum



Datenträger-
archiv



Raum für techn.
Infrastruktur



Schutzschränke



Häuslicher
Arbeitsplatz

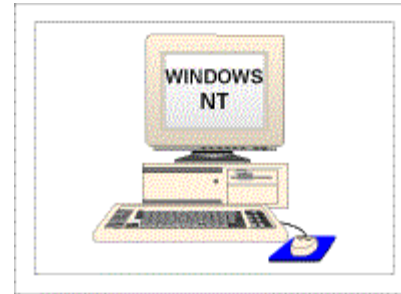
Sicherheitsmanagement: Der Alltag ist inhomogen



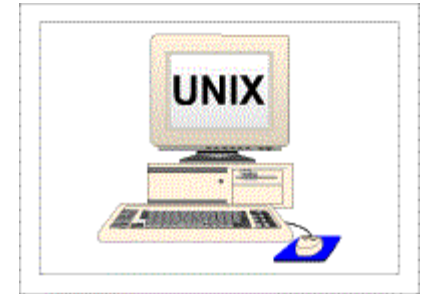
DOS-PC



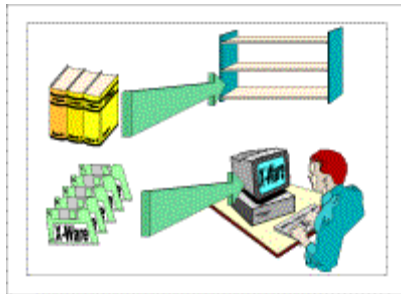
Notebook



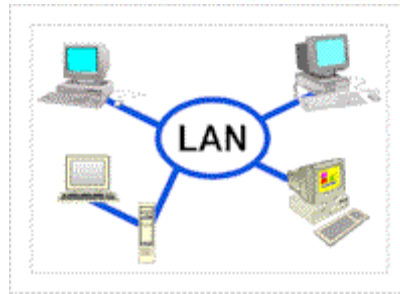
Windows-System



Unix-System



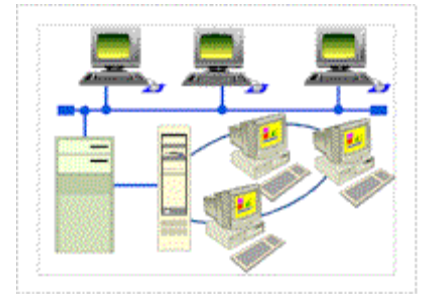
Vielfältige
Software-
komponenten



Servergestütztes
Netz

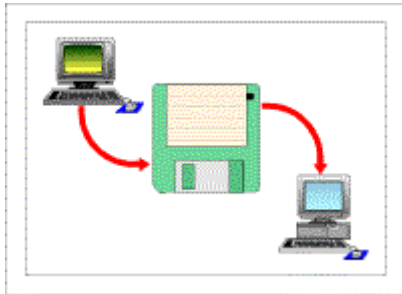


Peer-to-Peer-
Netz

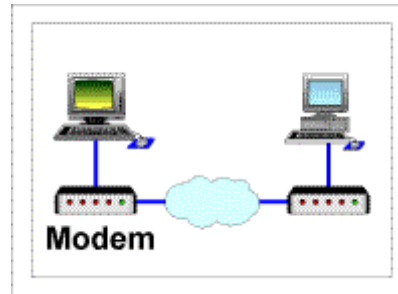


Heterogene
Netze

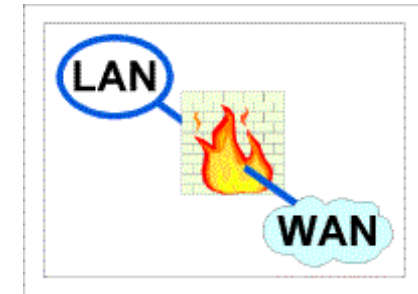
Sicherheitsmanagement: Datenübertragungseinrichtungen



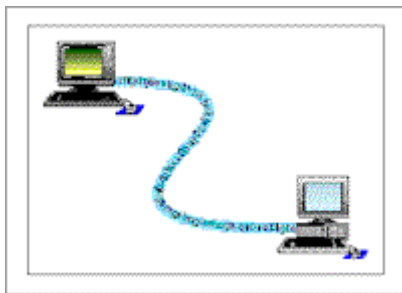
Datenträgeraustausch



Modem



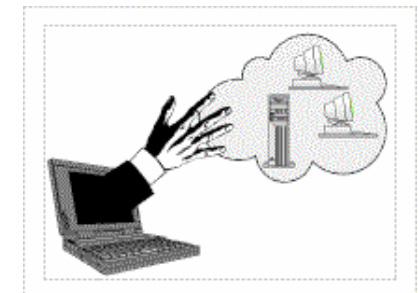
Firewall



E-Mail

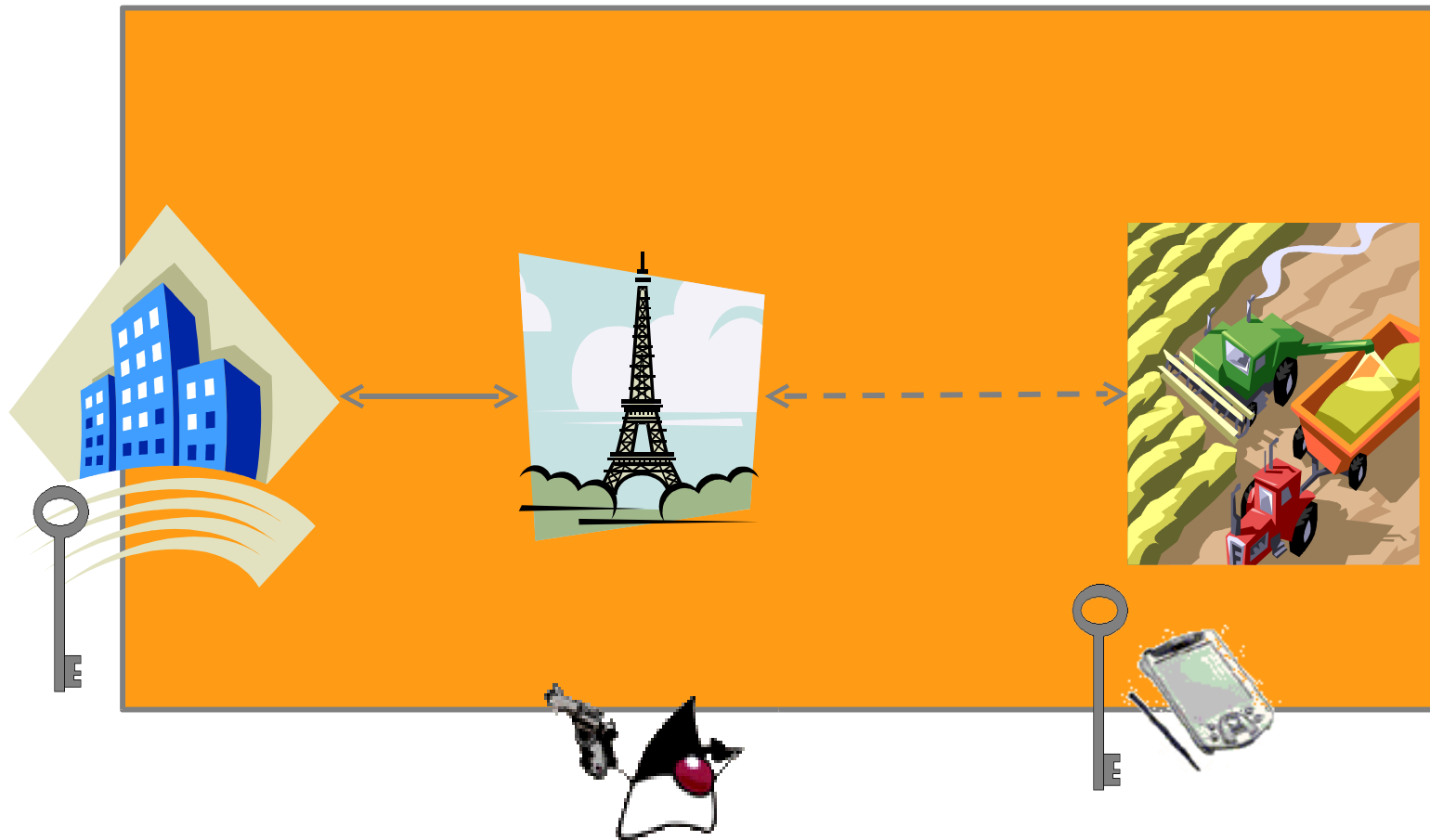


WWW-Server



Remote Access

Angriffe: Wo wird angegriffen?



Angriffe: Honeytrap

- Der *Honeytrap* (*Honigtopf*)

- ◆ Ist ein System, das zur Ablenkung im Netzwerk installiert wird
- ◆ Ein Honeytrap-Server trägt beispielsweise einen Namen, der mit Finanztransaktionen oder sehr geheimen (und interessanten) Firmendaten zusammenhängt
- ◆ Der Hacker greift dann eher diesen Server an als einen, auf dem beispielsweise Word-Vorlagen vermutet werden



- Das Honeytrap-System wird speziell beobachtet

- ◆ Dadurch ist eine frühzeitige Erkennung von Angriffen möglich
- ◆ Es läßt sich auch die Methodik des Angriffes erkennen
- ◆ Als ganzes Netzwerk auch *Honeynet* genannt

Angriffe: Social Engineering

- Der Anwender (meist ein Mensch) ist direkt angreifbar
- Deshalb werden menschliche Schwächen für Angriffe ausgenutzt
 - ◆ Faulheit
 - Es ist leichter, OK gleich anzuklicken als einen Dialog durchzulesen
 - ◆ Neugier
 - Bewußtes Eingehen kleiner Risiken in Erwartung positiver Ereignisse
 - ◆ Andere Eigenschaften, die sich ausnutzen lassen
 - Rachsucht
 - Habgier
 - Freundschaftsdienste
 - Triebe
 - Suchtverhalten
 - Liste nicht vollständig ;-)



Angriffe auf die Hardware

- Die Hardware von Systemen kann auf unterschiedliche Arten untersucht werden
 - ◆ Der Speicher eines Computers kann ausgelesen werden
 - ◆ Die Hardwarebausteine können mikroskopisch untersucht werden
 - ◆ Aus den gewonnenen Untersuchungsergebnissen (Codestücke, Schlüssel) können weitere Angriffe abgeleitet werden
 - ◆ Diese Attacken werden *Side-Channel-Attacks (Seitenkanalattacken)* genannt
- Die Hardware kann aber auch manipuliert werden durch
 - ◆ Strom- oder Spannungsimpulse
 - ◆ Ändern der anliegenden Taktfrequenz
 - ◆ Freilegen des Siliziums und Anblitzen mit einer Lampe/einem Laser
 - ◆ Erhitzen/Abkühlen des Chips
 - ◆ ...

Angriffe auf die Hardware: Kompromittierende Abstrahlung

- Ermöglicht das Auslesen von Daten ohne direkten Kontakt mit dem Gerät
 - ◆ Über *Induktion* können Systeme auch gestört werden
 - ◆ Elektromagnetische und akustische(!) Abstrahlung
- Am besten eignet sich der Röhrenbildschirm
 - ◆ Bei günstigsten Eigenschaften > 100m, auch vom mobilen Fahrzeug aus
 - ◆ Bis 20 Bildschirme sind auseinanderhaltbar
 - ◆ Verschlüsselung nützt hier nichts
 - ◆ Gesetzlich relativ unbedenklich
- Technische Signaldaten
 - ◆ Frequenz idealerweise 150-200 Mhz
 - ◆ Datenrate > 100 kBit/s
 - ◆ Kabel sind nur in unmittelbarer Nähe messbar
 - ◆ Lichtleiter sind hier eine sehr gute Alternative
 - ◆ Es gibt Schutztapeten und Schutzfenster im Handel

Hardwareangriffe:

Mobile Geräte sind besonders gefährdet

- Mobile Geräte
 - ◆ Können leichter gestohlen werden
 - ◆ Können deshalb leichter untersucht werden
- Verbesserung durch Nutzung von *Smartcards*
 - ◆ *Secure Tokens*, die speziell gegen Angriffe gerüstet sind
 - Durch Schutzgitter über der Siliziumschicht
 - Durch Sensoren für Temperatur, Spannung, Strom, Taktfrequenz, Licht, usw.
 - ◆ Verfügen über hochsichere Betriebssysteme



- **Verschiedene Methoden**
 - ◆ Optische Analyse
 - ◆ Speicher direkt auslesen/ verändern
 - ◆ Reverse Engineering
- **Gegenmaßnahmen von HW-Hersteller**
 - ◆ Verschlüsselte Ablage von Programm und Daten
 - ◆ Verschleiern des Layouts
 - ◆ Schutzschichten auf der Oberfläche
 - ◆ Sehr feine Halbleiterstrukturen
 - ◆ Scrambled Bus Logic
 - ◆ Doppelte Ausführung jedes Gatters in inverser Schaltungstechnik, um Rauschfreiheit in Bezug auf Stromaufnahme zu gewährleisten

Hardwareangriffe: Elektrische Attacken

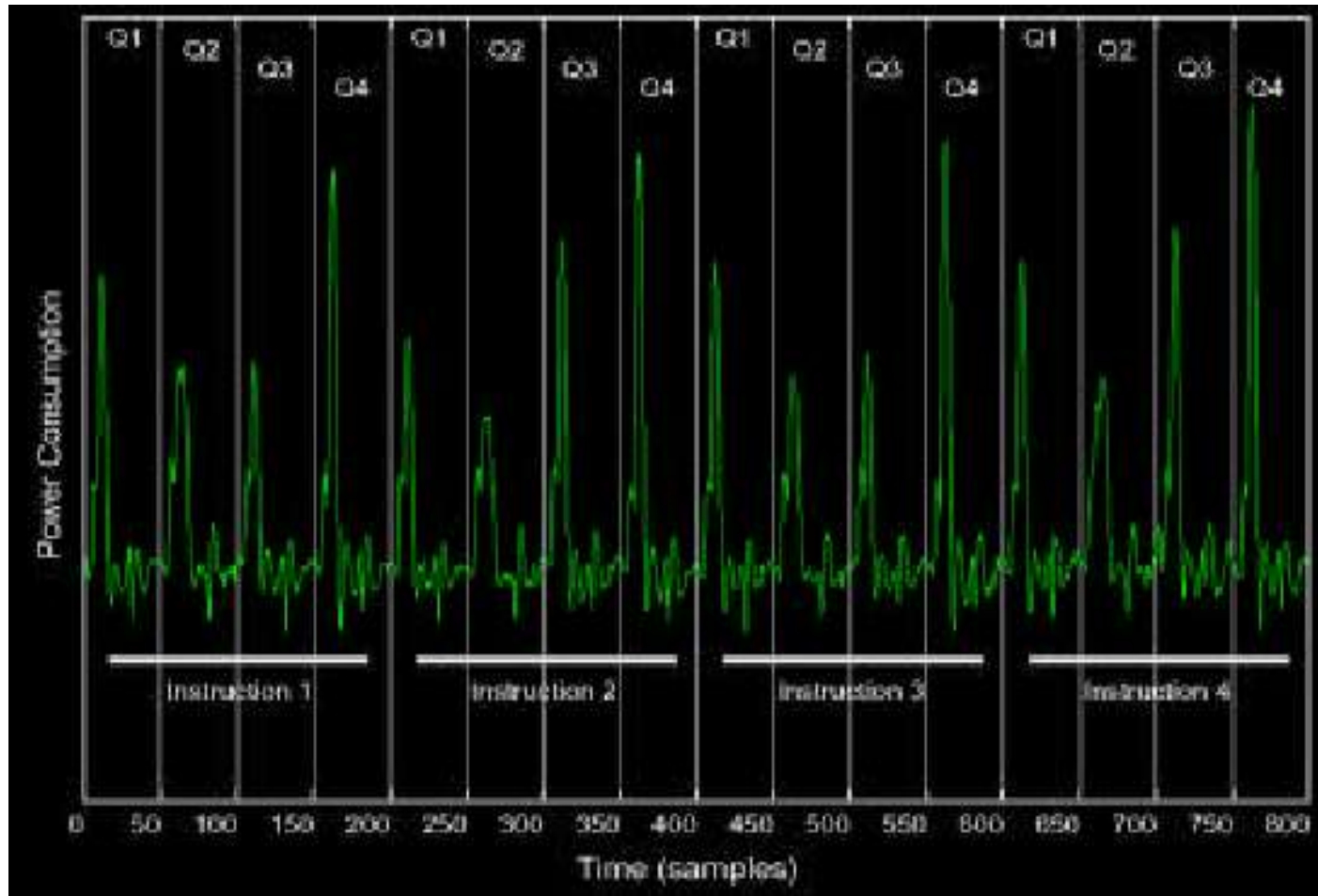
- Probing
 - ◆ Busleitungen während des Betriebs abgreifen und Daten visualisieren
 - ◆ Abwehr: scrambled bus, Schutzschichten
- Physical Stressing
 - ◆ Änderung des Programmablaufs durch Störung (Licht, Stromimpulse, ...)
 - ◆ Abwehr: Sensoren

Analytische Attacken: Strom

- Messung der Stromaufnahme (oder elektromagnetischen Abstrahlung)
- SPA (Simple Power Analysis)
 - ◆ Identifizierung einzelner Instruktionen und Operanden anhand eines statischen Stromprofils
 - ◆ Abwehr: Rauschen oder Glättung
- DPA (Differential Power Analysis)
 - ◆ Statistische Analyse einiger 1000 Messungen
 - ◆ Eliminierung des Rauschens durch Mittelwertbildung
 - ◆ Abwehr: Zufällige Zeitverschiebung
 - Z.B. durch Einfügen von „leeren“ Prozessortakten
 - Oder Veränderung der Auslastung von Pipelines im Prozessor

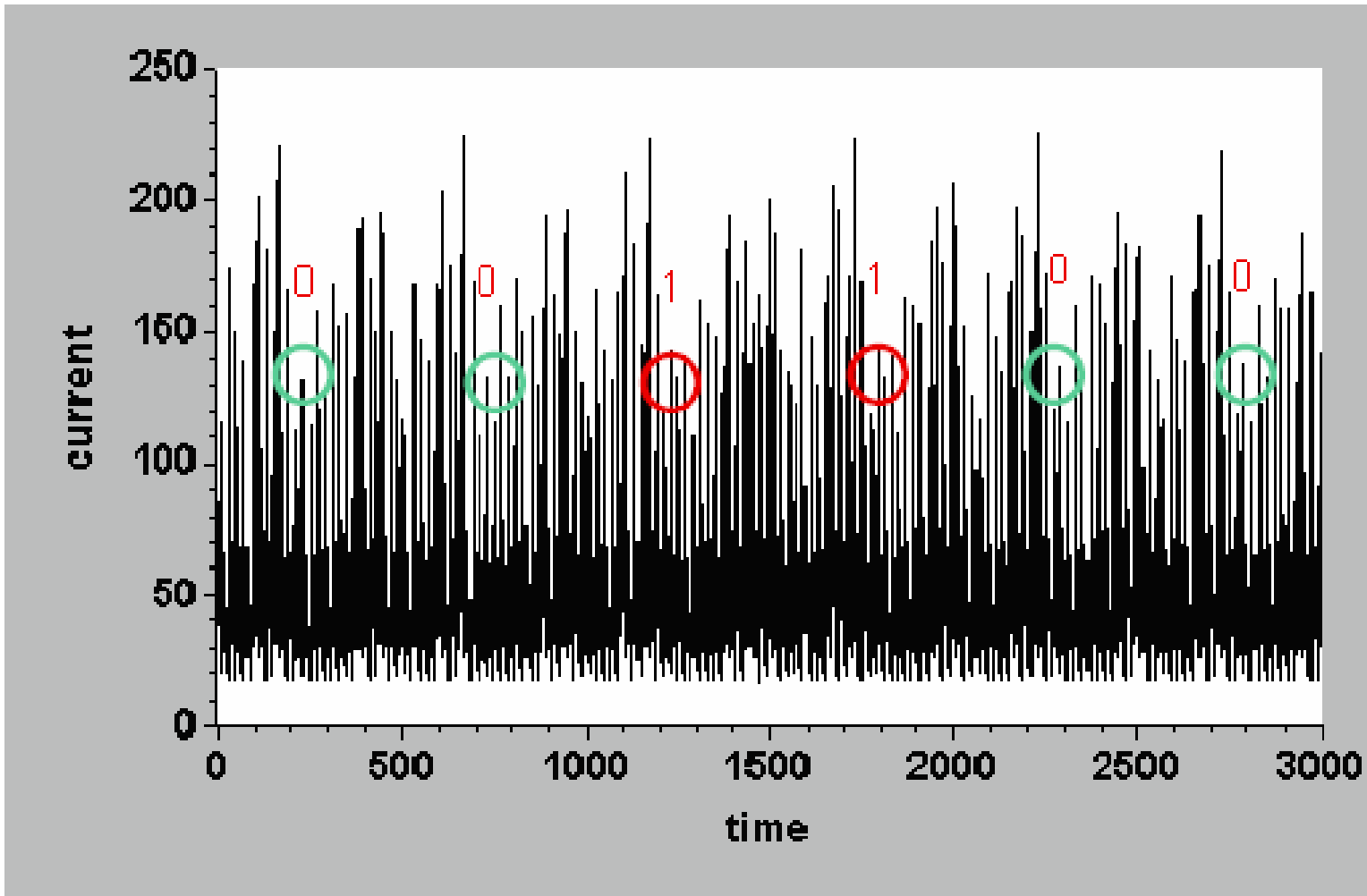
Analytische Attacken: SPA

Instruktionen eines Microchip PIC Prozessors



Analytische Attacken: SPA

Während der Berechnung eines DES



Analytische Attacken: Timing

- ◆ Die Verarbeitungszeit oft abhängig von Schlüssel und Eingabedaten (nicht *rauschfrei*)
- ◆ Einfaches Beispiel:
- ◆ Zeichenvergleich: Abbruch der Schleife bei erster Unstimmigkeit
- ◆ Abwehr: Gleiche Zeit für alle möglichen Verarbeitungen des Algorithmus
- ◆ Die Gegenmaßnahmen im Code sind nicht immer trivial zu lösen
 - Ein `if (i != 0)` in Assembler übersetzt benötigt normalerweise nicht für beide genommenen Zweige gleich lange (`JZ` und `JMP` sind nicht dieselben Befehle, haben also nicht unbedingt dieselbe Laufzeit)
 - `MOV Acc, @i`
 - `JZ not_null`
 - `; hier Code für i == 0`
 - `JMP naechster_schritt`
 - `not_null: ; hier Code für i != 0`
 - `naechster_schritt:`

Das Internet ist nicht sicher

- Das Internet wurde als Zusammenschluß von Universitätsrechnern entwickelt
- Sicherheit war beim Design kein Thema
- Dementsprechend unsicher ist das Internet heute
- Nächste Generation IPv6 berücksichtigt auch Aspekte der Sicherheit
- An das Internet sind (geschätzt) über **250.000.000** Benutzer angeschlossen
 - ◆ Selbst wenn **99,9%** davon redliche Anwender sind, verbleiben
250.000
 - ◆ Hacker, Freaks, Viren-Coder, Cracker, Cyber Punks, Script-Kiddies bzw. Script-Kiddiots, usw.

Das Internet ist nicht sicher: Kosten der Angriffe

- Die Angriffe – oft zum Spaß verübt – kosten die Wirtschaft erheblich Geld
 - ◆ Direkt über Schäden (auch verlorene Arbeitszeit ist ein Schaden)
 - ◆ Indirekt über Softwarekäufe und Wartungsmaßnahmen
- Beispiele
 - ◆ Phishing-Mails (betrügerische E-Mails von „Kreditinstituten“ oder ähnlichem, die auf das Zurücksenden von Kreditkartendaten abzielen)
 - Verursachten 2003 US\$ 1,2 Milliarden Schaden
 - Nehmen seit Anfang 2004 enorm zu
 - ◆ Laut *Network Associates* in Europa jährlicher Schaden von € 22 Milliarden durch Downtime infolge von Viren
 - Eine Virenattacke kostet ein Unternehmen im Schnitt € 5.000,-

Viren, Würmer und Trojaner: Klassifizierung

- Generelle Zusammenfassung unter dem Kunstwort *Malware*
- Oft auch *Computeranomalien* genannt
- Ein *Virus*
 - ◆ ist ein Codeabschnitt, der sich in einen Host einschließlich des Betriebssystems einschleust, um sich zu verbreiten. Er kann nicht unabhängig existieren. Stattdessen ist ein Wirtprogramm erforderlich, um den Virus zu aktivieren.
 - ◆ **Das** Virus med. <-> **Der** Virus inform.
- Ein *Wurm*
 - ◆ ist ein Programm, das unabhängig ausgeführt werden kann und die Ressourcen der Wirtssysteme von innen verzehrt, um sich zu erhalten. Der Wurm kann dann neue funktionsfähige Kopien von sich selbst auf weiteren Wirtssystemen erzeugen

Viren, Würmer und Trojaner: Klassifizierung

- Ein *trojanisches Pferd* bzw. *Trojaner*
 - ◆ umfasst Code, der sich als harmloses Programm tarnt, um sich dann auf unerwartete und in der Regel schädliche Art zu verhalten.
- Eine *logische Bombe*
 - ◆ ist eine Rachemaßnahme von Programmierern, die – meist auf Grund von Differenzen mit der Geschäftsleitung – zerstörende oder schädigende Mechanismen in das Programm einbauen
- Ein *Hoax-Mail*
 - ◆ ist eine (Viren-)warnung, die sich auf einen unechten Virus bezieht und sich lawinenartig ausbreitet

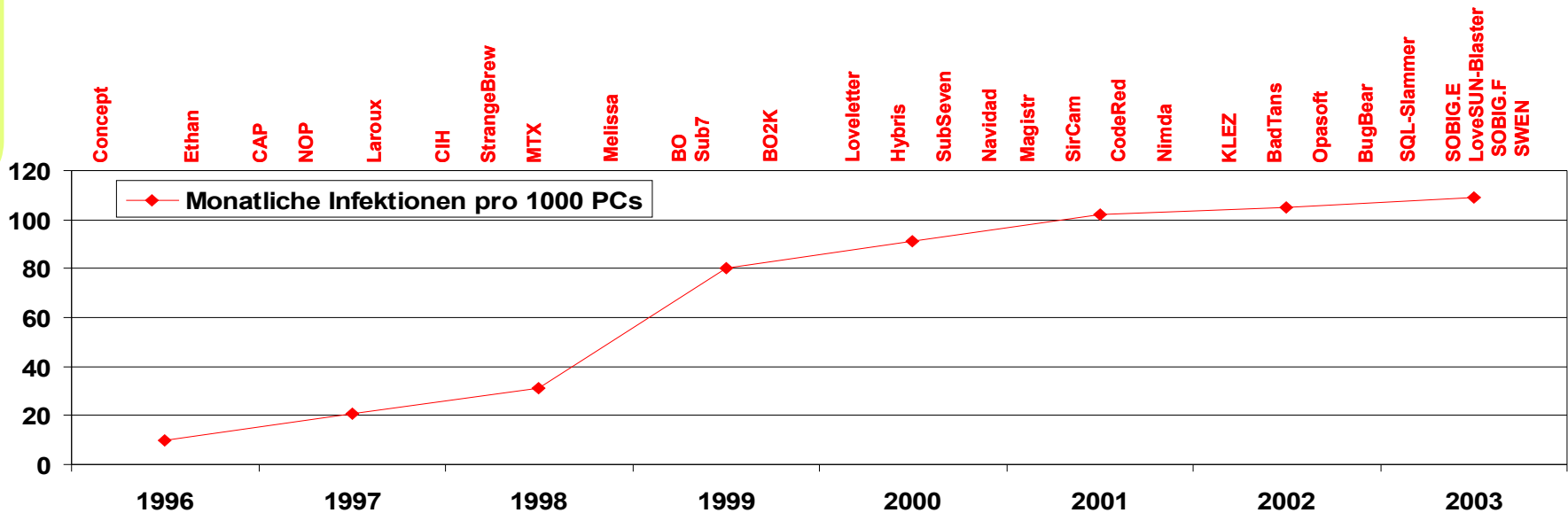
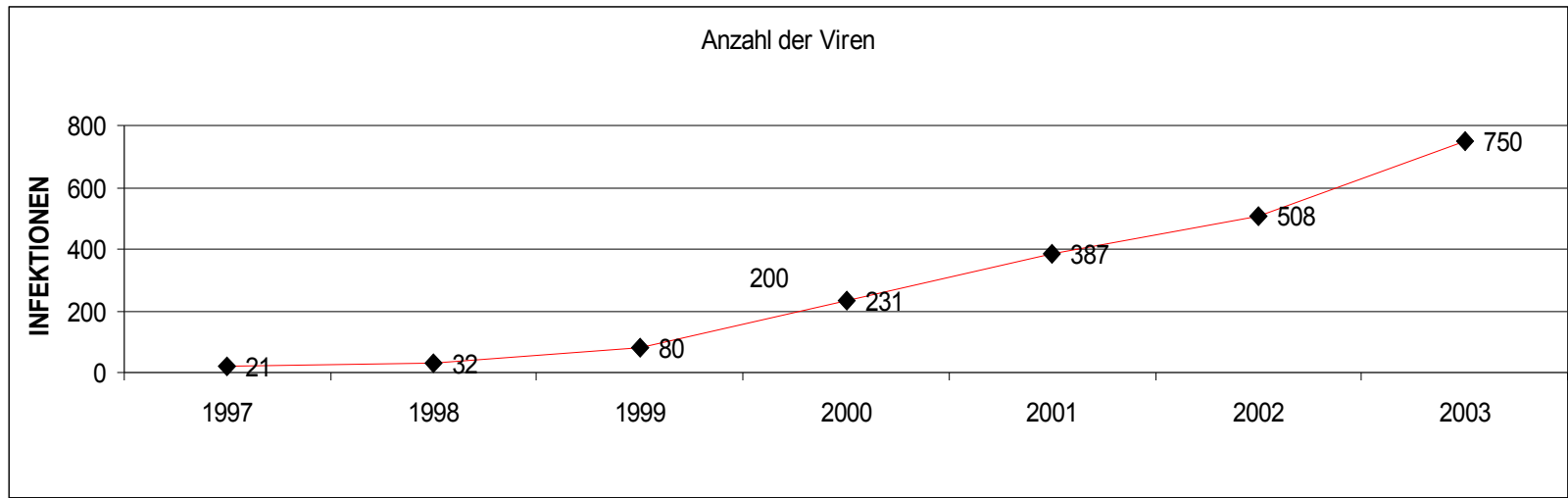
Malware: Grobgliederung

- ◆ Boot-Viren
 - ParityBoot/Angelina
- ◆ File-Viren
 - Tequilla, CIH, NIMDA, Magistr
- ◆ Makro-Viren
 - CAP, Laroux, Ethan
- ◆ VBS (Skript)-Viren
 - Loveletter, Fremlink
- ◆ Würmer
 - CodeRed, Slammer, Blaster, Sobig
- ◆ Trojaner
 - NetBus, BackOrifice, SpyBot
- ◆ Sonstige
 - IRC.StagesA, Toadie, Shell-Scrap

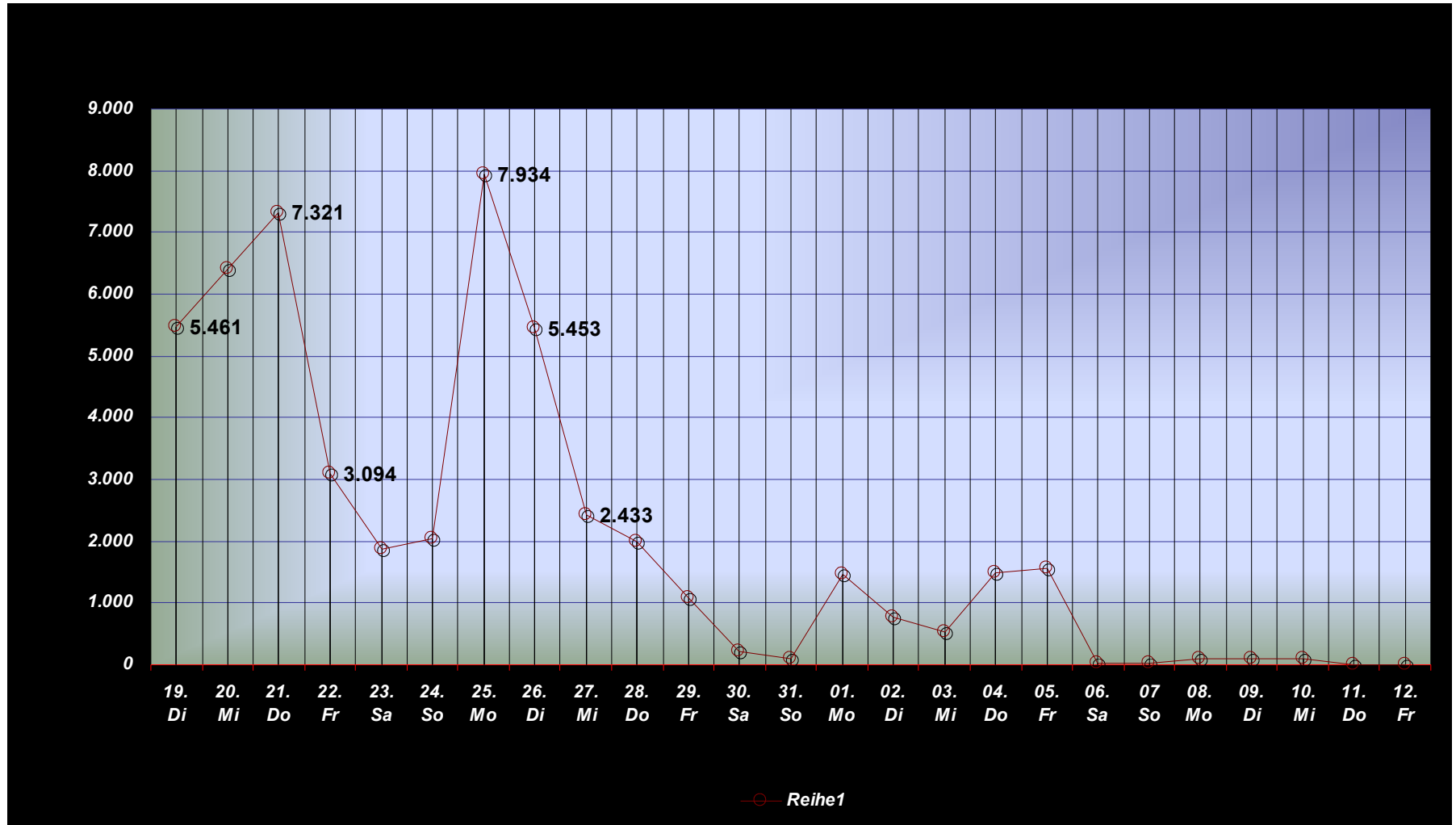
Malware: Die häufigsten Infektionswege

Diskette: sales demo	16	1	0
Diskette: shrink-wrapped software	3	1	0
Diskette: LAN manager/supervisor	3	1	0
Diskette: repair/service person	15	2	0
Diskette: malicious person	3	6	2
Diskette: other	78	9	0
CD: software	3	1	1
Download: BBS / ISP	99	31	7
Download: FTP, BBS, host	15	12	18
Email attachment	501	828	901
Automated software distribution	3	12	16
WWW: browsing	18	27	31
Other	9	12	11
None specified	15	7	4
Don't know	66	17	6
	1.000	1.000	1000

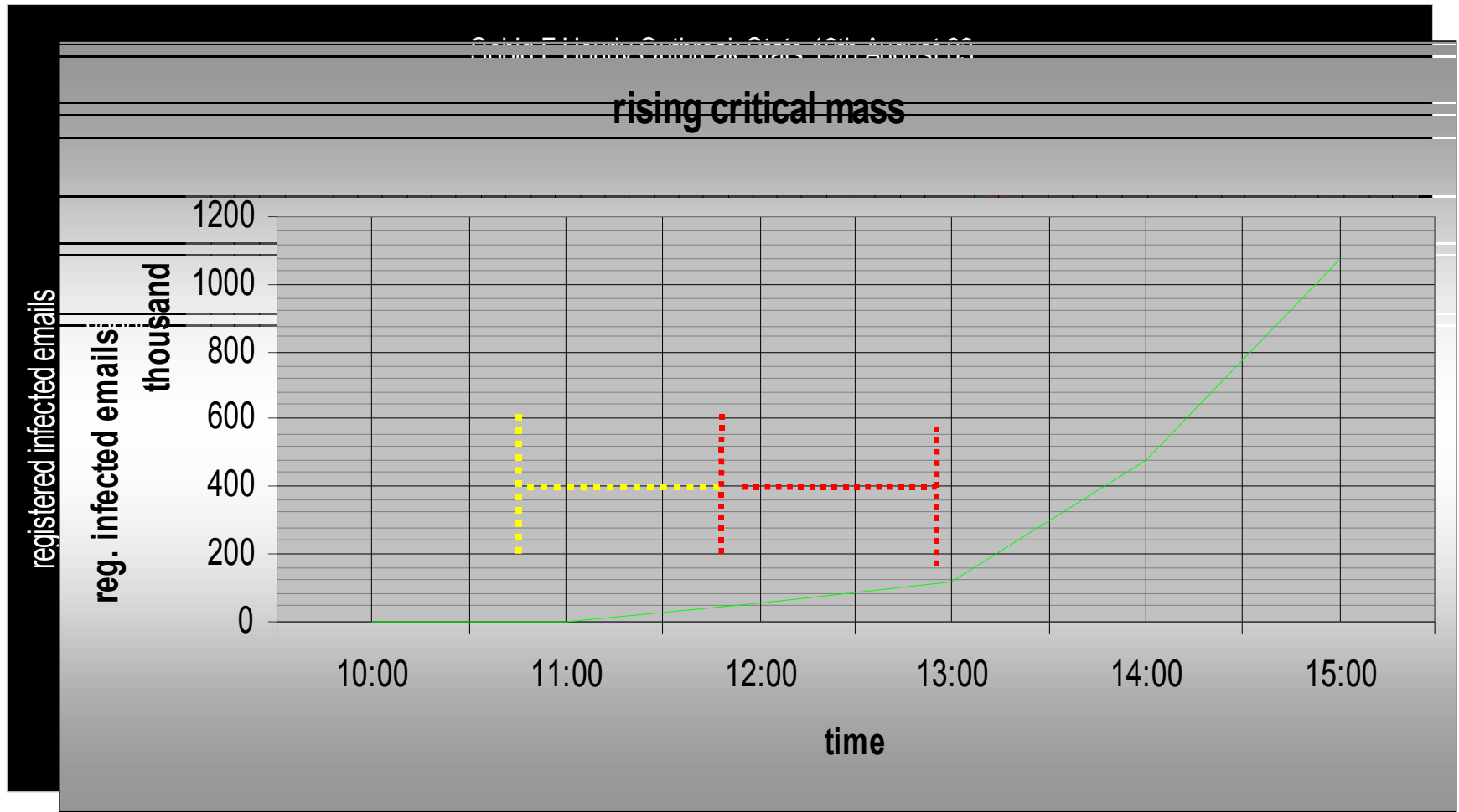
Malware: Virengefahr



Malware: Sobig.F - Verlauf

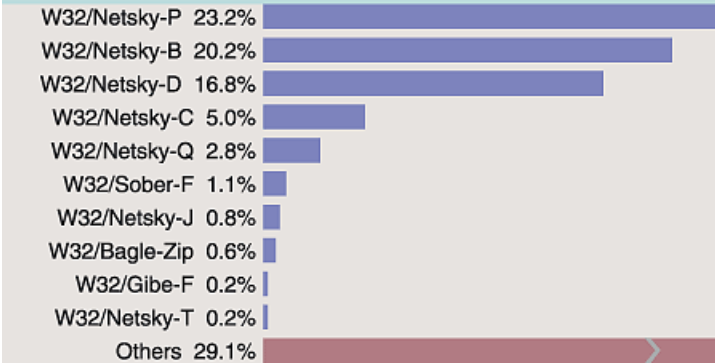


Malware: Wesentlicher Faktor: Antwortzeit

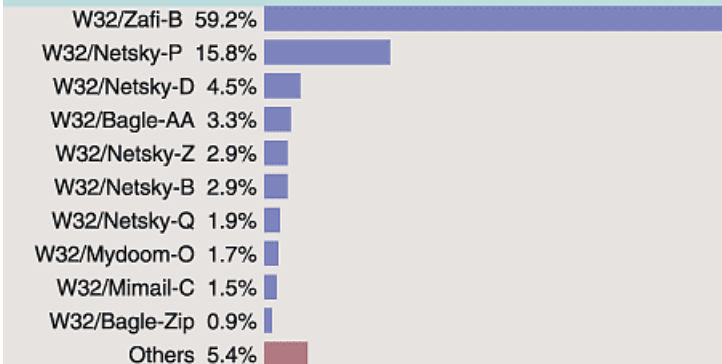


Malware: Aktuelle „Top 10“

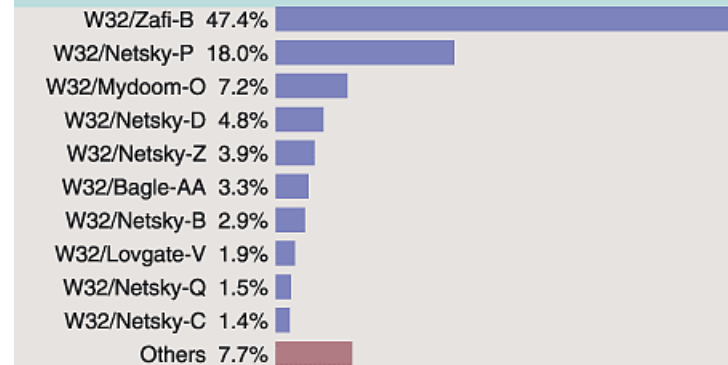
Top ten viruses reported to Sophos in April 2004



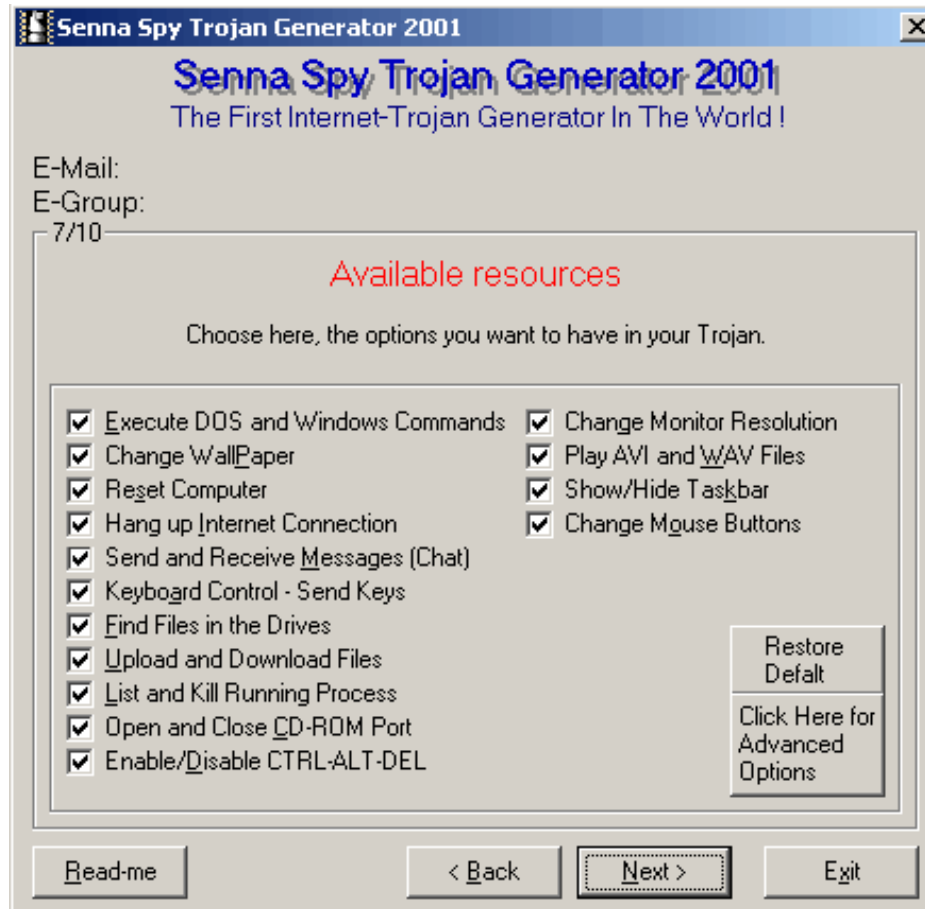
Top ten viruses reported to Sophos in July 2004



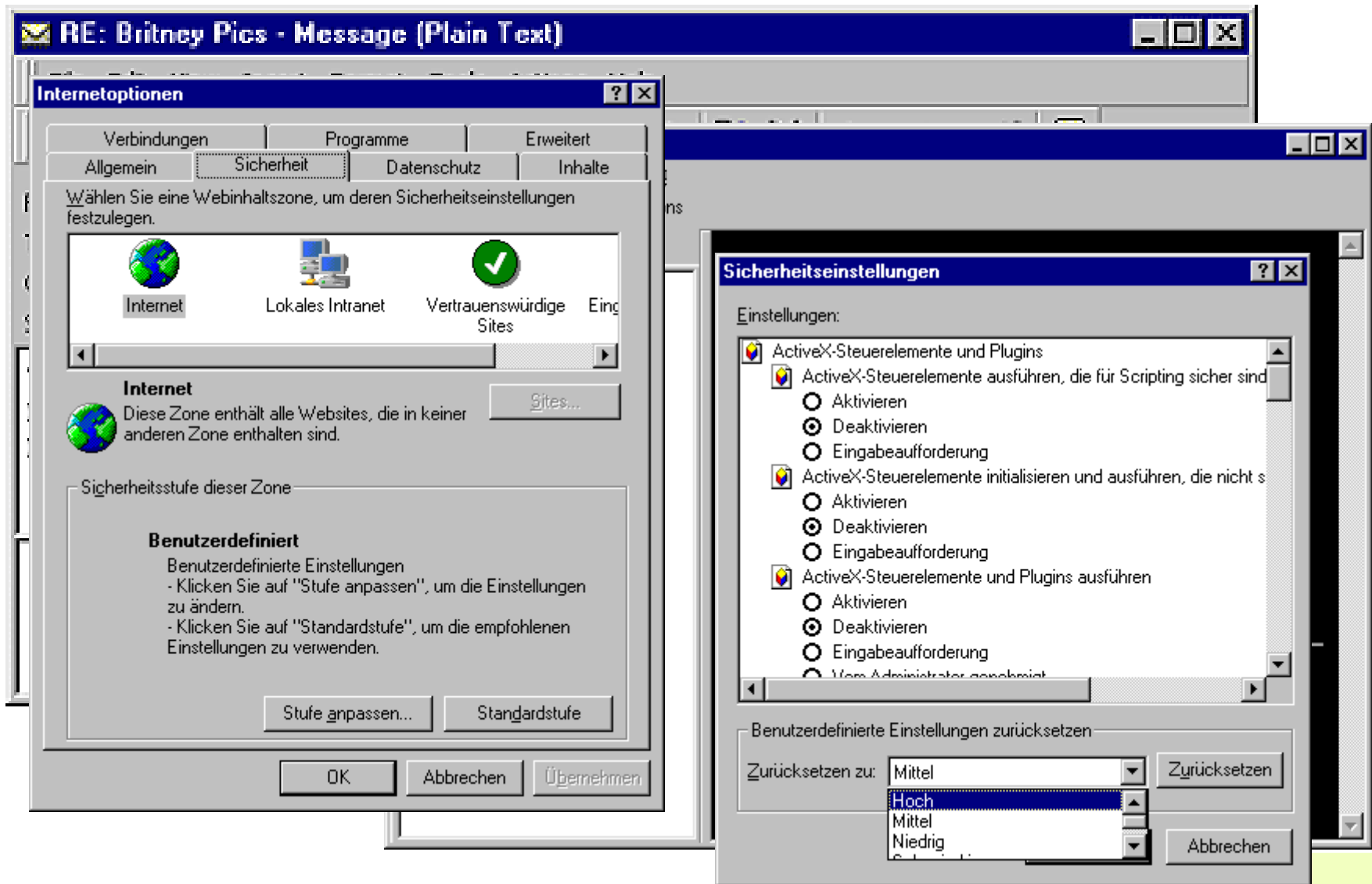
Top ten viruses reported to Sophos in August 2004



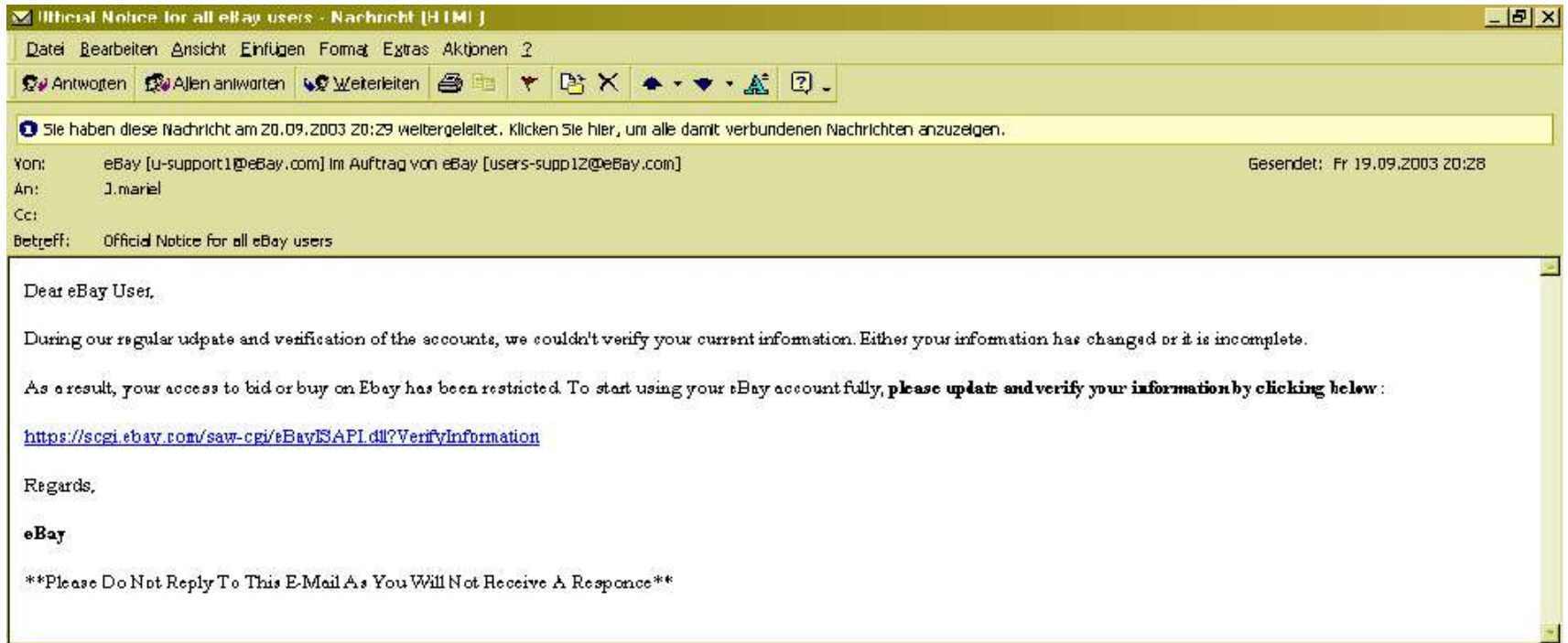
Malware: Generatoren für Dummys und Idiots



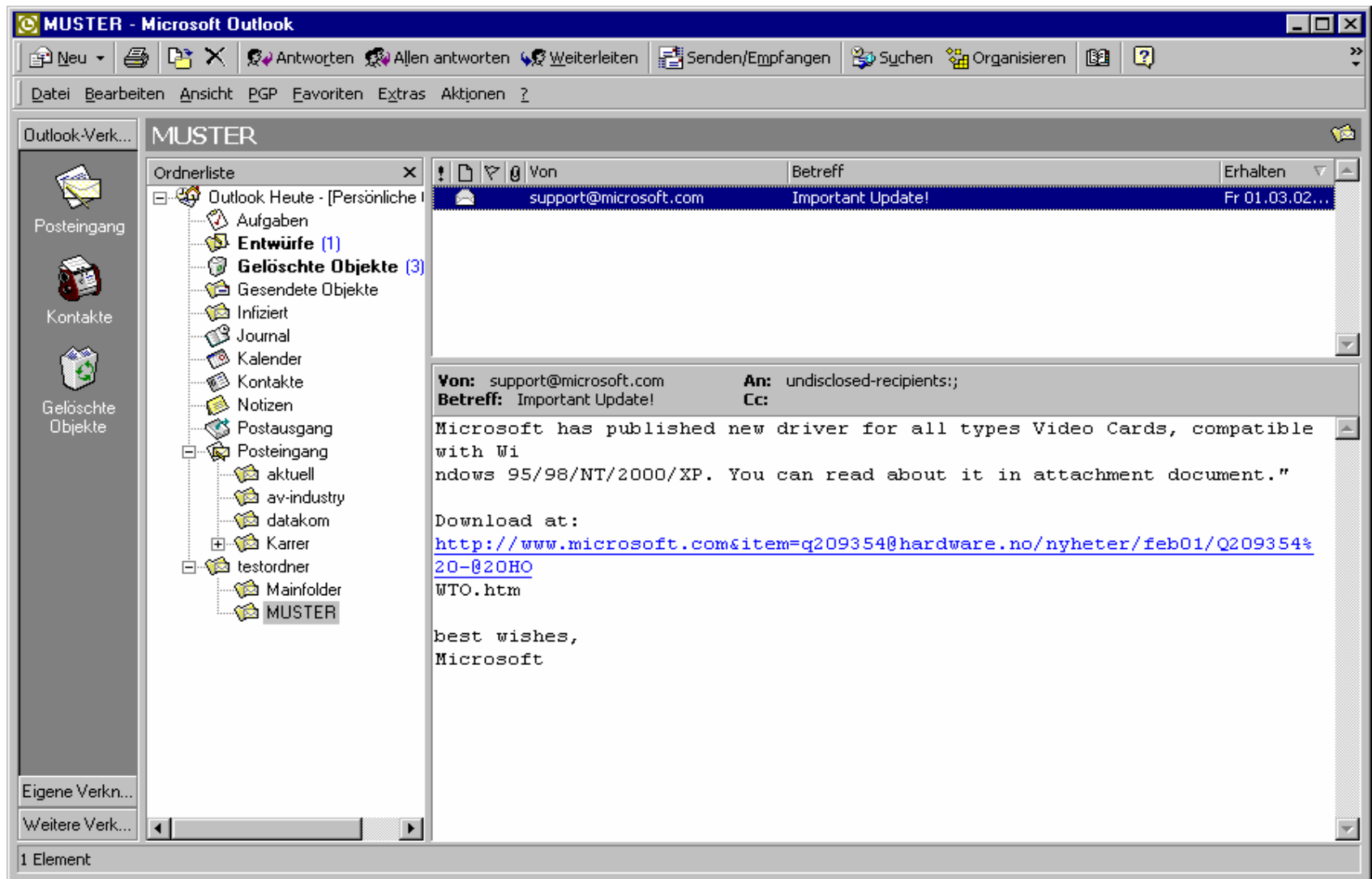
Malware: Direct User Involvement Attack



Malware: Direct User Involvement Attack



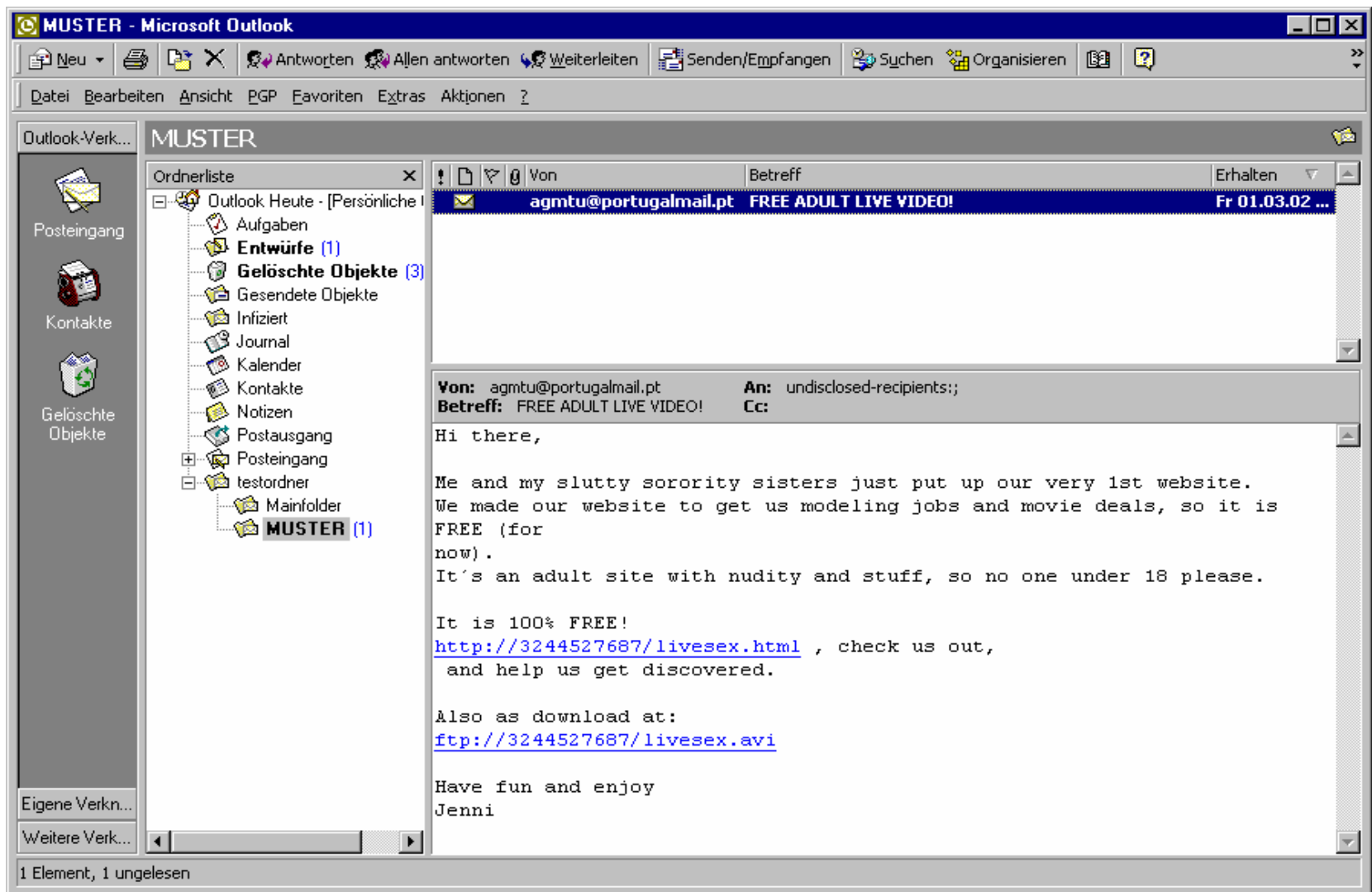
Malware: Indirect User Involvement Attack



Malware: Indirect User Involvement Attack

- URL Redirection Attack
- <http://www.microsoft.com&item=q209354@hardware.no/nyheter/feb01/Q209354%20-@20HOWTO.htm>
- URL redirection using the @ symbol
- URL redirection using DNS spoofing
- URL redirection using DNS cache injection

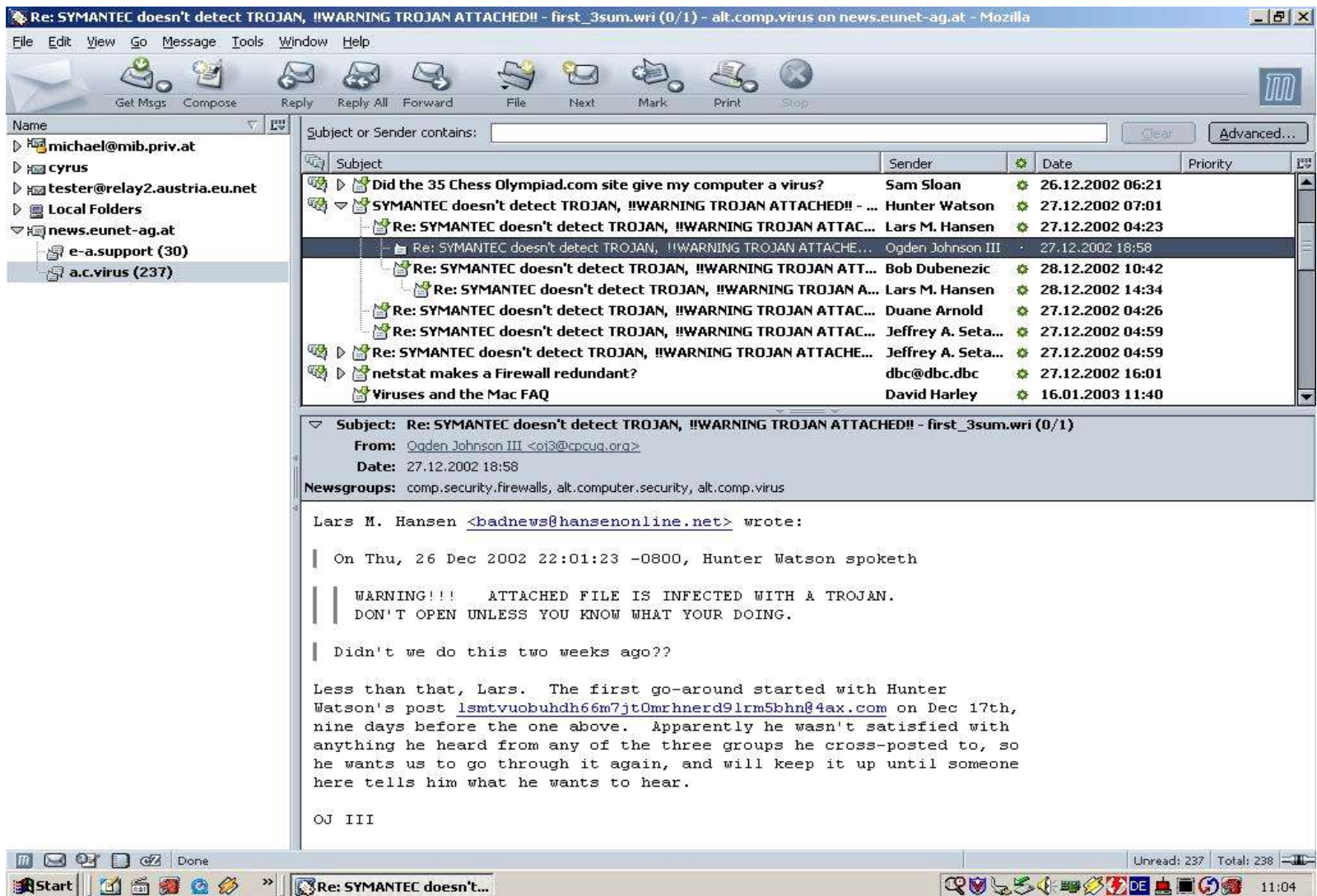
Malware: Indirect User Involvement Attack



Malware: Indirect User Involvement Attack

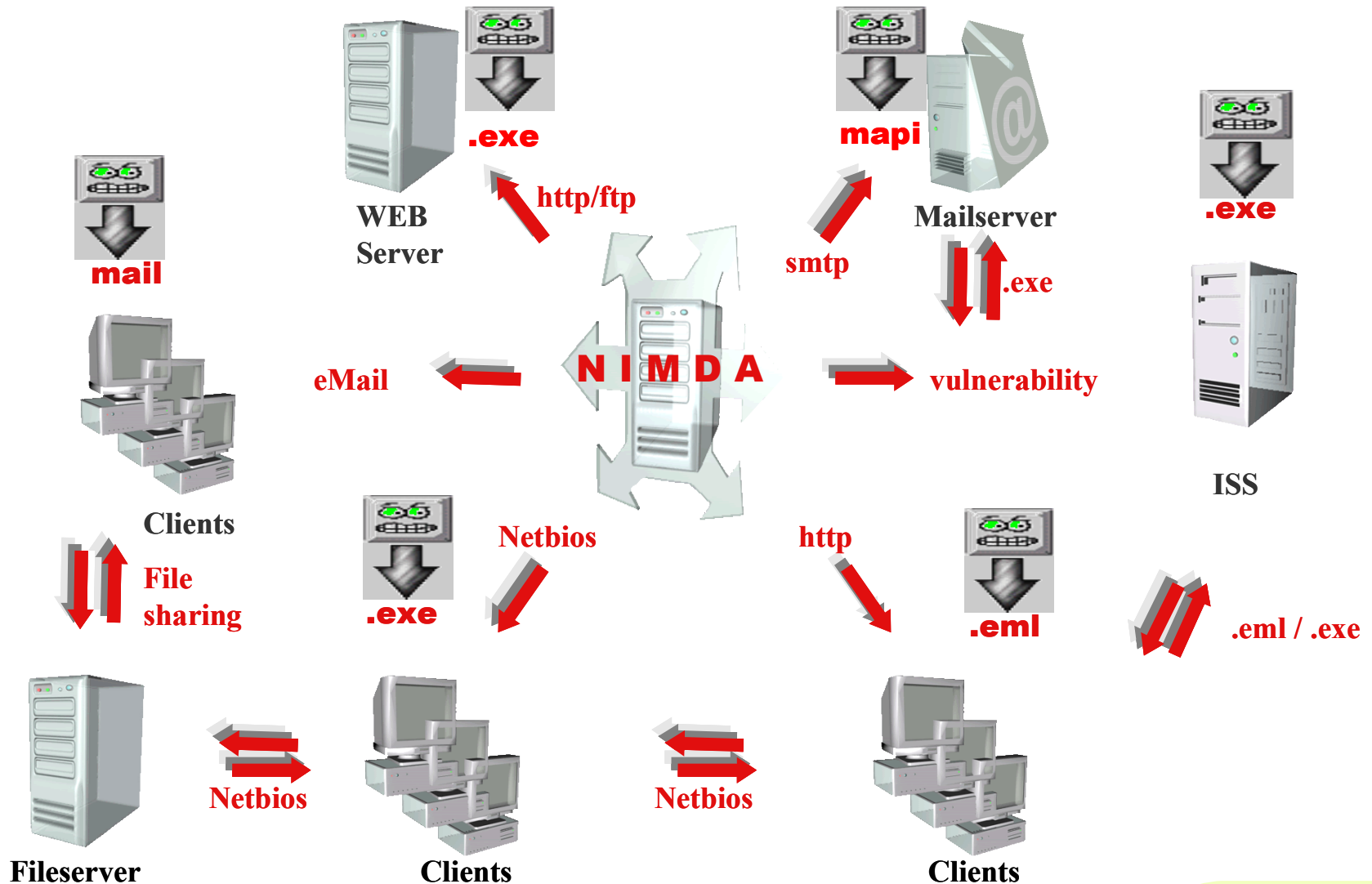
- „Undotted“ IP Address Attack
 - ◆ IE Trusted Sites können verändert werden
- `http://3244527687/livesex.html`
- Eine „punktlose“ IP Adresse verwenden
 - ◆ (z.B. `193.99.144.71 == 3244527687`)
- Diese Adresse überschreibt die „Trusted Site Settings“ von MS Internet Explorer
- -> Lokale Sicherheitsbeschränkungen gelten für die besuchte Seite!

Malware: Newsgroups verbreiten Viren



Malware: NIMDA

ein variantenreicher Mailwurm



Malware: Ablauf der SWEN-Attacke



Malware: Ablauf der SWEN-Attacke - 1

The screenshot shows the Microsoft Outlook interface. The left pane displays the folder structure, with 'Posteingang' (Inbox) selected. The main pane shows a list of emails. The selected email is from 'IKARUS Analyse - Tesar' with the subject 'Current Internet Pack'. The email content includes a Microsoft logo, a link to 'All Products | Support | Search | Microsoft.com Guide', and the text 'Microsoft Partner' followed by a paragraph about a security update.

Von	Betreff	Erhalten
IKARUS Analyse...	Current Internet Pack	Di 30.09.03 09:51
Microsoft	All Products Support Search Microsoft.com Guide	
Microsoft Home		
Microsoft Partner		
Ikarus Webmaster	Network Security Upgrade	Di 30.09.03 09:28
IKARUS Analyse - ...	Swen-Grafik	Di 30.09.03 09:25
Alexander Muffat	Nachtrag zu	Di 30.09.03 09:24
Hofer Robert	WG: Business in Austria...	Di 30.09.03 08:42
Thomas Mandl	WG: Infrastrukturmeeting	Di 30.09.03 08:40
Alexander Muffat	Morgen! Die	Di 30.09.03 08:21
Rajeev Nagar	FW: You tell, we listen. You ask, we deliver. The Windows Driver DevCon agendas here! (IFS Plugfes...	Mo 29.09.03 21:36
Fink, Sonja Judith	was ist mit mm seiner mutter	Mo 29.09.03 20:46
Fink, Sonja Judith	telekom	Mo 29.09.03 20:34
Köller Günter	AW: Aktualisiert: Betätigungsfelder bei der Stadt Wien	Mo 29.09.03 19:56
Köller Günter	Aktualisiert: Betätigungsfelder bei der Stadt Wien	Mo 29.09.03 19:46

Von: IKARUS Analyse - Tesar
Betreff: Current Internet Pack
An: joe pichlmayr
Cc:

Microsoft All Products | Support | Search | Microsoft.com Guide
Microsoft Home

Microsoft Partner

this is the latest version of security update, the "September 2003, Cumulative Patch" update which resolves all known security vulnerabilities affecting MS Internet Explorer, MS Outlook and MS Outlook Express. Install now to help maintain the security of your computer from these vulnerabilities, the most serious of which could allow an attacker to run executable on your computer. This update includes the functionality of all previously released patches.

Malware: Ablauf der SWEN-Attacke - 2

Microsoft

All Products | Support | Search | Microsoft.com Guide

Microsoft Home



MS User

this is the latest version of security update, the "September 2003, Cumulative Patch" update which eliminates all known security vulnerabilities affecting MS Internet Explorer, MS Outlook and MS Outlook Express. Install now to maintain the security of your computer from these vulnerabilities. This update includes the functionality of all previously released patches.

? System requirements	Windows 95/98/Me/2000/NT/XP
? This update applies to	MS Internet Explorer, version 4.01 and later MS Outlook, version 8.00 and later MS Outlook Express, version 4.01 and later
? Recommendation	Customers should install the patch at the earliest opportunity.
? How to install	Run attached file. Choose Yes on displayed dialog box.
? How to use	You don't need to do anything after installing this item.

Microsoft Product Support Services and Knowledge Base articles can be found on the [Microsoft Technical Support](#) web site. For security-related information about Microsoft products, please visit the [Microsoft Security Advisor](#) web site, or [Contact Us](#).

Thank you for using Microsoft products.

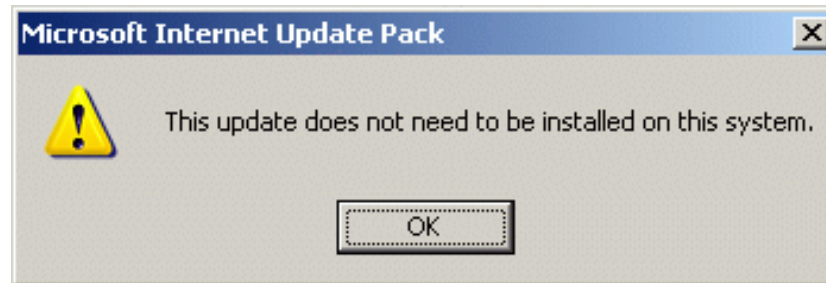
Please do not reply to this message. It was sent from an unmonitored e-mail address and we are unable to respond to any replies.

The names of the actual companies and products mentioned herein are the trademarks of their respective owners.

Contact Us | Legal | TRUSTe

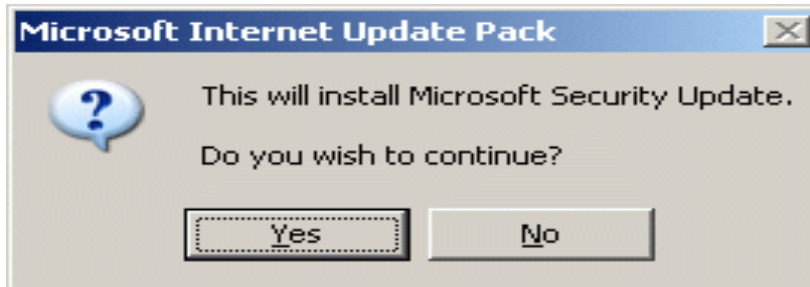
©2003 Microsoft Corporation. All rights reserved. [Terms of Use](#) | [Privacy Statement](#) | [Accessibility](#)

Malware: Ablauf der SWEN-Attacke - 3



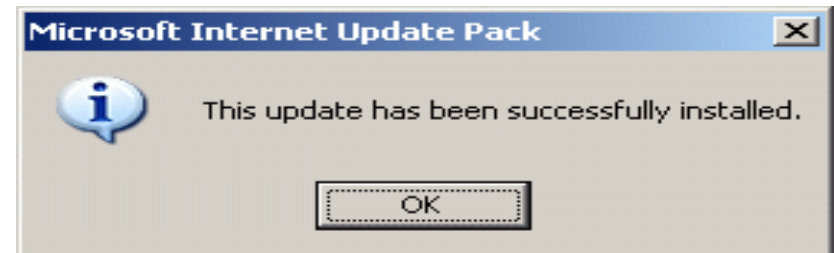
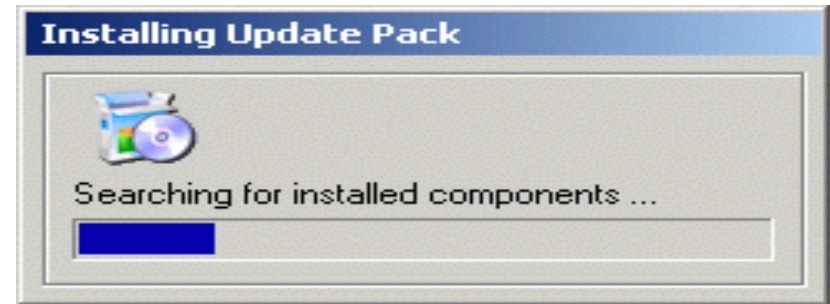
Checkt ob Virus schon auf
befallenem System aktiv war.
Findet der Wurm bereits eine
Kopie
seiner selbst kommt es zu
keiner neuen
Infektion.

Malware: Ablauf der SWEN-Attacke - 3



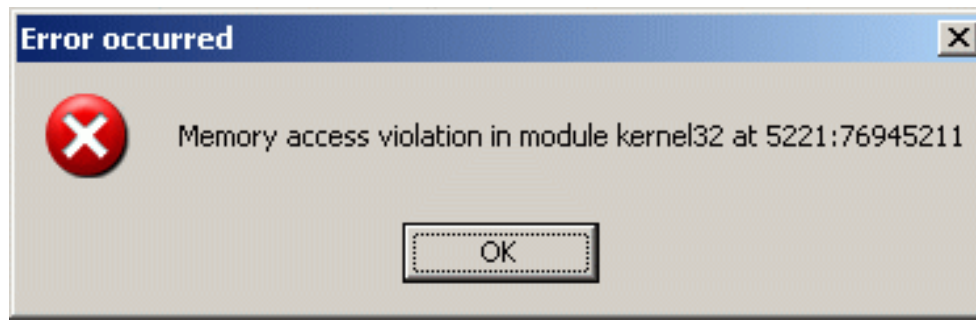
Bestätigen sie mit YES, wird der Virencode „offiziell“ als Security Update installiert.

Klicken Sie NO, wird der Code des Wurms trotzdem im Hintergrund installiert OHNE das der User dies bemerkt.



Malware: Ablauf der SWEN-Attacke - 4

Deaktiviert sofort die laufenden Prozesse (Services) von Online-Tools (WebShield, Guard) und Personal Firewalls und verhindert den Neustart dieser Dienste.



Selbständiges Erfassen der Ziel Adressen aus .html, .asp, .eml, .dbx, .wab, und .mbx Dateien

„Organisatorischer Kram“: Registry Einträge und Modifikation bestehender Values, Droppen von Batch Dateien

Malware: Ablauf der SWEN-Attacke - 5

MAPI32 Exception

An internal error has occurred in module mapi32.dll
Default mail account structure has a damaged table of contents. It is recommended to newly reconfigure your account records. MAPI32 needs these informations in order to be able to send and receive mail. Failure to do so may cause that some MAPI32 dependent applications (such as Outlook or Outlook Express) become non-functional.

In the edit box below, please enter your name as you would like it to appear in the "From" field of your outgoing message.

Your Name:

Please enter your email address. This address will be the address other people use to send email to you.

Email Address: (required)

Please enter the name of your outgoing mail server in the edit box below.

SMTP Server: (required)

Enter the name you will use to log into this account.

Login Name:

Please enter the password for current account.

Password:

Retype password:

Type in the full name of your incoming mail server.

POP3 Server:

Apply Cancel

Faked MAPI32 Exception Error mit dem Ziel an Email-Adresse, Username, Passwort, POP3 Server und SMTP Server Daten zu gelangen. Von dort „verwischt“ er seine Spuren und löscht alle „gesendeten“ eMails aus dem Gesendet-Ordner.

- Ausnützung von programmspezifischen Schutzmöglichkeiten
- Schaffung eines Sicherheitsbewußtseins bei den Anwendern
- Klare Richtlinien durch die IT-Verantwortlichen (Virenschutzpolicies)
- Einsatz von Software unter virenabwehrtechnischen Gesichtspunkten
- Entwicklung von Virenschutzpolicies, Warnsystemen und Notfallplänen

Malware: Technische Gegenmaßnahmen

- Einsatz von *unterschiedlichen* Virenschutzprogrammen auf verschiedenen Ebenen
- Aktivierung von *On-Access-Scannern*
- Realisierung von *white-lists* am Gateway
- Verwenden von aktuellen Sicherheitspatches
- Keine „ungesicherten Modems“ im Netz
- Keine „aktiven“ Inhalte ohne entsprechender Verifizierung

Malware: Funktionsweise von Virensclannern

- Problem Verschlüsselung

Die Bytefolge der Entschlüsselungsroutine ist immer gleich, anhand dieser kann der Virus entdeckt werden

- Problem Polymorphismus

Der Code des Virus kann durch zufällige Änderungen (z.B. Einfügen von NOPs oder ADDs) ein anderes Aussehen erlangen

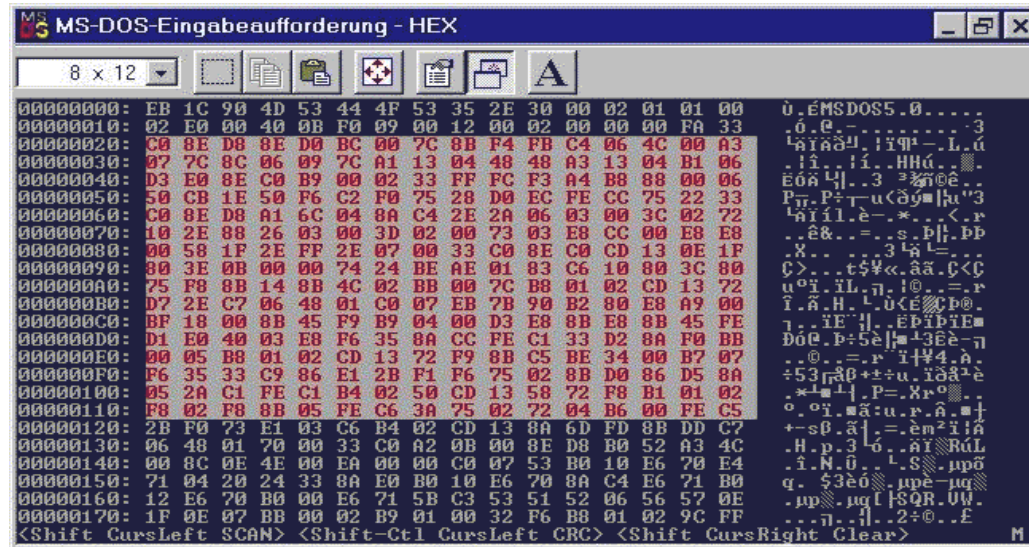
- Problem Verstecken des Virus (*Stealth*)

manipuliert das OS so, daß der Scanner nicht erkennen kann, ob die Datei infiziert ist oder nicht (Virus stellt Original zur Verfügung und kopiert sich nach Scanvorgang wieder in Datei; Virus hat einen Prozess gestartet bzw. interrupt hook gesetzt).

- Gegenmittel

- ◆ Speicherüberwachung ermöglicht das Finden dieses Prozesses

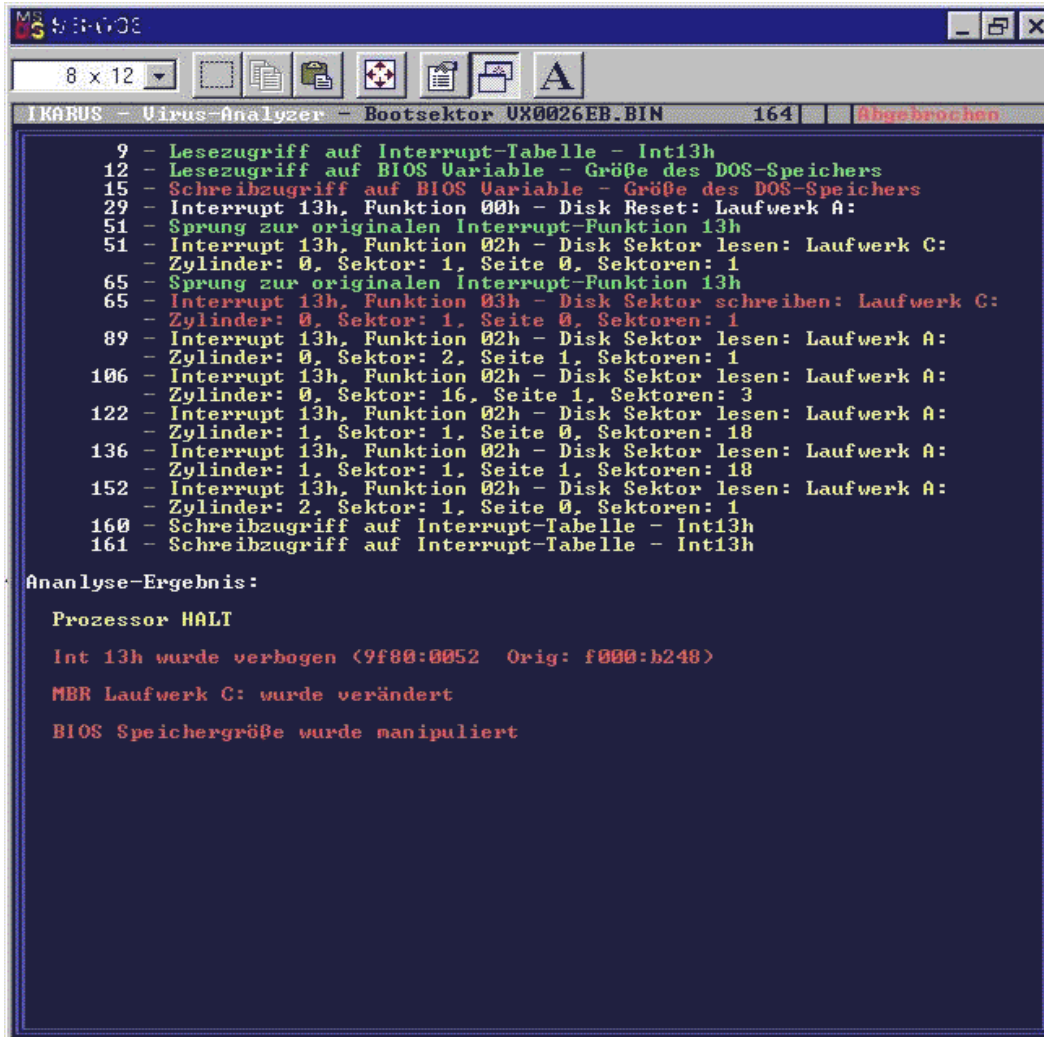
Malware: Funktionsweise von Virensclannern



```
MS-DOS-Eingabeaufforderung - HEX
8 x 12
00000000: EB 1C 90 4D 53 44 4F 53 35 2E 30 00 02 01 01 00  ù.ÉMSDOS5.0...
00000010: 02 E0 00 40 0B F0 09 00 12 00 02 00 00 00 FA 33  .ó.0-...:..3
00000020: C0 8E D8 8E D0 BC 00 7C 8B F4 FB C4 06 4C 00 A3  ÿiãðµ.ïÿñ-.L.ú
00000030: 07 7C 8C 06 09 7C A1 13 04 48 48 A3 13 04 B1 06  .ï.ï.ï.HHú.ï.
00000040: D3 E0 8E C0 B9 00 02 33 FF FC F3 A4 B8 88 00 06  ÉóãL|.3³%ñ0é..
00000050: 50 CB 1E 50 F6 C2 F0 75 28 D0 EC FE CC 75 22 33  P.ÿ.P÷ÿ-ÿ-u<ðÿ=|u"3
00000060: C0 8E D8 A1 6C 04 8A C4 2E 2A 06 03 00 3C 02 72  ÿiïl.è-.*....<.r
00000070: 10 2E 88 26 03 00 3D 02 00 73 03 E8 CC 00 E8 E8  ..â&..=-.s.p||.bb
00000080: 00 58 1F 2E FF 2E 07 00 33 C0 8E C0 CD 13 0E 1F  .x...3¹qL=-.
00000090: 80 3E 0B 00 00 74 24 BE AE 01 83 C6 10 80 3C 80  C>..t$%<.âã.C<C
000000A0: 75 F8 0B 14 8B 4C 02 BB 00 7C B8 01 02 CD 13 72  uoÿ.ïL.ñ.ï0.-.r
000000B0: D7 2E C7 06 48 01 C0 07 EB 7B 90 B2 80 E8 A9 00  î.ã.H.L.ù<ÉÿB0.
000000C0: BF 18 00 8B 45 F9 B9 04 00 D3 E8 8B E8 8B 45 FE  ÿ..ïÉ.ï|.ÉBïPïE=
000000D0: D1 E0 40 03 E8 F6 35 8A CC FE C1 33 D2 8A F0 BB  Dó0.p÷5è;|=+3Eè-ñ
000000E0: 00 05 B8 01 02 CD 13 72 F9 8B C5 BE 34 00 B7 07  ..0..=r.ï|ÿ4.ã.
000000F0: F6 35 33 C9 86 E1 2B F1 P6 75 02 8B D0 86 D5 8A  ÷53ÿãß++÷u.ÿðã²è
00000100: 05 2A C1 FE C1 B4 02 50 CD 13 58 72 F8 B1 01 02  .*+|=ÿ.P=-.x.r0.
00000110: F8 02 F8 8B 05 FE C6 3A 75 02 72 04 B6 00 FE C5  .o.ÿ.ã:u.r.ã.#+
00000120: 2B F0 73 E1 03 C6 B4 02 CD 13 8A 6D FD 8B DD C7  +-sß.ã|ÿ.=èm²ïã
00000130: 06 48 01 70 00 33 C0 A2 0B 00 8E D8 B0 52 A3 4C  .H.p.3|ÿ.ãÿRúL
00000140: 00 8C 0E 4E 00 EA 00 00 C0 07 53 B0 10 E6 70 E4  .î.N.0.ÿ.L.S.ÿpö
00000150: 71 04 20 24 33 8A E0 B0 10 E6 70 8A C4 E6 71 B0  q. $3è0.ÿ.µpe-µq
00000160: 12 E6 70 B0 00 E6 71 5B C3 53 51 52 06 56 57 0E  -µp.ÿ.µq|ÿSQR.UW.
00000170: 1F 0E 07 BB 00 02 B9 01 00 32 F6 B8 01 02 9C FF  ..-ñ.ï|.2÷0..É
<Shift CursLeft SCAN> <Shift-Ctrl CursLeft CRC> <Shift CursRight Clear> M
```

- Checksum / Signatur (Scan + Schutztechnik)
- Scan
 - In der zu prüfenden Datei wird eine Checksumme gebildet. Wird diese gefunden, dann handelt es sich um das Virus.
- Schutz
 - Summe über die gesamte Datei; verändert sich diese; wurde die Datei manipuliert.

Malware: Funktionsweise von Virensclannern



```
MS-DOS
8 x 12
IKARUS - Virus-Analyzer - Bootsektor UX0026EB.BIN 164 Abgebrochen

 9 - Lesezugriff auf Interrupt-Tabelle - Int13h
12 - Lesezugriff auf BIOS Variable - Größe des DOS-Speichers
15 - Schreibzugriff auf BIOS Variable - Größe des DOS-Speichers
29 - Interrupt 13h, Funktion 00h - Disk Reset: Laufwerk A:
51 - Sprung zur originalen Interrupt-Funktion 13h
51 - Interrupt 13h, Funktion 02h - Disk Sektor lesen: Laufwerk C:
   - Zylinder: 0, Sektor: 1, Seite 0, Sektoren: 1
65 - Sprung zur originalen Interrupt-Funktion 13h
65 - Interrupt 13h, Funktion 03h - Disk Sektor schreiben: Laufwerk C:
   - Zylinder: 0, Sektor: 1, Seite 0, Sektoren: 1
89 - Interrupt 13h, Funktion 02h - Disk Sektor lesen: Laufwerk A:
   - Zylinder: 0, Sektor: 2, Seite 1, Sektoren: 1
106 - Interrupt 13h, Funktion 02h - Disk Sektor lesen: Laufwerk A:
     - Zylinder: 0, Sektor: 16, Seite 1, Sektoren: 3
122 - Interrupt 13h, Funktion 02h - Disk Sektor lesen: Laufwerk A:
     - Zylinder: 1, Sektor: 1, Seite 0, Sektoren: 18
136 - Interrupt 13h, Funktion 02h - Disk Sektor lesen: Laufwerk A:
     - Zylinder: 1, Sektor: 1, Seite 1, Sektoren: 18
152 - Interrupt 13h, Funktion 02h - Disk Sektor lesen: Laufwerk A:
     - Zylinder: 2, Sektor: 1, Seite 0, Sektoren: 1
160 - Schreibzugriff auf Interrupt-Tabelle - Int13h
161 - Schreibzugriff auf Interrupt-Tabelle - Int13h

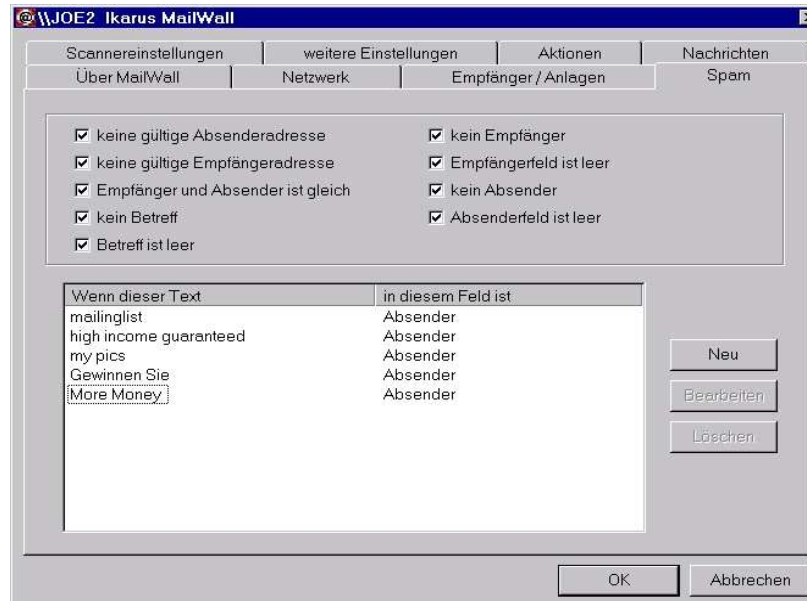
Analyse-Ergebnis:
  Prozessor HALT
  Int 13h wurde verbogen (9f80:0052 Orig: f000:b248)
  MBR Laufwerk C: wurde verändert
  BIOS Speichergröße wurde manipuliert
```

• Heuristic

- ◆ Die überprüfte Datei wird in einer „virtuellen“ Betriebsumgebung gestartet
- ◆ Die Funktion der Datei wird in einem eigenen Simulator vom Scanner interpretiert
- ◆ Beurteilung erfolgt nach virentypischem Verhalten

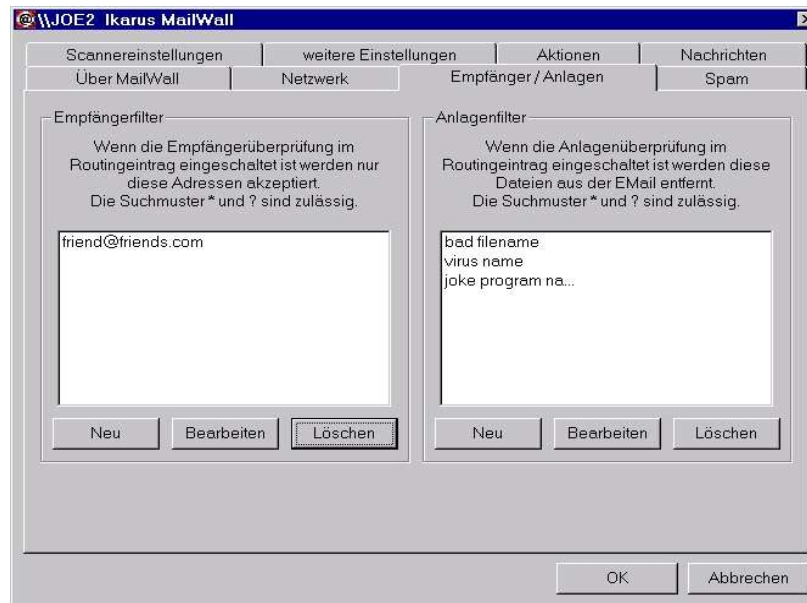
Malware: Scanner: Black-List Ansatz

- ◆ Vordefinierte Dateien (-Typen) oder Inhalte **dürfen nicht** ausgeführt oder verbreitet werden
- ◆ Erfordert genaue Analyse möglicher Bedrohungsszenarien der für den Arbeitsprozess eingesetzten Systeme, Dateien und Mitarbeiter
- ◆ Unterschiedliche Gruppensegmente ermöglichen einen optimal angepassten Virenschutz
- ◆ Weniger restriktiv, ermöglicht jedoch ein flexibleres Risikomanagement, allerdings extrem wartungsintensiv



Malware: Scanner: White-List Ansatz

- ◆ *Nur vordefinierte* Dateien (-Typen) oder Inhalte **dürfen** ausgeführt oder verbreitet werden
- ◆ Erfordert genaue Analyse der für den Arbeitsprozess erforderlichen Dateien und Inhalte
- ◆ Unterschiedliche Gruppensegmente ermöglichen einen optimal angepassten Virenschutz
- ◆ Restriktiv aber ausgesprochen wirkungsvoll und kostenschonend, da kaum wartungsanfällig



Malware: Grenzen der AV-Techniken

RetroViren

In der Biologie erhält ein RetroVirus RNS (Ribonukleinsäure);
Genetische Materie desVirus wird in die DNS eingefügt

„Tödlich“ - kein Virus

// some kill machine function
// -> delete or change necessary files

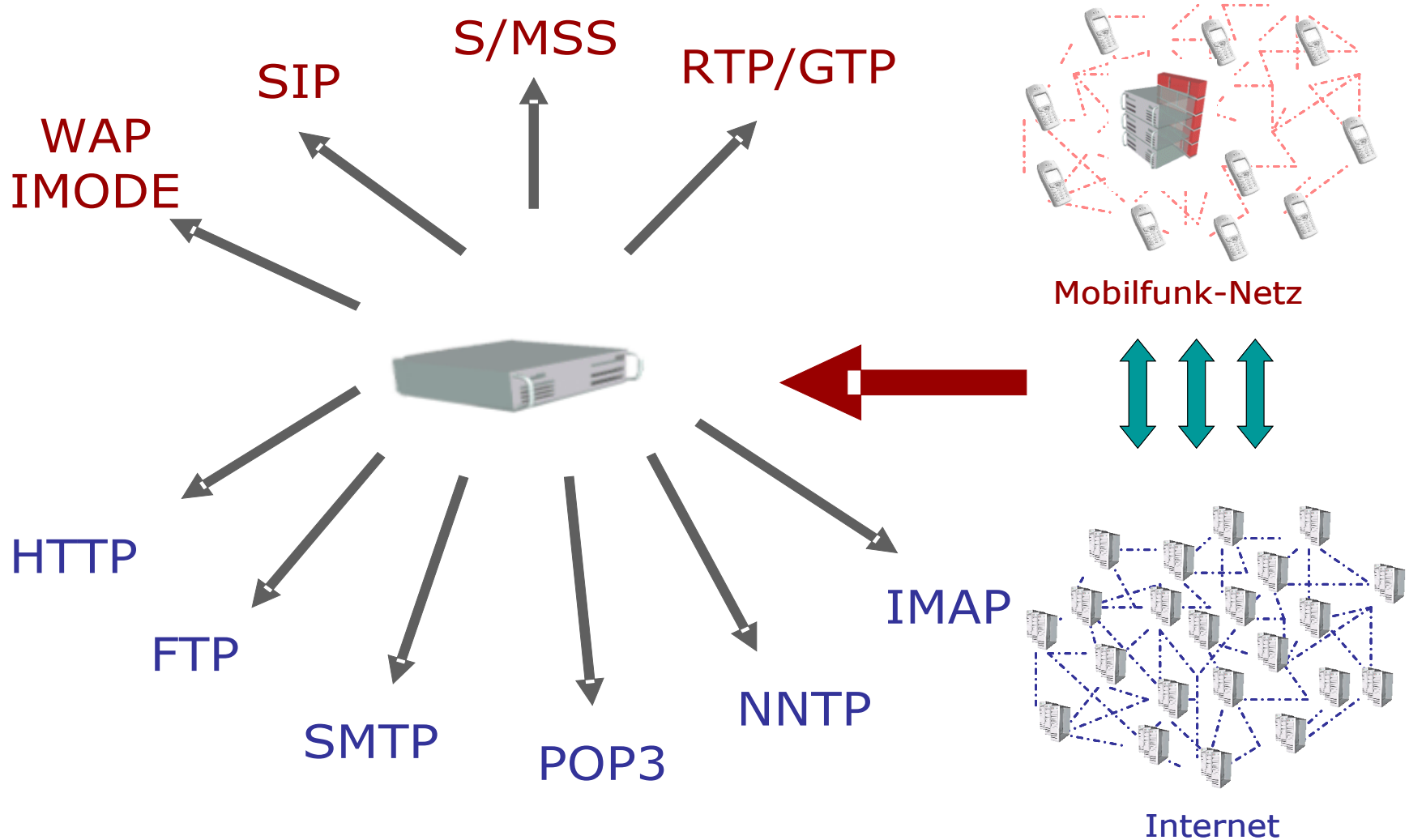
Hochpolymorph

Verschlüsselte Dateien (Backdoors) sind bereits jetzt ein
großes Problem werden noch eskalieren, wie die letzten Viren
zeigten. Auch die Viren-Tools (Generatoren) haben bereits
Verschlüsselungen eingebaut

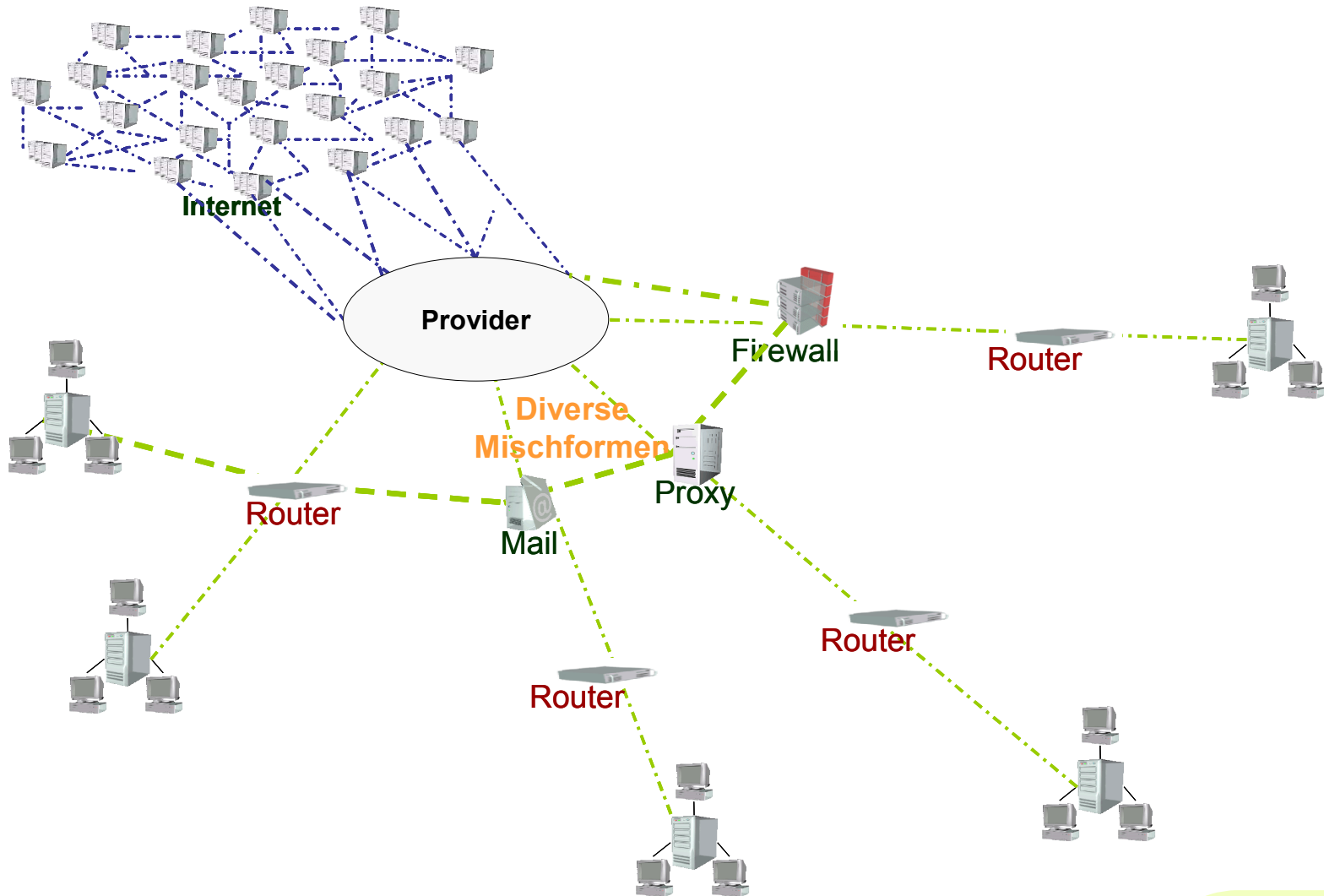
„Longraider - Attacken“

Angriff erfolgt über ein partielles Übertragen des
viralen/malicious Codes in vielen kleinen Teilen über sehr
lange Zeiträume

Malware: Gefährdung mobiler Systeme



Malware: Standorte der Virens Scanner



Malware: Was uns erwartet

- Anzahl und Variationen steigen
 - ◆ besonders infolge der veröffentlichten Sourcen z.B. von Phatbot
- HLL-Viren, die sich via WEB (http) verbreiten
 - ◆ Chat (IRC/ICQ/...) Viren
- Steigende Netzwerkfunktionalität (Verknüpfung Wurm/Trojaner)
- Entwicklung neuer Angriffstrategien v.a. mit Würmern
 - ◆ DDOS
 - ◆ Versenden von SPAM
- Angriffe gegen Mobile Endgeräte (PDA, Handy)
 - ◆ Kann sehr teuer werden, wenn 0900-Nummern gewählt werden
- Angriffe gegen Service Provider (ASP's)
- Verstärktes Auftreten von Backdoor-Programmen

Angriffe: Denial of Service (DoS)

- DoS versucht, einen Service zu blockieren
 - ◆ Kann großen Schaden anrichten, wenn Firmen von Webservices leben (z.B. Amazon)
 - ◆ Betrifft aber auch das Attackieren (Lahmlegen) lokaler Services
- Lokale DoS Angriffe
 - ◆ Beeinträchtigung der Prozesse
 - ◆ Überlastung und „Hängenbleiben“ des Prozessors
 - ◆ Aufzehren der Festplattenkapazität
 - ◆ Belegung der Indexknoten in UNIX
 - ◆ Ausführung unzulässigen Codes mit Absturz des Rechners
- DoS Angriffe über das Netzwerk
 - ◆ Von einem Rechner aus
 - ◆ Verteilte DoS-Angriffe: Distributed DoS (DDoS)

Angriffe: Informationslecks

- Ein System, das Informationen “verliert”, ist inhärent unsicherer als ein abgeschottetes
 - ◆ Die Hersteller von Systemen veröffentlichen oft eine Liste bekannter Fehler
 - ◆ Die bekannten Fehler treten meist in einem bestimmten Bereich von Versionsnummern des Produkts auf
- Ein Angreifer versucht, die Fehler mit Hilfe der Informationen über das System und dessen Version über angepaßte Exploits auszunutzen
 - ◆ Es existieren Programme (z.B. nmap), die über das Systemverhalten (Timing, Fehlercodes, ...) versuchen, das Betriebssystem und die Version zu erraten

- *Normaler* Dateizugriff
 - ◆ Erschleichen des Zugriffs auf Standarddateien
 - ◆ Ausnutzen der eigenen Dateizugriffsberechtigungen, um Dateien anderer Benutzer zu ändern (SymLink-Angriff)
 - ◆ Einspielen falscher System-/Treibersoftware
- *Spezieller* Dateizugriff
 - ◆ Spezialdateien (von Applikationen) bzw. Datenbanksysteme bieten erweiterte Möglichkeiten des Angriffs
 - ◆ über die Betriebssystemfunktionalität
 - ◆ über die Anwendung (Logins, Datenverarbeitung, ...)

Angriffe: Remoteausführung von Code

- Wird typischerweise mit eigenen Tools ermöglicht
 - ◆ Der Angreifer untersucht automatisiert die Schwachstellen des Systems
 - ◆ Und nutzt diese (und weitere Tools), um sich administrativen Zugriff auf das System zu verschaffen
- Üblicherweise erhält der Angreifer die Rechte des angegriffenen Programms
 - ◆ Administratoren und Systemdesigner sollten versuchen, die benötigten Rechte für ihre (Server-) Software möglichst gering zu halten

Angriffe: Privilegien ausbauen

- Oft gelingt nur ein schwach privilegierter Zugriff auf ein System (wie ein gewöhnlicher User oder der http-Daemon)
 - ◆ Dies bedeutet nicht, daß die Sicherheit des Systems gewährleistet ist
 - ◆ Bei einem hartnäckigen Angriff ist es nur noch eine Frage der Zeit, bis die Rechte erweitert sind
 - ◆ Es kann nun vom System selbst aus angegriffen werden
 - ◆ Hiermit stehen die Chancen höher, an Informationen über das System und damit dessen Sicherheitslücken zu gelangen

Angriffe: Pufferüberläufe

- Beim Aufruf einer Funktion werden auf den Stack verschiedene Werte geschrieben (abhängig von System und Compiler sowie dessen Optimierung)
 - ◆ Rücksprungadresse
 - ◆ lokale Variablen und Parameter
 - ◆ Adresse des Stackframes für die Funktion
- Geschicktes Überschreiben von Puffern kann das System manipulieren
 - ◆ meist im Stack
 - ◆ Heap ist als Angriffspunkt auch möglich
- Teilweise Abhilfe schaffen *Canary Values (Lockvögel)*
 - ◆ Schreiben eines Zufallswertes, der bei der Rückkehr aus der Funktion geprüft wird; hiermit kann festgestellt werden, ob der Wert überschrieben wurde

Angriffe: Pufferüberläufe

- Der Angriff funktioniert meist über unsichere Funktionen, die aufgerufen werden
 - ◆ `char *strcpy(char *dest, const char *src)`
 - ◆ `char *strcat(char *dest, const char *src)`
 - ◆ `char *gets(char *buffer)`
 - ◆ und viele andere
- Ist die Programmiersprache C der/die Böse?
 - ◆ Einerseits ja: In C/C++ kann beliebiger hardwarenaher Code implementiert werden
 - ◆ Andererseits nein: Das Betriebssystem müsste sich vor Angriffen schützen

Angriffe: Format-String-Schwachstellen

- Sehr ähnlich zu Pufferüberläufen, aber
 - ◆ es handelt sich mehr um Eingabefunktionen
 - Diese werden dann z.B. als Ausgabefunktion mißbraucht
 - ◆ der Angreifer weiß, wo die eingeschleusten Codes/Daten im Speicher stehen werden
- Die `printf()`-Funktionen sind
 - ◆ `printf()`, `fprintf()`, `sprintf()`, `snprintf()`
 - ◆ und andere
- Eine Codezeile wie `printf(argv[1]);` ist sehr gefährlich, da der Angreifer Formatanweisungen einschleusen kann und sich so (nicht für ihn bestimmte) Daten anzeigen lassen kann
 - ◆ Dies eignet sich z.B. für einen *Dump* des Speichers, um nach dem privaten Schlüssel zu suchen

Angriffe: Datenvernichtung

- Der Angreifer kann Zugang zu gelöschten Datenträgern bekommen
 - ◆ z.B. über den Müll
 - ◆ Dateien, die gelöscht werden, befinden sich noch auf den Datenträgern; meist werden nur die Einträge im Inhaltsverzeichnis gelöscht und der verwendete Speicherplatz als frei markiert
 - ◆ Ein sicheres Löschen heißt *Wipe*
- Hardware wird durch Daten geändert
 - ◆ Sogar die Zellen von RAM-Bausteinen ändern sich, wenn lange Zeit dieselben Daten gespeichert werden
 - ◆ Bausteine mit Schlüsseln müssen vernichtet werden

Angriffe: Datenvernichtung

- CDs müssen (mindestens) zerbrochen werden
 - ◆ Zerkratzen reicht nicht wegen Codierung mit Mehrfachfehlerkorrektur (Löcher bis ca. 2 mm können „repariert“ werden)
 - ◆ Zerbrechen verzieht die Scheiben und macht ein genaues Zusammensetzen äußerst schwierig
 - ◆ Gilt für ROMs. Auf wiederbeschreibbaren Medien kann ein Wipe durchgeführt werden. Bei langer Datenhaltung empfiehlt sich dennoch die Zerstörung.
- Eigene Firmen beschäftigen sich mit der Zerstörung von Daten
 - ◆ Der sicherste Weg, wenn die Firma vertrauenswürdig ist

Sicherheitszertifizierung: BSI

- www.bsi.de
 - ◆ Bundesamt für Sicherheit in der Informationstechnik, Deutschland
- IT-Grundschutz-Handbuch zum Herunterladen
 - ◆ Praxisnah
 - ◆ Viele Informationen zum Thema der Sicherheitsinfrastruktur
- Mehrstufiges Zertifizierungsprogramm für Institutionen und Unternehmen
 - ◆ Mehrstufig mit zeitlich beschränkter Selbsterklärung und
 - ◆ Extern durchgeführtem Audit
- Grundschutz-Tool
 - ◆ Hilft bei der Erstellung von eigenen Sicherheitskonzepten

Kryp|to-
gramm *das*; -s, -e: 1. ein Text, aus dessen Worten sich durch einige besonders gekennzeichnete Buchstaben eine neue Angabe entnehmen läßt (z. B. eine Jahreszahl, eine Nachricht); vgl. Chronogramm. 2. (veraltet) Geheimtext. **Kryp|to|graph** *der*; -en, -en: (veraltet) Gerät zur Herstellung von Geheimschriften (für den telegrafischen Verkehr). **Kryp|to|graphie** *die*; -, ...ien: 1. absichtslos entstandene Kritzelzeichnung bei Erwachsenen (Psychol.). 2. (veraltet) Geheimschrift.

aus: DUDEN, Fremdwörterbuch

Einteilung

- Kryptographie
 - ◆ Wissenschaft von den Methoden der Ver- und Entschlüsselung von Daten
- Kryptanalyse
 - ◆ Wissenschaft von der Wiederherstellung des Klartextes ohne Zugriff auf den Schlüssel

Kryptologie: Beispiel Geheimschrift

A	B	C
D	E	F
G	H	I

J	
K	L
M	

N.	Ö	.P
Q.	.R	.S
T.	Ü	.V

W	
X.	.Y
Z	

Zum Beispiel ist

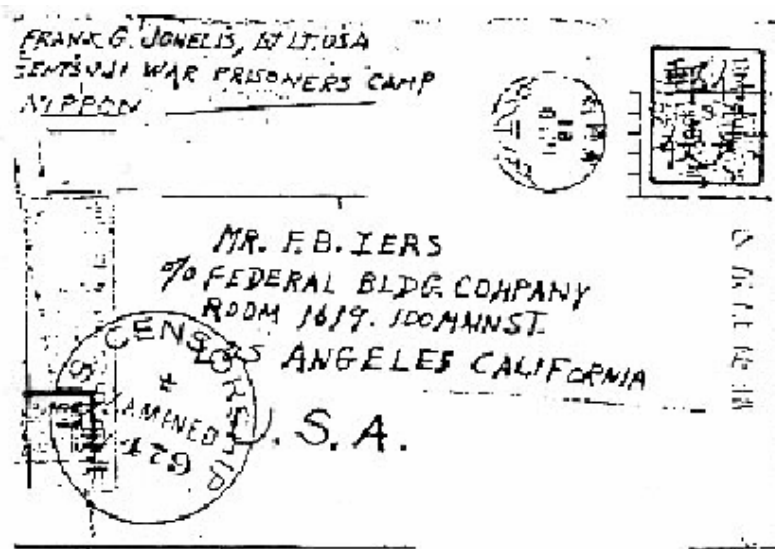
> ◻ < ◻ ◻ ◻ ◻ ◻ ◻ ◻ ◻ ◻ ◻ ◻ ◻

nichts anderes als das Wort KRYPTOGRAPHIE.

- Kein Schlüssel wird verwendet
-> Einfache Transposition

Kryptologie: Stenographie

- Verbergen der Existenz von Informationen
- Linguistische Steganographie



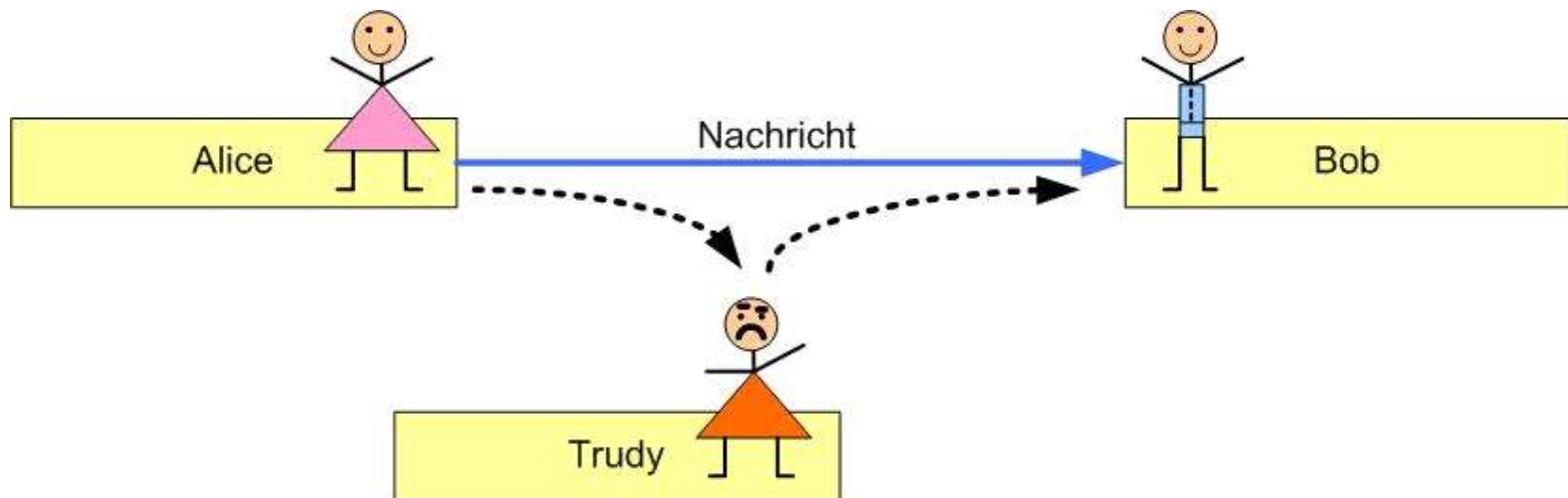
An das FBI:

**AFTER SURRENDER FIFTY PERCENT AMERICANS
LOST; IN NIPPON 30%**

- Technische Steganographie
 - ◆ Geheimtinte
 - ◆ Microdot
 - ◆ Mikrofilm
 - ◆ Verstecken in einem Bild (Bitmap)
 - Manipulation der Farb- und Helligkeitswerte einzelner Pixel
 - z.B. Änderung des Least Significant Bit eines Pixels
 - Nur geeignete Pixel verändern
 - R(ot) G(rün) B(lau) -> 3*8 Bit pro Pixel
 - ◆ Verstecken in anderen Daten
- Es sind viele freie Tools zur Steganographie als Open Source verfügbar
 - ◆ Unter anderem unter www.cotse.com

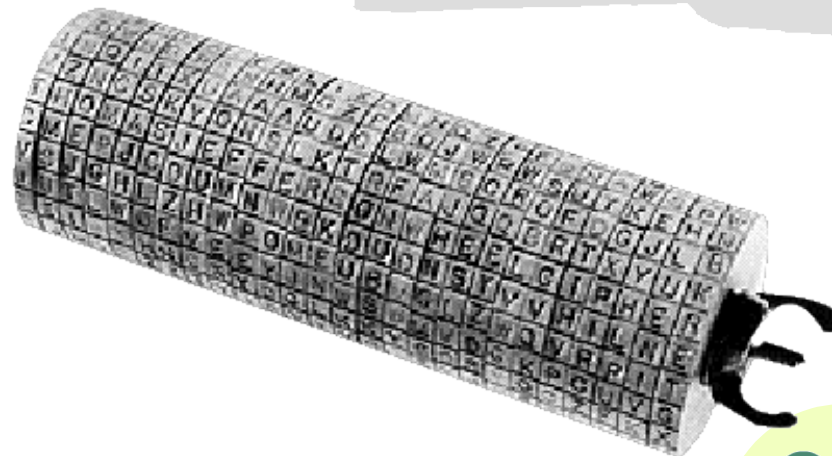
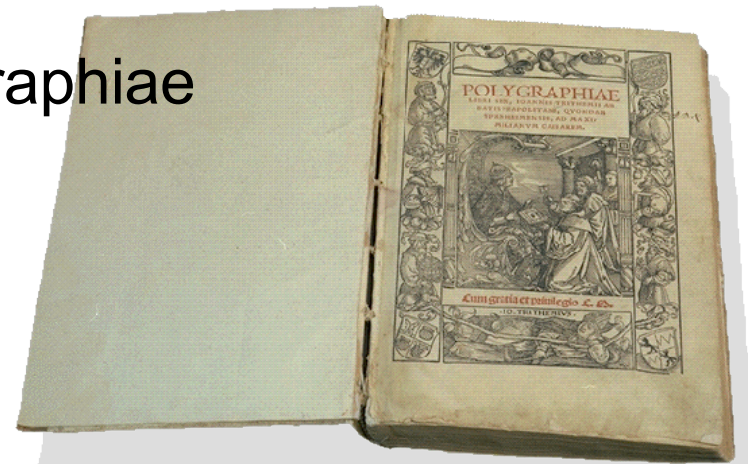
Kryptologie: Verschlüsselung

- Alice und Bob wollen vertraulich miteinander kommunizieren
- Trudy will die Nachrichten abhören/verändern
- Alice und Bob verschlüsseln die Nachricht damit Trudy nichts versteht



Kryptologie: Historische Entwicklung

- Caesar-Code: Ersetzungsalgorithmus mit Positionsdifferenz als Schlüssel
- Johannes Trithemius, 1518: Polygraphiae
- Jefferson Zylinder, 18. Jh.
- Enigma, 2. WK, Deutschland



Kryptologie: Auguste Kerckhoff (1835-1903)

- Unterschied zwischen taktischen (kurzzeitiger Schutz) und strategischen (langzeitiger Schutz) Zielen der Geheimhaltung
- Die Sicherheit eines Systems sollte nur vom Schlüssel abhängen („Kerckhoff-Prinzip“)
 - ◆ Eine Art „Open Source Denken“ wurde gefordert
 - ◆ 6 Regeln (5 + Prinzip)
 - Die verschlüsselte Nachricht sollte praktisch unknackbar sein
 - Sender und Empfänger dürfen keinen Schaden erleiden, wenn das System geknackt wurde
 - Der Schlüssel muß leicht auswendig zu lernen und veränderbar sein
 - Das System muß einfach zu benutzen sein und sollte keine übermäßigen Anstrengungen verlangen
 - Das Chiffriersystem sollte von Experten gut untersucht sein

Kryptologie: Symmetrische Algorithmen

- Gleicher Schlüssel für Ver- und Entschlüsselung
 - ◆ Bzw. kann ein Schlüssel je aus dem anderen berechnet werden
- $C = \text{enc}(K, P)$; $P = \text{dec}(K, C)$
- Schwierigkeit
 - ◆ Sender und Empfänger müssen den Schlüssel erst austauschen



Kryptographie: Asymmetrische Algorithmen (auch *Public-Key-Cryptosystems*)

- Ver- und Entschlüsselung erfolgen mit verschiedenen Schlüsseln
 - ◆ Jeder Teilnehmer hat einen geheimen Schlüssel (private key) zum Entschlüsseln
 - ◆ Mit einem öffentlichen Schlüssel (public key) können die Sender die Nachricht kodieren; dieser Schlüssel kann anderen mitgeteilt werden, da von ihm nicht (einfach) auf den geheimen Schlüssel geschlossen werden kann
- $C = \text{enc}(K_{\text{pub}}, P)$; $P = \text{dec}(K_{\text{priv}}, C)$

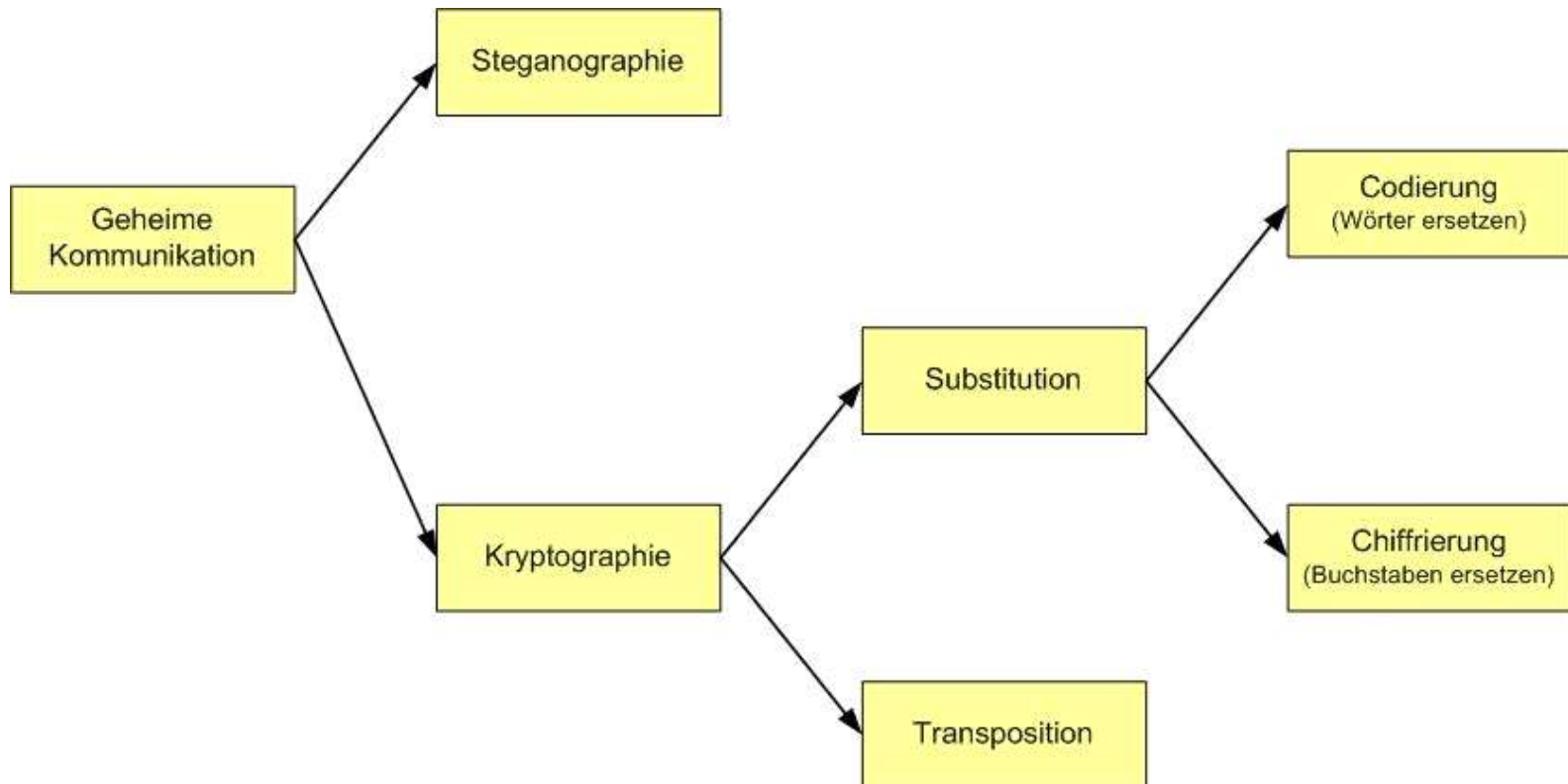


Kryptographie: Asymmetrische Algorithmen

- Die Signatur einer Nachricht erfolgt umgekehrt zur Verschlüsselung:
 - ◆ Verschlüssen der Signatur mit dem privaten Schlüssel und Überprüfen der Signatur mit dem zugehörigen öffentlichen Schlüssel
- Die Verfahren beruhen auf mathematischen Falltürfunktionen (*Trapdoor Functions*)
 - ◆ In eine Richtung leicht zu berechnen (z.B. Multiplikation) und in die andere Richtung schwierig (z.B. Faktorisierung, also Zerlegung in Primfaktoren)

- Stromchiffrierung
 - ◆ Verschlüsselung einzelner Bits (oder Bytes) eines beliebig langen Datenstroms
 - ◆ Bytes können ständig hinzugefügt werden
- Blockchiffrierung
 - ◆ Es werden immer ganze Datenblöcke bearbeitet
 - ◆ Die Blöcke müssen immer mit Daten angefüllt (gepadding) sein
 - ◆ Heute maschinell eingesetzte Verfahren gehören meist in diese Kategorie

Kryptologie: Chiffrierungsarten



- ◆ Klartext (*Plaintext P*)
- ◆ Schlüsseltext, Chiffre (*Ciphertext C*)
- ◆ Schlüssel (*key K*)
- ◆ $C = \text{enc}(K, P)$; $P = \text{dec}(K, C)$ (*encrypt, decrypt*)
- ◆ $S = \text{sign}(K_{\text{priv}}, D)$; $\text{verify}(K_{\text{pub}}, S, D)$
- ◆ *Kompromittieren*
 - Brechen eines Systems oder Systemteils (z.B. Schlüssel)
- ◆ *Rauschfreiheit*
 - Von Daten und Schlüssel unabhängige Ausführung (Laufzeit, Stromverbrauch, ...)
- ◆ Unterscheidung der Sicherheit in
 - Theoretische Sicherheit: Selbst mit unbeschränkten Ressourcen ist das System nicht zu knacken
 - Praktische Sicherheit: Das System ist knackbar, aber die Ressourcen dafür existieren (noch) nicht

Kryptologie: XOR-Operation

- Wird als Basis der maschinellen Kryptographie verwendet

$$0 \oplus 0 = 0$$

$$1 \oplus 0 = 1$$

$$0 \oplus 1 = 1$$

$$1 \oplus 1 = 0$$

$$\text{UND: } a \oplus a = 0, a \oplus b \oplus b = a$$

- Werden Daten mit einem Schlüssel über XOR verknüpft, hängt das Chiffre „zu gleichen Teilen“ von den Daten und vom Schlüssel ab

Kryptologie: One-Time Pads

- Der Schlüssel besteht aus einer echten Zufallszahlenfolge
- Beim Verschlüsseln wird jedes Schlüsselbytes nur für ein Datenbyte verwendet
- Die Vermeidung von Perioden und die Zufälligkeit des Schlüssels macht das Verfahren perfekt (es ist nicht brechbar)!
- Botschaften tauschen sehr lange Schlüsselsätze aus
- In der Praxis aufgrund des Schlüsselaustausches nur für sehr teure Spezialanwendungen geeignet

- Data Encryption Algorithm (DEA) und Data Encryption Standard (DES)
 - ◆ 1977 als US-Norm FIPS 46 publiziert
 - ◆ bis heute wurden nicht alle Entwicklungskriterien bekannt gegeben
- Konfusion und Diffusion
 - ◆ K.: Die Statistik des Schlüsseltextes soll die Statistik des Klartextes so beeinflussen, daß kein Rückschluß möglich ist
 - ◆ D.: Jedes Bit des Klartextes und des Schlüssels sollen möglichst viele Bits des Schlüsseltextes beeinflussen
- DES ist der meistverwendete symmetrische Algorithmus

Kryptologie: DES

- 64 Bit Blöcke werden mit 56 Bit Schlüsseln verschlüsselt
- Schlüsselraum 256 (8 Byte inkl. 8 Parity-Bits) klein
- 10.000 parallele Recheneinheiten können den Schlüssel in

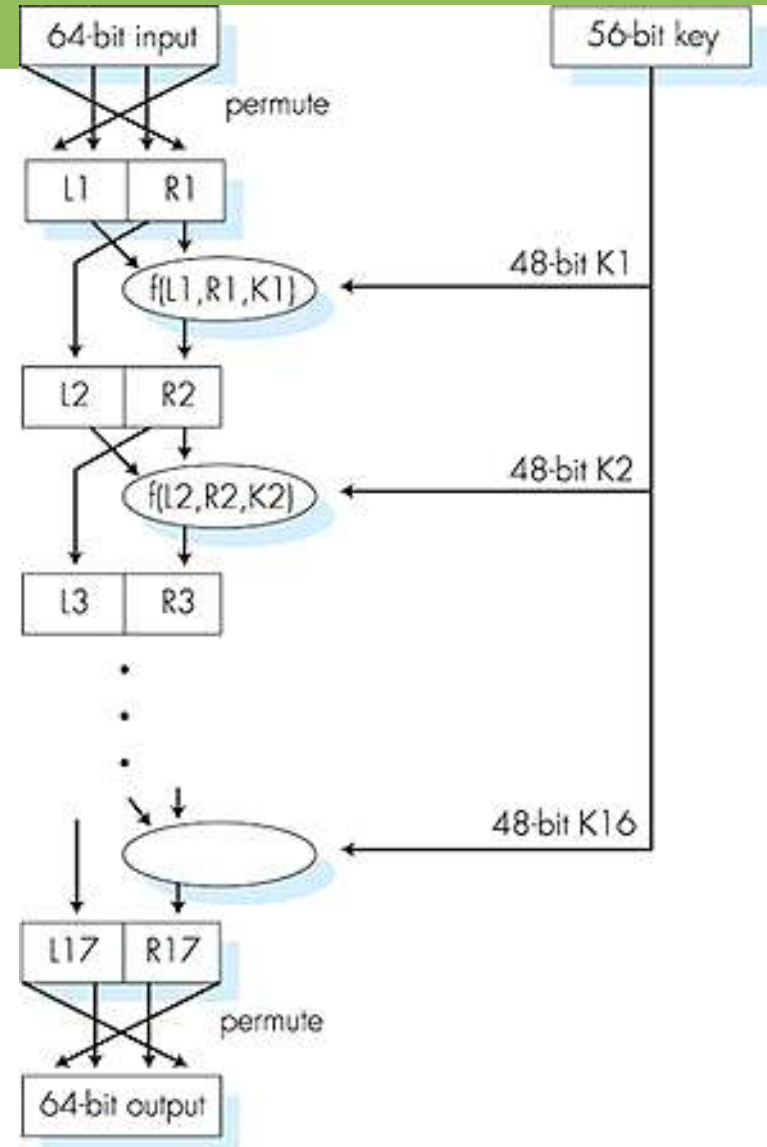
$$\frac{2^{56} \cdot 64n \text{ sec}}{10000} \cdot \frac{1}{2} \approx 64h$$

- Geschwindigkeit des DES:

Realisierung	Geschwindigkeit
Chipkarte 4,9MHz, Coproz.	80µs/8Byte=800kBit/s
PC (Pentium, 200MHz)	4µs/8Byte=16MBit/s
DES Hardwarebaustein	64ns/8Byte=100MBit/s

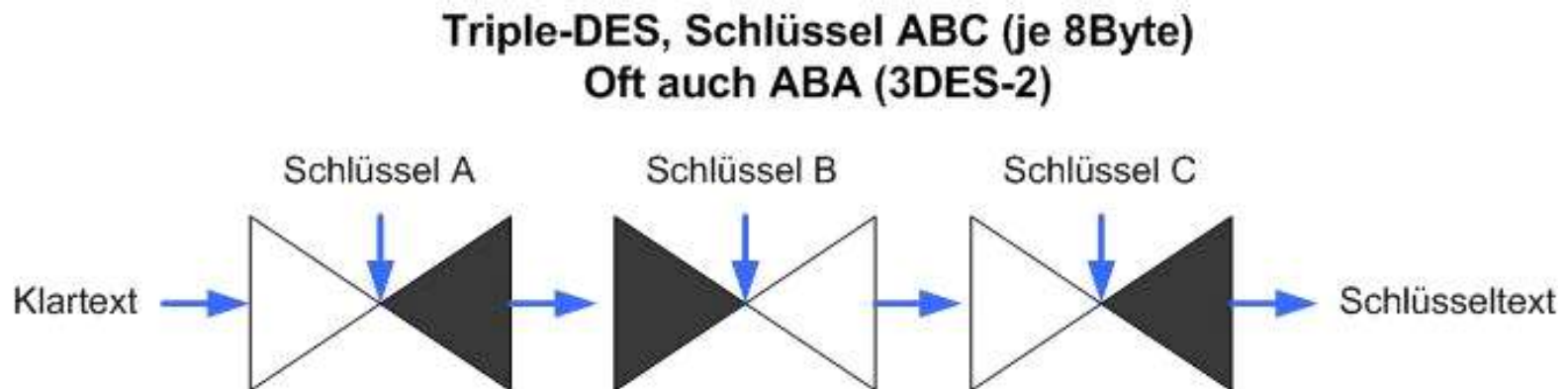
Kryptologie: Struktur von DES

- Initiale Permutation
- 16 identische “Runden” von Einzelverschlüsselungen, jede mit einem 48 Bit langen abgeleiteten Schlüssel
- Finale Permutation



Kryptologie: Triple-DES

- Erweiterung des Schlüsselraumes des „alten“ DES
- Schlüsselraum erweitert sich nicht im gleichen Maß wie der Berechnungsaufwand
- 3DES-2 wird heute hauptsächlich in der Bankenwelt zur symmetrischen Verschlüsselung verwendet



- DES ist aufgrund des beschränkten Schlüsselraums nur auf absehbare Zeit sicher
- 2000 wurde aus den Einsendungen eines Wettbewerbs der Algorithmus Rijndael ausgewählt
 - ◆ Blöcke sind 128 Bit, Schlüssel sind 128, 192 oder 256 Bit lang
 - ◆ Gute Realisierung in Hardware von 8Bit bis 64 Bit-Prozessoren
 - ◆ Lizenzfrei, 20 Jahre Lebensdauer
 - ◆ Sehr klare und einfache Struktur
 - ◆ Seit der Veröffentlichung bereits Schwächungen im Schlüsselraum

- Freier Algorithmus von Bruce Schneier
 - ◆ Schnell
 - auf 32 Bit Prozessoren (26 CPU-Zyklen pro Byte)
 - ◆ Kompakt
 - < 5 KB Speicher
 - ◆ Einfach
 - Nur einfache Operationen (Addition, XOR und Tabellenindizierung) werden verwendet
 - Der Entwurf lässt sich leicht analysieren und damit gegen Implementierungsfehler absichern
 - ◆ Variable Sicherheit
 - Schlüssellänge variabel bis 448 Bit bei 128 Bit Daten
 - ◆ Aber
 - Nicht als kryptographische Hashfunktion einsetzbar
 - Nicht für Chipkarten geeignet (hoher RAM-Bedarf)

- Sehr einfache Mathematik, beruht auf dem Problem der Faktorisierung von Zahlen
- $c = p^{K_{pub}} \bmod n$
- $p = c^{K_{priv}} \bmod n$
- $n = p * q$ (geheime Primzahlen)

- Einsatz heute mit 1024 Bit langen Schlüsseln
 - ◆ Typische öffentliche Schlüssel sind: 2, 3, 17, 65537
 - ◆ Hoher Speicher- und Performancebedarf
 - ◆ Berechnung im 200MHz-Pentium benötigt 6ms zum Verschlüsseln, 60ms zum Entschlüsseln

RSA: Beispiel (1/2)

Schlüsselgenerierung

- **1.** Suche zwei Primzahlen p und q
 $p=3, q=11$
- **2.** Berechne öffentlichen Modulus
 $n = p * q = 33$
- **3.** Berechne Hilfsvariable zur Schlüsselerzeugung
 $z = (p-1) * (q-1) = 20$
- **4.** Berechne öffentlichen Schlüssel e mit ($e < z$) und ($\text{ggT}(z, e) = 1$); wähle eine der in Frage kommenden Zahlen aus
 $[1, 3, 7, 9, 11, 13, 17]$ $e = 7$
- **5.** Berechne geheimen Schlüssel d mit ($d * e \text{ mod } z = 1$)
 $d = 3$

RSA: Beispiel (2/2)

- Verschlüsselung

- ◆ Klartext p (mit $p < n$) sei 4

$$p = 4$$

- ◆ Verschlüsselung

$$c = 4^7 \bmod 33 = 16384 \bmod 33 = 16$$

- Entschlüsselung

- ◆ Schlüsseltext c (mit $c < n$) sei 16

$$c = 16$$

- ◆ Entschlüsselung

$$p = 16^3 \bmod 33 = 4096 \bmod 33 = 4$$

Kryptologie: Elliptische Kurven

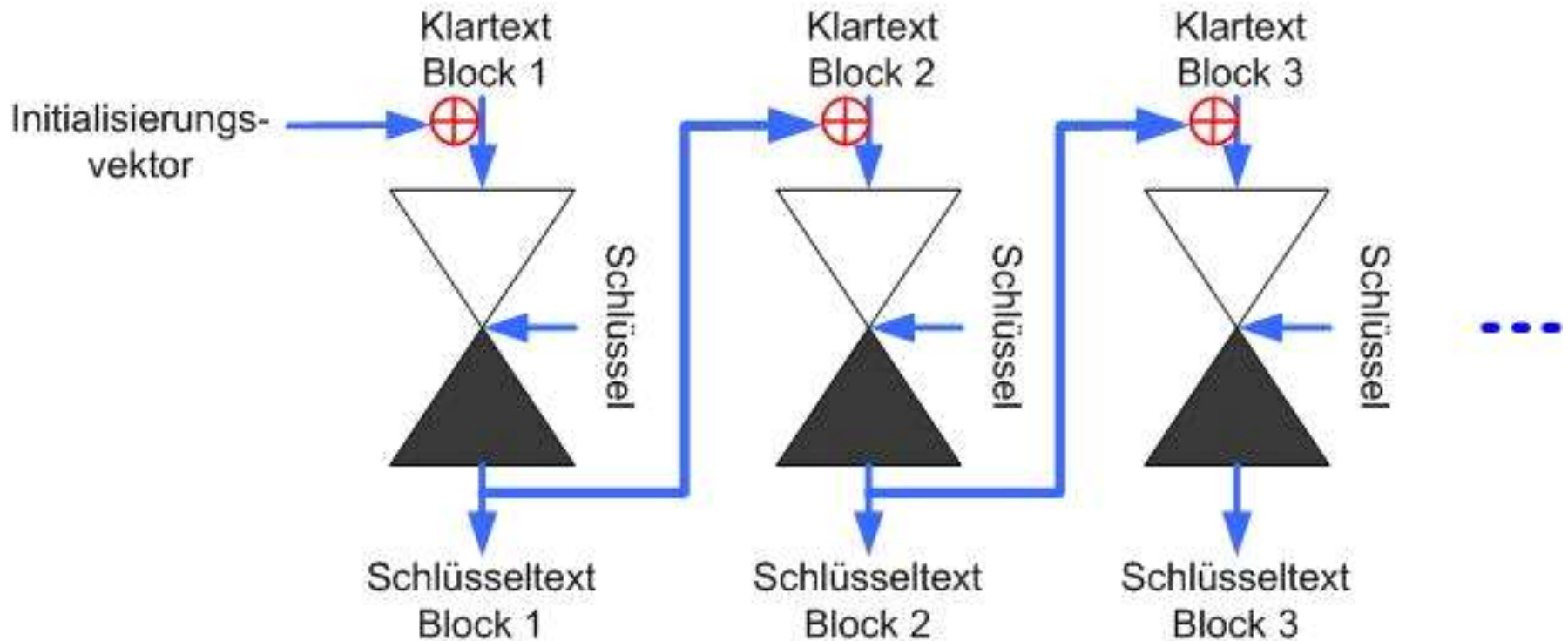
- ECC, Elliptic Curve Cryptosystems, 1985
- Elliptische Kurven sind zusammenhängende ebene Kurven mit der Gleichung

$$y^2 = x^3 + ax + b \text{ in einem endlichen Körper}$$

- Die kompliziertere Mathematik bietet eine bessere Falltürfunktion als die ausgereifte Faktorisierung
 - ◆ 1024 Bit Schlüssellänge bei RSA entsprechen 160 Bit bei elliptischen Kurven
- In der Praxis
 - ◆ nicht im Bankenfeld verwendet, da das Vertrauen in die neue Kryptographietechnik noch nicht aufgebaut ist

Kryptologie: Blockalgorithmen, Betriebsart CBC

CBC-Modus (Cipher Block Chaining)



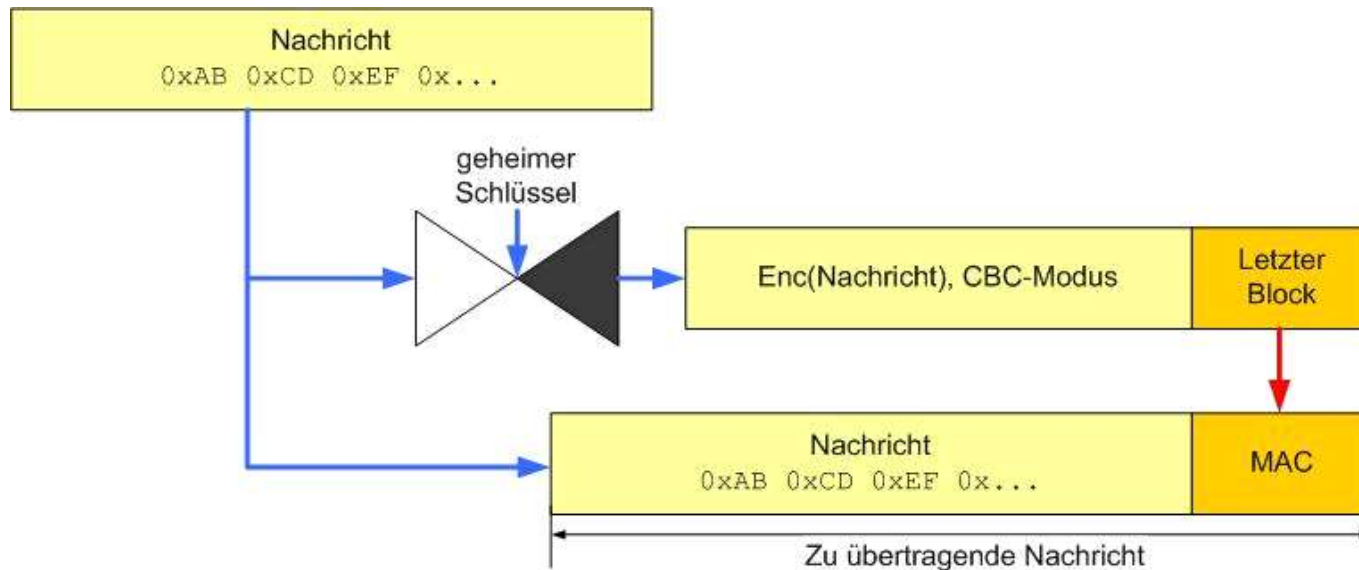
- Padding

- ◆ Blockorientierte Algorithmen verlangen immer ganze Blöcke
- ◆ Füllen die Daten einen Block nicht, müssen die Blöcke mit einem speziellen Muster gefüllt werden
- ◆ Üblicherweise wird mit `0x80 00 ... 00` aufgefüllt (ISO9797, Methode 2), es gibt auch andere standardisierte Möglichkeiten

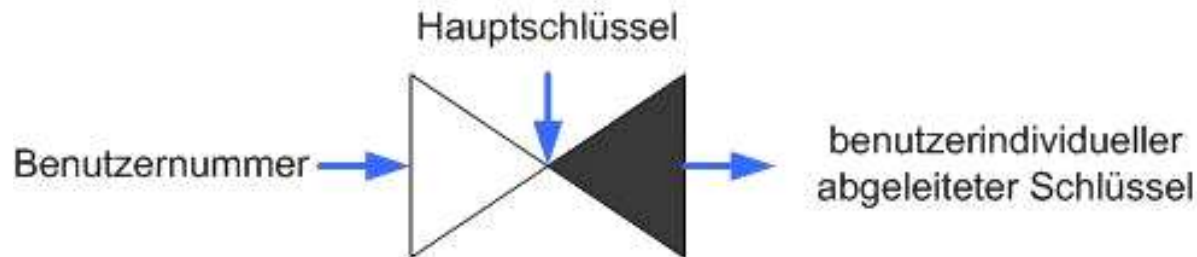


Kryptologie: MAC

- MAC, Message Authentication Code
- Kryptographische Checksumme zur Überprüfung, ob ein Dokument verändert wurde
 - ◆ Eine digitale Signatur ist beruht (meist) auf asymmetrischer Kryptographie
 - ◆ Ein MAC beruht meist auf symmetrischer Kryptographie



- Es soll nicht immer derselbe Schlüssel verwendet werden
 - ◆ Bei einer Kompromittierung des Schlüssels soll nicht das ganze System kompromittiert werden
 - ◆ Das gesamte Schlüsselsystem soll durchschaubar bleiben und nicht für jeden Kunden ein Schlüssel gespeichert werden
- Abgeleitete Schlüssel
 - ◆ Für jeden Kommunikationspartner (z.B. jeden Bankkunden) wird ein eigener Schlüssel aus einem Hauptschlüssel abgeleitet



- Schlüsselversionen

- ◆ In einem festen Zeitraster (z.B. jedes Jahr) wird ein neuer Hauptschlüssel verwendet oder über das Datum abgeleitet

- Dynamische Schlüssel

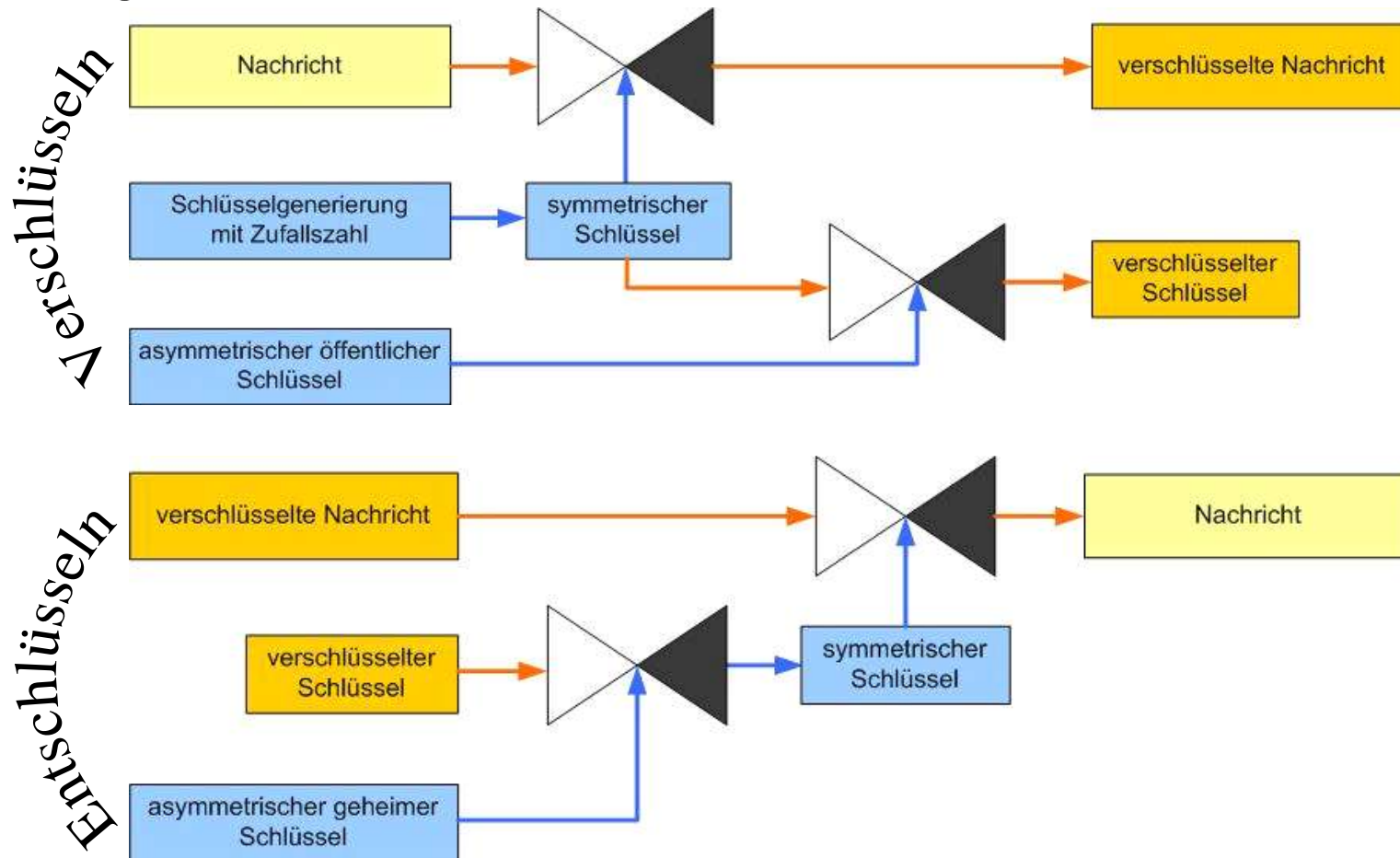
- ◆ Schlüssel werden eigens für eine Sitzung oder Transaktion abgeleitet (Session Keys, temporäre Schlüssel)
- ◆ Ableitung z.B. über Zufallszahl, die unter den Partnern ausgetauscht wird
- ◆ ANSI X9.17:

$$\text{Key}_{i+1} = \text{enc}(\text{Key}_{\text{Gen}}, \text{enc}(\text{Key}_{\text{Gen}}, (T_i \parallel \text{Key}_i)))$$

ist auch nicht rückrechenbar

Kryptologie: Schlüsselmanagement

- Schlüssel können auch über asymmetrische Kryptographie ausgetauscht werden

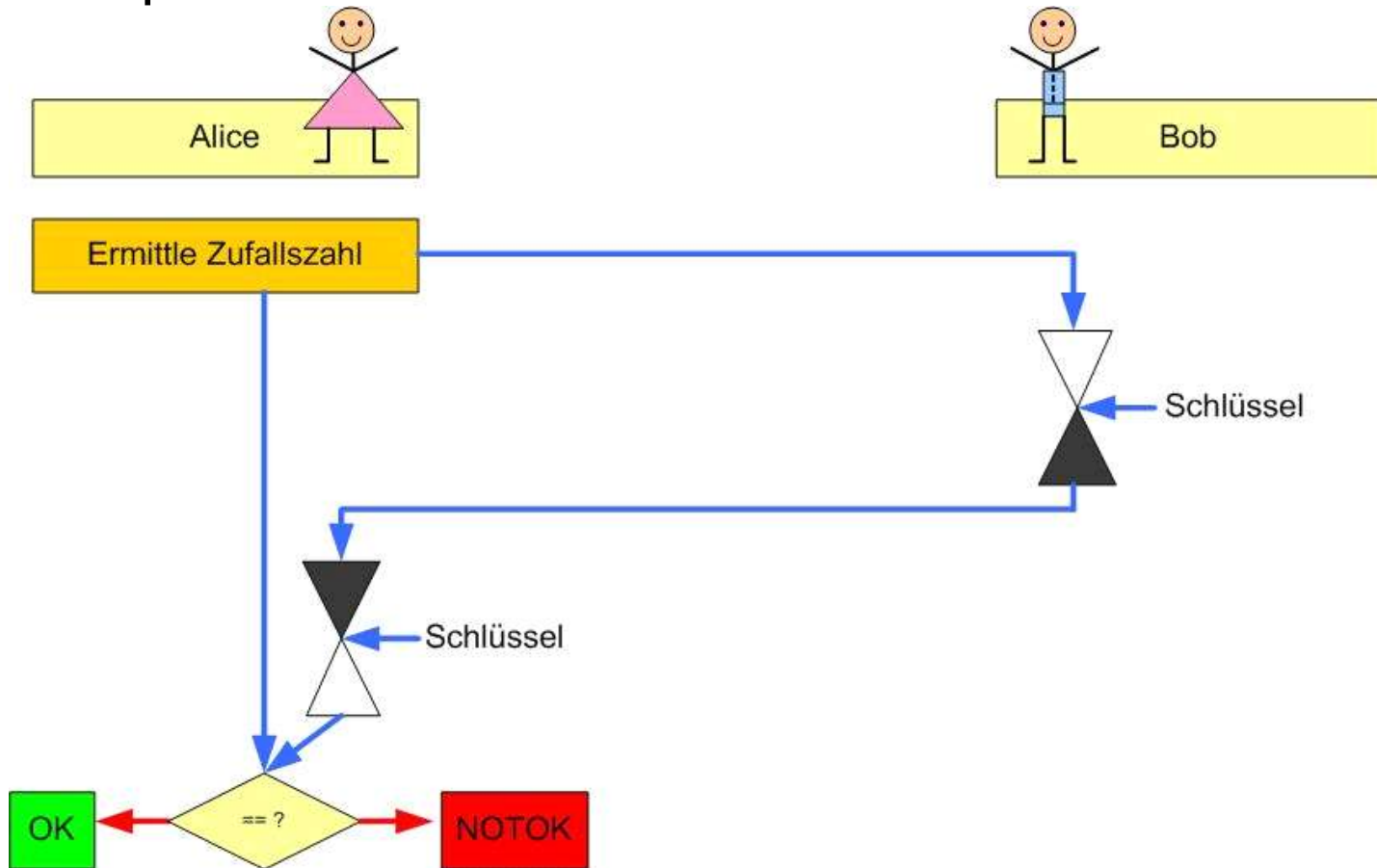


Kryptologie: Authentifizierung

- Auch Authentisierung, Authentifikation
 - ◆ (nicht aber Autorisierung)
- Dient zur Sicherstellung der Echtheit des Kommunikationspartners
 - ◆ Bob soll Alice beweisen, daß er Bob ist
- Grobe Unterscheidung
 - ◆ *einseitig* (nur Bob beweist seine Authentizität, z.B. in GSM der Nutzer gegenüber dem Operator) oder *gegenseitig* (Alice und Bob beweisen ihre Authentizität, z.B. Bankomat)
 - ◆ *symmetrisch* oder *asymmetrisch* aufgrund des verwendeten Algorithmus
- Oft wird eine Authentisierung über das *Challenge-Response-Verfahren* durchgeführt
 - ◆ Challenge ist meist eine Zufallszahl

Kryptologie: Authentifizierung

- Einseitige symmetrische Authentisierung über Challenge-Response



- Gegenseitige Authentifizierung funktioniert so:
 - ◆ Alice und Bob wollen sich gegenseitig authentifizieren
 - ◆ Alice schickt Bob eine Zufallszahl $Rand_A$
 - ◆ Bob verschlüsselt erzeugt eine eigene Zufallszahl $Rand_B$, berechnet und sendet folgende Daten
 - ◆ $enc(\text{Schlüssel}, Rand_A || Rand_B)$
 - $||$ bedeutet das Zusammenhängen von Daten
 - ◆ Alice entschlüsselt die Daten, prüft die Zufallszahl $Rand_A$, berechnet und sendet folgende Daten
 - ◆ $enc(\text{Schlüssel}, Rand_B || Rand_A)$
 - ◆ Bob entschlüsselt nun die Daten und überprüft, ob Alice die Daten korrekt verschlüsseln konnte
- Auf diese Weise werden weniger Nachrichten ausgetauscht als wenn zwei einseitige Authentifizierungen durchgeführt würden

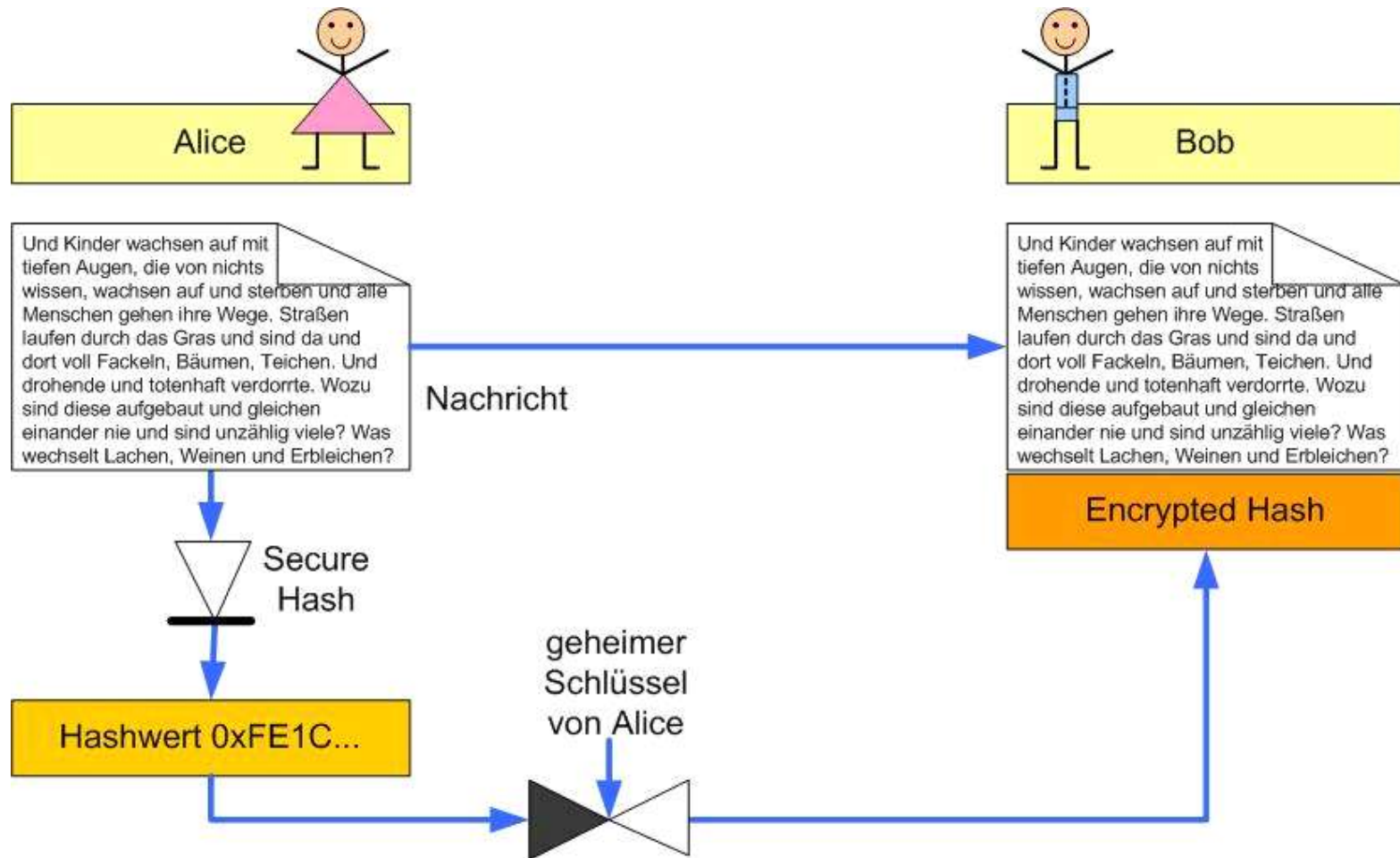
Kryptologie:

Kryptographische Hashfunktionen

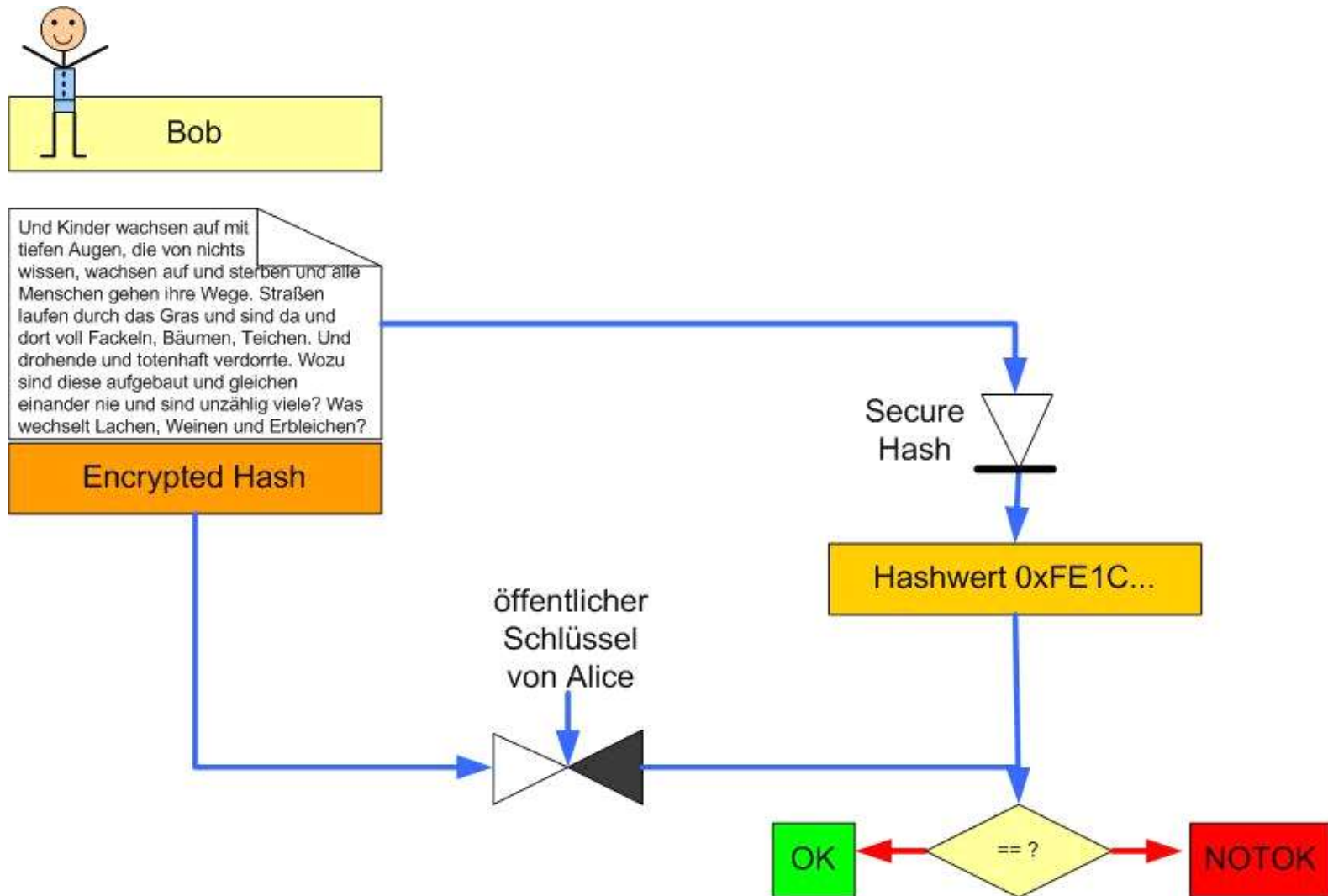
- Auch *Message Digest* (MD) genannt
- Zur kryptographischen Kompression von Eingangsdaten
 - ◆ Jedes geänderte Eingangsbit sollte eine große Änderung in den Ausgangsbits mit sich bringen
 - ◆ Somit kann eine Änderung in den Eingangsdaten erkannt werden
- Aus den komprimierten Daten können die Originaldaten nicht mehr hergestellt werden (Einwegfunktion)

Name	Eingangsblöcke	Hashwert
MD5	512 Bit	128 Bit
MDC-4	512 Bit	128 Bit
RIPEMD-160	512 Bit	160 Bit
SHA-1	512 Bit	160 Bit

Kryptologie: Digitale Signatur, Alice signiert eine Nachricht



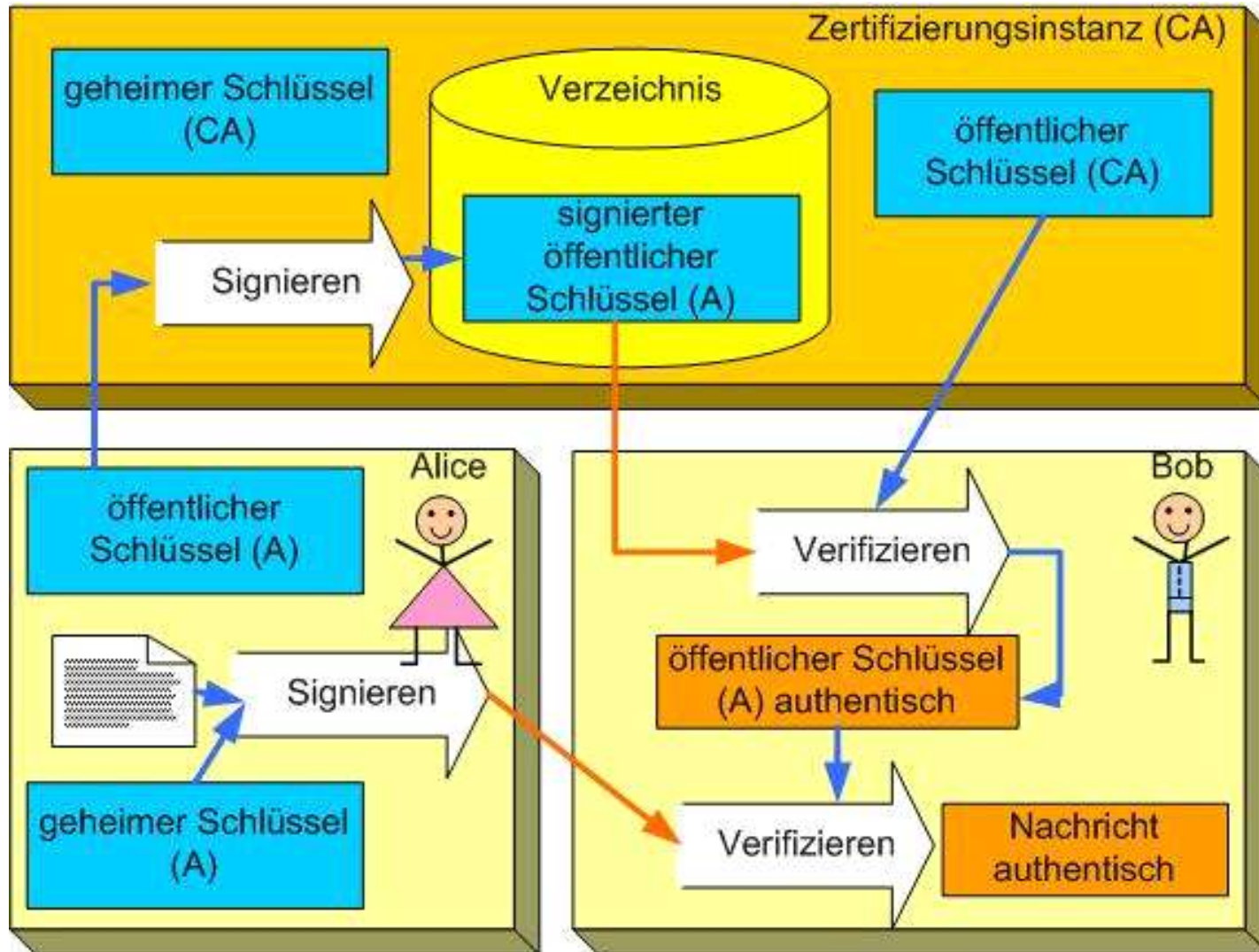
Kryptologie: Digitale Signatur, Bob verifiziert die Nachricht



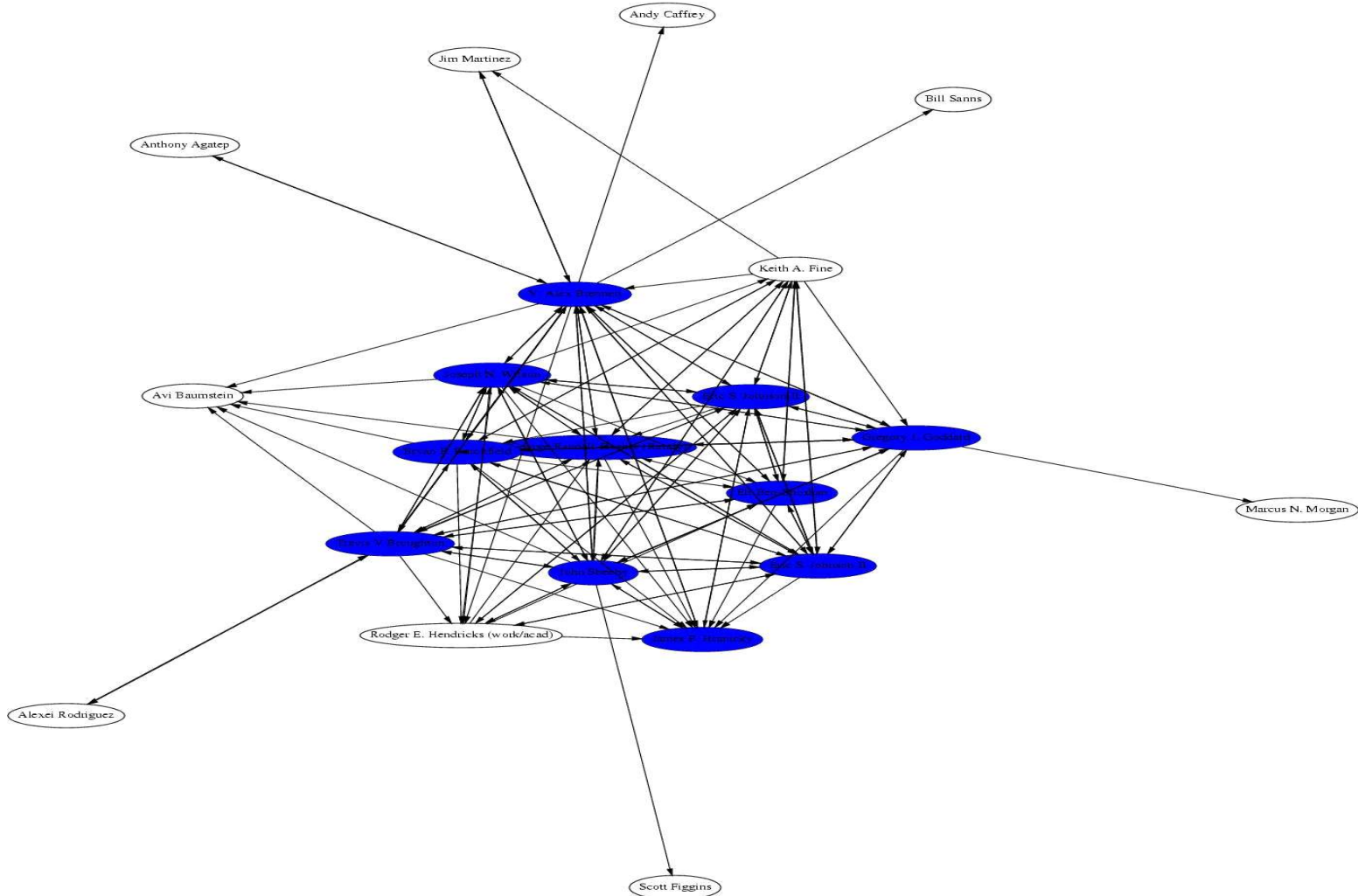
Kryptologie: PKI, Public Key Infrastructure

- Es muß überprüft werden können, ob
 - ◆ der erhaltene öffentliche Schlüssel authentisch ist
 - ◆ die Signatur eines Dokumentes nicht abgelaufen ist
- Hierfür wird eine *Public Key Infrastructure* aufgebaut
- Es gibt verschiedene Konzepte
 - ◆ Hierarchische PKI mit Zertifizierungsinstanzen (CA, Certificate Authority); für sichere Signaturen (“gesetzeskonforme Unterschriften”) vorgeschrieben
 - ◆ PKI mit Hilfe eines Web-of-Trust, also eines Vertrauensnetzwerkes, in dem sich unterschiedliche Leute kennen und die Authentizität von Dritten bestätigen

Kryptologie: Public Key Infrastructure, Hierarchisch mit CA



Kryptologie: Public Key Infrastructure, Web of Trust

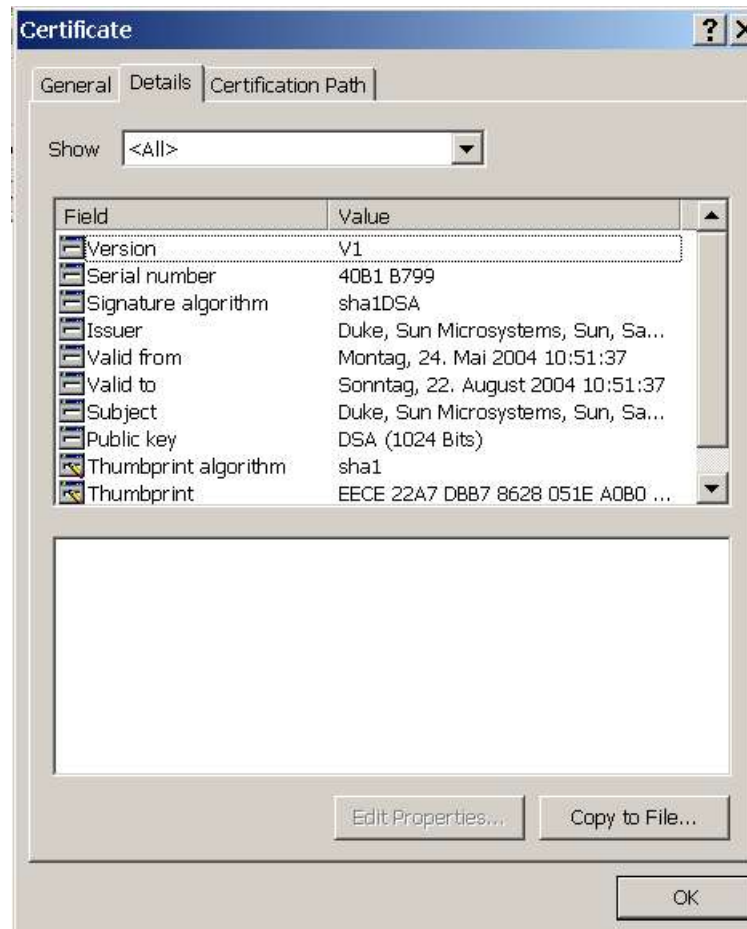


- Einen von einer CA unterschriebener öffentlicher Schlüssel mit dazugehöriger Signatur und zusätzlichen Parameter bezeichnet man als *Zertifikat*
- Ein *Trustcenter* erstellt und verwaltet Zertifikate mit Sperrlisten und kann optional auch die Schlüsselgenerierung vornehmen
 - ◆ Wichtigster Standard ist ITU-T X.509 (ISO/IEC 9594-8) mit Elementen wie
 - *X.509-Version (v3), Serial Number, Signature Algorithm Identifier, Issuer Name, Validity Period, Subject Name, Public Key, Signature, ...*
 - Typisches X.509-Zertifikat hat ~ 1 KByte
 - Kodierung nach ASN.1 (*Abstract Syntax Notation One*) mit Ausprägung TLV-DER (*Tag-Length-Value Distinguished Encoding Rule*)

- Einfache Erstellung eigener Zertifikate mit dem Java Hilfsprogramm `keytool`

- ◆ `C:\j2sdk1.4.2_01\bin>keytool -printcert -file dukecertfile.cer`
- ◆ Eigentümer: CN=Duke, OU=Sun Microsystems, O=Sun, L=San Diego, ST=California, C=US
- ◆ Aussteller: CN=Duke, OU=Sun Microsystems, O=Sun, L=San Diego, ST=California, C=US
- ◆ Seriennummer 40b1b799
- ◆ Gültig ab: Mon May 24 10:51:37 CEST 2004 bis: Sun Aug 22 10:51:37 CEST 2004
- ◆ Zertifikatfingerabdrücke:
- ◆ MD5:13:BA:88:D1:AF:D1:E1:6F:3D:8B:27:88:EF:53:53:39
- ◆ SHA1:EE:CE:22:A7:DB:B7:86:28:05:1E:A0:B0:EA:4C:2B:EF:9A:21:25:73

Anzeige im MS Internet Explorer



Kryptologie: Zertifikate, BER-Code

```
HexEdit - C:\j2sdk1.4.2_01\bin\dukecertfile.cer
File Edit Find View About
0 30 82 03 11 30 82 02 cf 02 04 40 b1 b7 99 30 0b 0...0.....@...0.
10 06 07 2a 86 48 ce 38 04 03 05 00 30 6e 31 0b 30 ..*.H.8....0n1.0
20 09 06 03 55 04 06 13 02 55 53 31 13 30 11 06 03 ...U....US1.0...
30 55 04 08 13 0a 43 61 6c 69 66 6f 72 6e 69 1 31 U....California1
40 12 30 10 06 03 55 04 07 13 09 53 61 6e 20 44 69 .0...U....San Di
50 65 67 6f 31 0c 30 0a 06 03 55 04 0a 13 03 53 75 ego1.0...U....Su
60 6e 31 19 30 17 06 03 55 04 0b 13 10 53 75 6e 20 n1.0...U....Sun
70 4d 69 63 72 6f 73 79 73 74 65 6d 73 31 0d 30 0b Microsystems1.0.
80 06 03 55 04 03 13 04 44 75 6b 65 30 1e 17 0d 30 ..U....Duke0...0
90 34 30 35 32 34 30 38 35 31 33 37 5a 17 0d 30 34 40524085137Z..04
a0 30 38 32 32 30 38 35 31 33 37 5a 30 6e 31 0b 30 0822085137Z0n1.0
b0 09 06 03 55 04 06 13 02 55 53 31 13 30 11 06 03 ...U....US1.0...
c0 55 04 08 13 0a 43 61 6c 69 66 6f 72 6e 69 61 31 U....California1
d0 12 30 10 06 03 55 04 07 13 09 53 61 6e 20 44 69 .0...U....San Di
e0 65 67 6f 31 0c 30 0a 06 03 55 04 0a 13 03 53 75 ego1.0...U....Su
f0 6e 31 19 30 17 06 03 55 04 0b 13 10 53 75 6e 20 n1.0...U....Sun
100 4d 69 63 72 6f 73 79 73 74 65 6d 73 31 0d 30 0b Microsystems1.0.
110 06 03 55 04 03 13 04 44 75 6b 65 30 82 01 b8 30 ..U....Duke0...0
120 82 01 2c 06 07 2a 86 48 ce 38 04 01 30 82 01 1f ,...*.H.8..0...
130 02 81 81 00 fd 7f 53 81 1d 75 12 29 52 df 4a 9c .....S..u.)R.J.
140 2e ec e4 e7 f6 11 b7 52 3c ef 44 00 c3 1e 3f 80 .....R<.D...?.
150 b6 51 26 69 45 5d 40 22 51 fb 59 3d 8d 58 fa bf .Q&iE]@"Q.Y=.X..
160 c5 f5 ba 30 f6 cb 9b 55 6c d7 81 3b 80 1d 34 6f ...0...U1...;.4o
170 f2 66 60 b7 6b 99 50 a5 a4 9f 9f e8 04 7b 10 22 .f`.k.P.....{."
180 c2 4f bb a9 d7 fe b7 c6 1b f8 3b 57 e7 c6 a8 a6 .0.....;W....
```

Kryptologie: Angriffe, Arten der Kryptanalyse

- ◆ Ciphertext-Only
 - Der Angreifer hat nur den Schlüsseltext zur Verfügung
- ◆ Known-Plaintext
 - Der Angreifer kennt Klartext-Schlüsseltext-Paare
- ◆ Chosen-Plaintext
 - Der Angreifer kann Klartext in das System einschleusen
 - Hier gibt es noch adaptive Verfahren
- ◆ Chosen-Key
 - Zusammenhänge zwischen Schlüsseln sind bekannt
- ◆ Kryptanalyse mit Gewalt
 - Der Kryptanalytiker bedroht, erpreßt oder quält den Schlüsselträger solange bis dieser den Schlüssel verrät
 - Bestechung nennt man *Angriff mit gekauftem Schlüssel*
 - Äußerst wirkungsvolle und oft schnelle Möglichkeiten, denen auf infrastrukturellem Wege begegnet werden muß

Kryptologie: Angriffe, Brechen einer Verschlüsselung

- Brute-Force
 - ◆ Ausprobieren aller möglichen Schlüssel
- Analyse mittels sprachlicher Häufigkeiten
 - ◆ Buchstaben, Silben, Konsonantenstellungen, ...
 - ◆ Häufigkeit im Deutschen:

a	6,51	n	9,78
b	1,89	o	2,51
c	3,06	p	0,79
d	5,08	q	0,02
e	17,40	r	7,00
f	1,66	s	7,27
g	3,01	t	6,15
h	4,76	u	4,35
l	7,55	v	0,67
j	0,27	w	1,89
k	1,21	x	1,89
l	3,44	y	0,04
m	2,53	z	1,13

- Suche durch massives Ausprobieren
 - ◆ Mit Rechnerclustern
 - ◆ Mit spezialisierter Hardware
- 1997: Erschöpfende Schlüsselsuche beim DES
 - ◆ 8 B mit je 7 Schlüsselbits = 56 Bit Schlüssellänge
 - ◆ $2^{56} = \sim 7,2 * 10^{16}$
- Lösung: längere Schlüssel
 - ◆ 3DES-2: 112 Bit
 - ◆ AES: 128, 192, 256 Bit
 - ◆ RSA: 512, 768, 1024, 2048, 4096 Bit

Kryptologie: Angriffe, Größenordnungen

Beispiel	Zahl
Wahrscheinlichkeit, vom Blitz erschlagen zu werden	1 : 9 Milliarden (2^{33})
Wahrscheinlichkeit für den Hauptgewinn in der US-Staatslotterie	1 : 4.000.000 (2^{22})
Wahrscheinlichkeit, am gleichen Tag den Hauptgewinn zu haben und vom Blitz erschlagen zu werden	1 : 2^{55}
Wahrscheinlichkeit, zu ertrinken	1 : 2^{16}
Wahrscheinlichkeit, bei einem Autounfall ums Leben zu kommen	1 : 88
Beginn der nächsten Eiszeit	2^{14} Jahre
Zeit bis die Sonne zur Nova wird	2^{30} Jahre
Anzahl der Atome in der Erde	2^{170}
Zeit bis sämtliche Materie zu Eisen zerfällt	$2^{10^{76}}$

- DFA, *Differential Fault Analysis* bzw. *Bellcore Attack*
- Durch Einstreuung eines Fehlers in Verschlüsselung kann Schlüssel berechnet werden
 - ◆ RSA: viele Messungen; RSA-CRT: eine
 - ◆ DSA: zwei Messungen; EC DSA: zwei
 - ◆ DES: 200 Messungen
- Abwehr
 - ◆ Zweifache Ausführung mit Vergleich der Ergebnisse oder
 - ◆ Multiplikation mit einer Zufallszahl (Basisblinding bei RSA)
 - Diese Zufallszahl wird durch die MODULO-Operation automatisch wieder „herausgerechnet“

- CRT: *Chinese Remainder Theorem*
 - ◆ beschleunigt die RSA-Berechnung durch Aufteilung in zwei Summenteile
 - ◆ RSA:
 - N Produkt von 2 zufälligen Primzahlen p und q
 - d Geheimer Schlüssel, e Öffentlicher Schlüssel
 - Verschlüsselung (Verifikation): $s := m^e \bmod N$
 - Entschlüsselung (Signatur): $m := s^d \bmod N$
 - ◆ Signatur mit RSA-CRT:
 - $S_1 := m^d \bmod p$
 - $S_2 := m^d \bmod q$
 - Signatur: $S := a*S_1 + b*S_2$
für vordefinierte Konstanten a, b aus \mathbf{Z}_N

Kryptologie: DFA für RSA mit CRT

- Wir kennen zwei Signaturen derselben Nachricht m
 - ◆ S , korrekte Signatur
 - ◆ S' , fehlerhafte Signatur
- Annahme, daß während Berechnung der Signatur S' mit CRT
 - ◆ $S_1' := m^d \bmod p$ falsch (i.e. $S_1 \neq S_1' \bmod p$)
 - ◆ $S_2' := m^d \bmod q$ richtig (i.e. $S_2 = S_2' \bmod q$)
- Deshalb
 - ◆ $S \neq S' \bmod p$
 - ◆ $S = S' \bmod q$, i.e. $S - S' = 0 \bmod q$
- Also
 - ◆ **$ggT(S - S', N) =: q$**
 - ◆ $N / q =: p$
 - ◆ d ableitbar aus N, p, q, e .

Protokolle: Übersicht

- Protokolle ermöglichen den sicheren Datenaustausch zwischen unterschiedlichen Partnern
- Für unterschiedliche Anwendungszwecke haben sich unterschiedliche Protokolle durchgesetzt
 - ◆ Schlüsselaustausch
 - Diffie/Hellman Key Exchange
 - ◆ E-Mail
 - PGP, S/MIME
 - ◆ Secure IP
 - SSL, HTTPS
 - ◆ e-Commerce
 - SET, EMV, Quick im Internet
 - ◆ Paketvermittlung
 - IPsec, VPN

- ◆ Phil Zimmermann entwickelte 1991 *Pretty Good Privacy*
 - Er führte das Programm wegen der kryptographischen Exportbeschränkung der USA in Buchform nach Europa
- ◆ Sehr weite Verbreitung, als Open Source GPG
- ◆ Wie auf der Folie „Schlüsselmanagement“ wird ein zufällig erzeugter Session Key asymmetrisch verschlüsselt und mit dem Dokument (symmetrisch mit dem Session Key verschlüsselt) ausgetauscht
- ◆ Die Anwender stellen ihre öffentlichen Schlüssel zur Verfügung (z.B. auf Websites)
- ◆ Anwender können sich die Schlüssel gegenseitig zertifizieren
 - Web of Trust
- ◆ Vorgeschlagene Mailerweiterungen heißen
 - PGP/MIME und OpenPGP (neuer)

Protokolle: PGP Signatur

```
Untitled - Notepad
File Edit Format Help
Hash: SHA1
Mein werter Herr Gemahl ist heute Nacht nicht zu Hause.
In feuriger Sehnsucht,
Alice.
-----BEGIN PGP SIGNATURE-----
Version: PGP 8.0.2
iQEUAwUBQLS0Qd4K5EvPraCiAQLwnQgAu7Qnr9AxDuFo3cry0ADS1PYe6ayUh4+w
FmXePdc2tR1K1XQt60sw/bk4Iuyc1Ywy04NUvM50TLz0h8W9276+CAoBThF+gLiQ
mXFMDvoHXydb2xQhSMbMUs/H7Y53rSGNHa5JqsGikppEB1oAn$JJHUI+Kh/ZmvPN
d5+nf441GuYDwJ1zjjQHey42NRp7pE9j0YzYuAZ3+gUCMmnpJmuxNTu0sF3/aAF
A/KUjn3ojiZt/BurjFBp58dzPTFzYggkdU0UU5IAL7uQMX5PhMdK0
61GSP7MGYc2dGRhZ5kr-fuHbydoRmHpqwr2036A4SFPr3g1EvoU16d
=MZsp
-----END PGP SIGNATURE-----
```

```
Text Viewer
*** PGP SIGNATURE VERIFICATION ***
*** Status: Good Signature
*** Signer: Alice Testkey <alice@nowhere.nn> (0xCFADA0A2)
*** Signed: 26.05.04 14:32:01
*** Verified: 26.05.04 14:32:27
*** BEGIN PGP VERIFIED MESSAGE ***
Mein werter Herr Gemahl ist heute Nacht nicht zu Hause.
In feuriger Sehnsucht,
Alice.
*** END PGP VERIFIED MESSAGE ***
Copy to Clipboard OK
```

Protokolle: PGP Verschlüsselung

The image shows a sequence of three windows illustrating PGP decryption:

- Untitled - Notepad:** Displays a PGP message starting with "-----BEGIN PGP MESSAGE-----" and "Version: PGP 8.0.2". The message body is a long string of base64-encoded data.
- PGPTray - Enter Passphrase:** A dialog box showing the public key used for encryption: "Bob Testkey <bob@nowhere.nn> (DH/2048)". It prompts the user to "Enter passphrase for your private key:" with a "Hide Typing" checkbox.
- Text Viewer:** Shows the results of decryption and signature verification:

```
*** PGP SIGNATURE VERIFICATION ***
*** Status: Good Signature
*** Signer: Alice Testkey <alice@nowhere.nn> (0xCFADA0A2)
*** Signed: 26.05.04 14:37:34
*** Verified: 26.05.04 16:48:01
*** BEGIN PGP DECRYPTED/VERIFIED MESSAGE ***

Mein werter Herr Gemahl ist heute Nacht nicht zu Hause.

In feuriger Sehnsucht,
Alice.

*** END PGP DECRYPTED/VERIFIED MESSAGE ***
```

Protokolle: S/MIME

- *Secure Multipurpose Internet Mail Extensions*
- Von RSA Labs vorgeschlagen
 - ◆ abgeleitet von eigenem Standard PKCS#7 (*Public Key Cryptosystems*)
- Derzeit S/MIMEv3
- Verwendet X.509v3 Zertifikate
 - ◆ Hierarchische Infrastruktur zur Zertifikatsverwaltung kann verwendet werden
- Geeignet zum Verschlüsseln und Signieren von Nachrichten
- Weit verbreitet und anerkannt
 - ◆ Mail-Clients wie Outlook und Mozilla verstehen S/MIME
 - ◆ OpenSSL kann ebenfalls mit S/MIME umgehen

- *Secure Socket Layer* wurde von Netscape entwickelt und steht nun als TLS (*Transport Layer Security*) allgemein zur Verfügung
 - ◆ Eine Adaption in WAP (WTLS) existiert
 - ◆ TLS wird oft noch SSLv3.1 genannt
- Transparente Arbeitsweise
 - ◆ Meist keine separate Benutzerinteraktion notwendig
- Einbettung im TCP/IP Protokollstack
 - ◆ Implementiert als eine Zwischenschicht zwischen Transportschicht und Anwendungsschicht
 - ◆ Herkömmliche Protokolle funktionieren ohne Änderung

- Aufbau einer TLS-Session

- - ◆ TLS Handshake Protocol

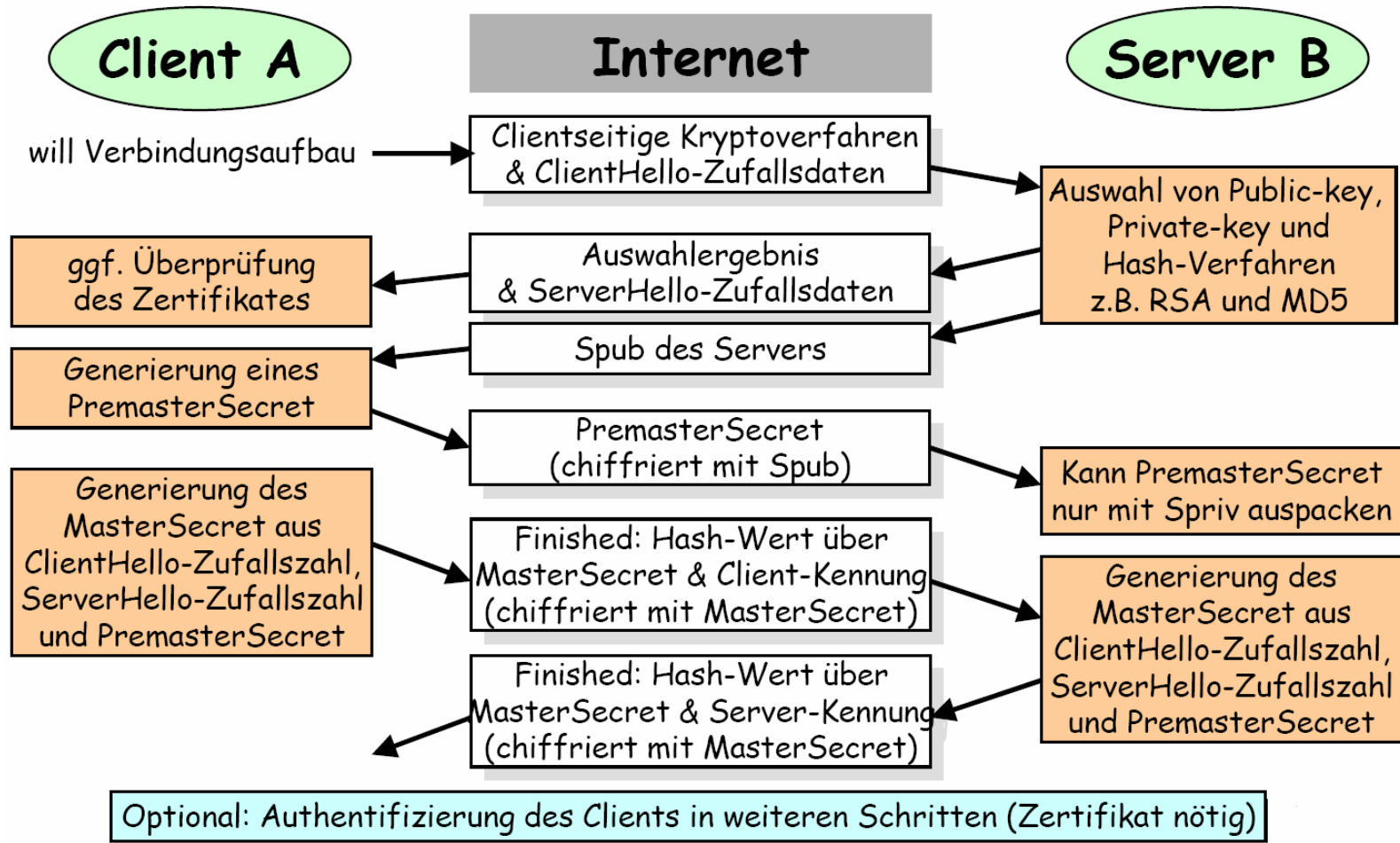
- - Browser und Server einigen sich auf kryptographische Algorithmen, die beiden genügen
 - Die Partner authentifizieren sich gegeneinander (jede Richtung ist optional)
 - Der Browser generiert einen Sitzungsschlüssel
 - Und schickt diesen asymmetrisch an den Server
 - Anschließend wird verschlüsselt kommuniziert

- - ◆ TLS Record Protocol

- - Jeder übertragene Record erhält einen Hashwert
 - Die Daten werden symmetrisch verschlüsselt

- Wird verwendet für
 - ◆ Asymmetrische Authentifizierung
 - ◆ Symmetrische Verschlüsselung (nach asymmetrischem Schlüsselaustausch)
- Sobald im Browser eine **https**-Verbindung aufgebaut werden soll, initiiert der Browser die TLS-Verbindung
- Es kann eingestellt werden, wer sich authentifiziert
 - ◆ Der Browser gegenüber dem Server
 - ◆ Der Server gegenüber dem Browser
 - ◆ Die Authentifizierung erfolgt über X.509-Zertifikate

Protokolle: SSL/HTTPS



- Vorteile

- ◆ Keine separate Software bei Kunden und Händlern nötig
- ◆ Keine eigene Kundenregistrierung notwendig
- ◆ Händler werden durch Zertifikate authentifiziert
- ◆ Weite Verbreitung
- ◆ Einfache Verwendung in Programmen

- Nachteile

- ◆ Der Händler erhält alle Daten des Kunden
- ◆ Fehlende Kundenauthentizität Händler trägt das Risiko
- ◆ Abrechnung wird nicht unterstützt
- ◆ Höhere Transaktionsgebühren für Händler

- *Secure Electronic Transactions*
- Geldbörse auf dem Endgerät (meist PC)
 - ◆ *Browser Wallet*
- Speziell für den Zahlungsverkehr
 - ◆ Von Visa und MasterCard entwickelt
- Drei Parteien
 - ◆ Kunde, Händler, Bank des Händlers
- Der Händler erhält nie unverschlüsselte Kundendaten
 - ◆ Er reicht die verschlüsselten Informationen direkt an seine Bank weiter
- Nur zertifizierte Parteien möglich
 - ◆ Absicherung über hierarchische PKI

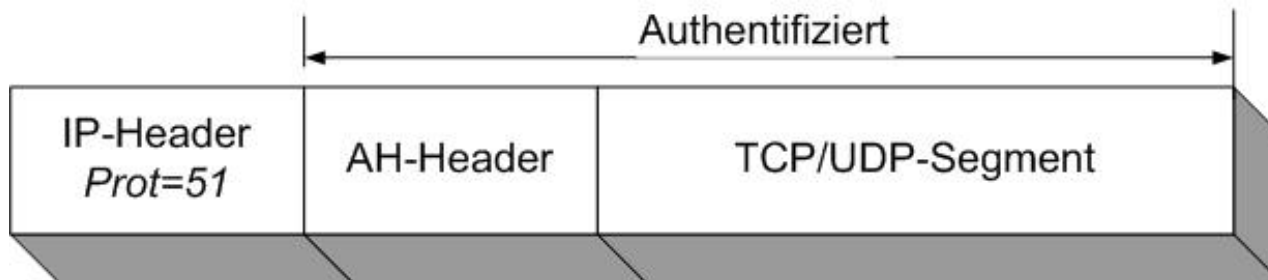
- Europay, MasterCard und Visa
- Ursprünglich Standard für Chips auf Karten im Zahlungsverkehr
- Erweiterungen erlauben das Zahlen mit Karte im Internet
 - ◆ Im Gegensatz zu SET sind keine Daten auf dem Endgerät
 - ◆ Der Chip (und damit ein Chipkartenleser) wird für die Bezahlung benötigt
 - ◆ Wie SET ein Modell mit drei Parteien
 - Kontoinhaber, Händler, Bank des Händlers

- Quick ist die österreichische Implementierung einer digitalen Geldbörse
 - ◆ Die erste flächendeckend eingesetzte digitale Geldbörse der Welt
 - ◆ Einführung vor EMV -> nicht voll kompatibel
- Ursprünglich nur an Terminals einsetzbar
- Erweiterung zur Zahlung im Internet
 - ◆ Quick im Internet
 - ◆ Ein Chipkartenleser wird benötigt
 - ◆ Durch die Chipkarte (*secure token*) sehr sicher

- ◆ *IP Security*
 - Sicherheit auf der Vermittlungsschicht
 - Kompliziertes Gebilde, viele RFCs
- ◆ Verschlüsselung der Daten aller IP-Pakete
 - Bietet generelle Sicherheit
- ◆ Zwei grundsätzliche Protokolle
 - AH-Protocol (*Authentication Header*)
 - ESP-Protocol (*Encapsulation Security Payload*)
- ◆ Vor dem Paketaustausch müssen Quell- und Zielhost einen Handshake durchführen
 - Logischer Kanal, SA (*Security Association*) wird aufgebaut. Dieser wird durch 3 Elemente identifiziert
 - Sicherheitsprotokoll: AH oder ESP
 - IP-Adresse für die Simplex-Verbindung (unidirektional)
 - 32 Bit Verbindungsidentifizierung, SPI (*Security Parameter Index*)

Protokolle: IPsec, AH-Protokoll

- ◆ Authentizität von Quellhost und Daten
- ◆ Im IP-Header zusätzlich ein AH-Header mit folgenden Feldern
 - *Nächster Header*
 - Tatsächlicher Protokolltyp der Daten
 - *Security Parameter Index*
 - *Sequenznummer*
 - Initialisiert mit 0, als Schutz gegen Replay-Attacken
 - *Authentifikationsdaten*
 - Signatur der Daten
- ◆ IP Protokoll wird auf 51 gesetzt
 - Zielhost weiß, daß es sich um AH-Paket handelt
 - Router können das Paket weiterhin zustellen



Protokolle: IPsec, ESP-Protokoll

- Geheimhaltung und Authentizität
- ESP Header ist ähnlich wie AH Header
 - ◆ Feld *Nächster Header* wird mit den Daten verschlüsselt, damit ein Angreifer nicht weiß, um welchen Pakettyp es sich handelt
- IP Protokoll wird auf 50 gesetzt
 - ◆ Aus denselben Gründen wie bei AH Protokoll

