

# Systeme II

## 7. Sicherheit

Christian Schindelhauer

Technische Fakultät

Rechnernetze und Telematik

Albert-Ludwigs-Universität Freiburg

(Version 04.07.2016)

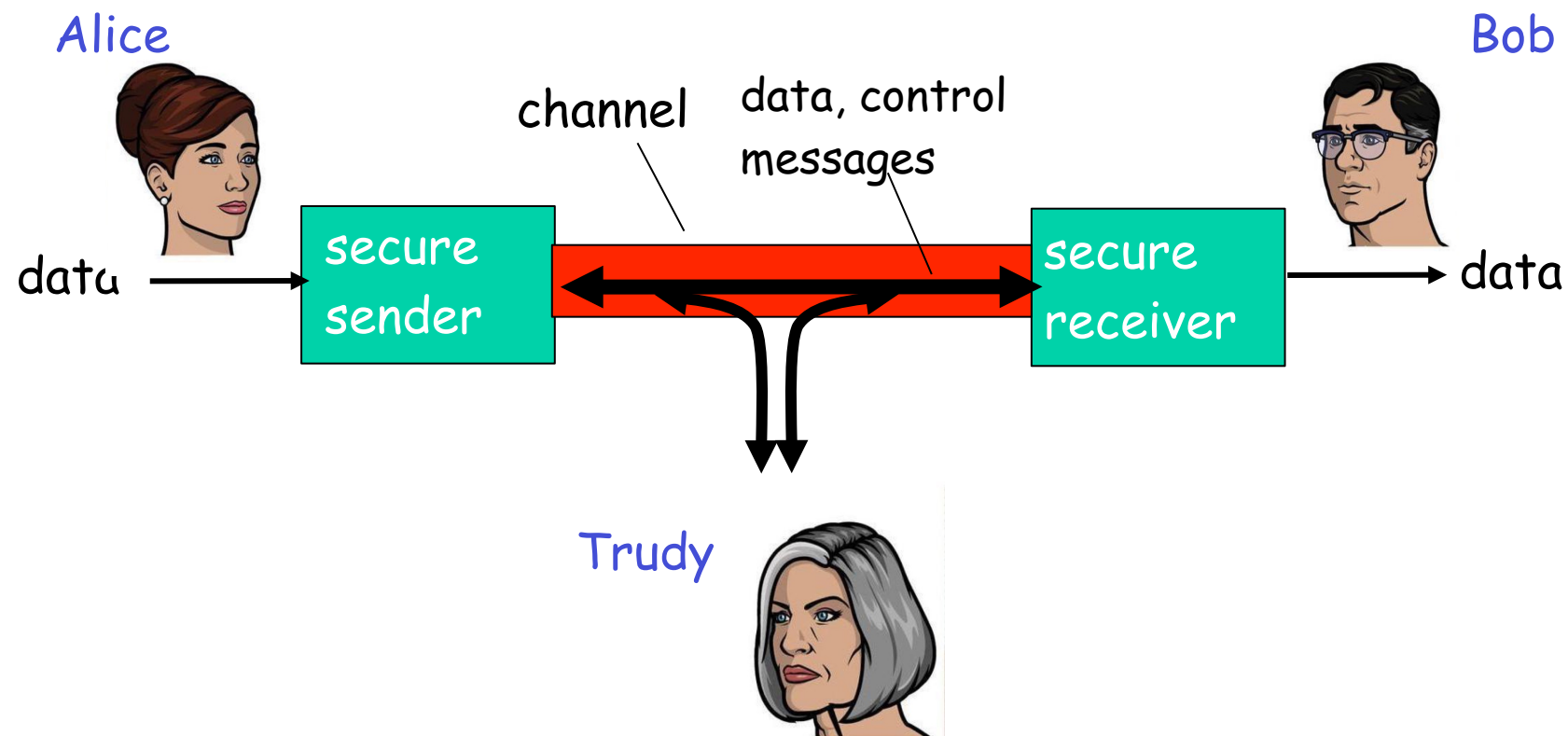
- Folien und Inhalte aus
  - Computer Networking: A Top Down Approach  
5th edition.  
Jim Kurose, Keith Ross  
Addison-Wesley, April 2009.
  - Copyright liegt bei den Autoren Kurose und Ross

- Grundlagen von Netzwerksicherheit
  - Kryptographie und deren vielfältige Einsatzmöglichkeiten
  - Authentifizierung
  - Message Integrity
- Sicherheit in der Praxis
  - Firewalls und Intrusion Detection
  - Sicherheit in Anwendungs-, Transport-, Vermittlungs- und Sicherungsschicht

- Vertraulichkeit (Confidentiality)
  - Nur der Sender, gewünschter Empfänger sollte den Nachrichteninhalt „verstehen“
- Authentifizierung
  - Sender und Empfänger möchten sich ihrer Identität versichern
- Integrität (message integrity)
  - Sender und Empfänger wollen, dass eine Nachricht nicht unbemerkt verändert werden
    - bei der Übertragung oder später
- Zugriff und Verfügbarkeit
  - von Diensten

# Freunde und Feinde: Alice, Bob und Trudy

- Standardnamen im Sicherheitsbereich
- Alice und Bob möchten „sicher“ kommunizieren
- Trude (In-Trude-r) möchte mithören, löschen, hinzufügen, verändern



# Wer steckt hinter Alice und Bob

---

- Echte Menschen
- Web-Browser
- Online-Banking-Clients und Servers
- DNS-Servers
- Routers, die Routing-Tabellen austauschen
- etc.

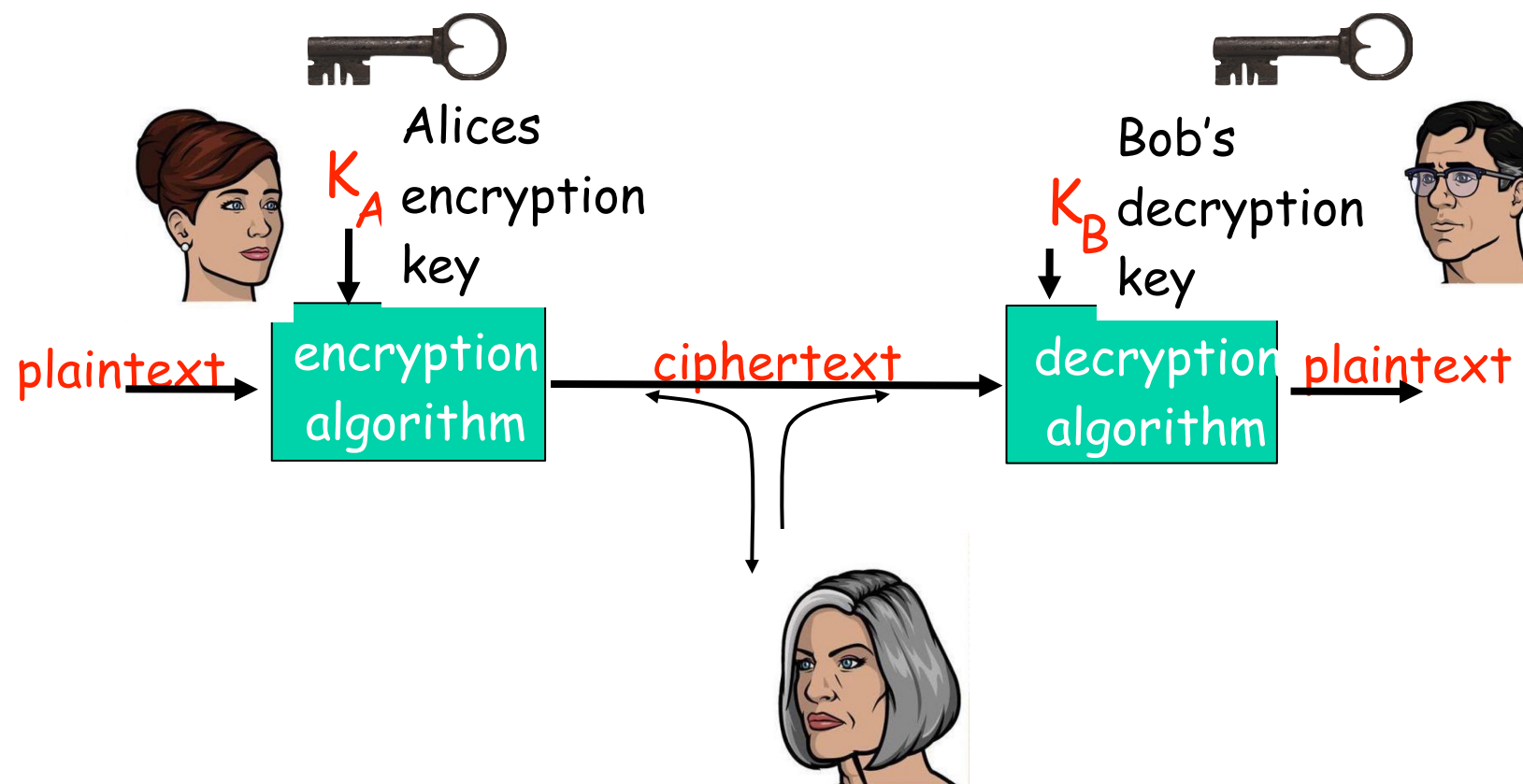
# Was kann ein böser Mensch so tun?

---

- Abhören (eavesdrop)
  - Nachrichten abfangen und lesen
- Einfügen von Nachrichten
  - Nachrichten werden in die bestehende Verbindung eingefügt
- Sich als jemand anders ausgeben (impersonation)
  - Quell-Adresse kann in einem Paket gefälscht werden
- Hijacking
  - Übernahme einer bestehenden Verbindung durch Ersetzen des Empfängers oder Senders
- Denial of Service
  - Dienst abschalten
    - durch Überlast oder direkten Angriff

# Ein kurzer Rundgang durch die Kryptographie

- $m$ : Originalnachricht (message)
- $K_A(m)$ : mit Schlüssel  $K_A$  verschlüsselte Nachricht
- $m = K_B(K_A(m))$





- Monoalphabetischer Schlüssel
  - ersetze jeden Buchstaben durch einen anderen
- Beispiel: Edgar Allen Poe „The Gold Bug“
  - 53305))6\*;4826)4)4;806\*;488¶60))85;1-(;:\*8-83(88)5\*
  - ;46(;88\*96\*?;8)\*(;485);5\*2:\*(;4956\*2(5\*-4)8¶8\*;40692
  - 85);)68)4;1(9;48081;8:81;4885;4)485528806\*81(9;48;
  - (88;4(?34;48)4;161;:188;?;
- Jedes Symbol steht für einen Buchstaben:
  - 8 = e
  - ; = h
  - ...

- ersetze jeden Buchstaben durch einen anderen

E.g.: Plaintext: bob. i love you. alice  
ciphertext: nkn. s gktc wky. mgsbc

- n monoalphabetische Schlüssel,  $M_1, M_2, \dots, M_n$
- Zyklus-Muster
  - e.g.,  $n=4$ ,  $M_1, M_3, M_4, M_3, M_2$ ;  $M_1, M_3, M_4, M_3, M_2$ ;
- Für jeden neuen Buchstaben aus den monoalphabetischen Schlüsseln einer ausgewählt
  - „aus“: a from  $M_1$ , u from  $M_3$ , s from  $M_4$
  - Schlüssel: n Schlüsselverfahren und der Zyklus

- Cipher-text only Attack
  - nur mit verschlüsselten Text
  - Zwei Ansätze:
    - Durchsuche alle Schlüssel und teste ob sie einen vernünftigen Text produzieren
    - Statistische Analyse des Schlüssels
- Known-Plaintext-Attack
  - mit der Originalnachricht und dem verschlüsselten Text
- Chosen Plaintext Attack
  - Trudy wählt den Text und lässt Alice ihn verschlüsseln
  - Trudy erhält den verschlüsselten Text

- Geheime Schlüssel sind die Sicherheitsgrundlage
  - Der Algorithmus ist bekannt
    - außer bei „security by obscurity“
- Public-Key-Cryptography
  - verwendet zwei Schlüssel
  - ein geheimer und ein öffentlicher Schlüssel
- Symmetrische Kryptographie
  - beide Seiten verwenden den selben geheimen Schlüssel
- Hash-Funktion
  - Ohne Schlüssel und ohne Geheimnis

- Stromchiffrierer (stream cipher)
  - verschlüsselt bitweise
- Blockchiffre, Blockverschlüsselung (block ciphers)
  - Originaltext wird in gleichgroße Blöcke unterteilt
  - Jeder Block wird einzeln kodiert

- Kombiniere jedes Bit eines Schlüsselstroms (key stream) mit dem Original bit
  - $m(i)$  = i-tes Bit der Nachricht
  - $ks(i)$  = i-tes Bit des Key Streams
  - $c(i)$  = i-tes bit des verschlüsselten Texts
- Verschlüsselung
  - $c(i) = ks(i) + m(i) \pmod{2}$   
     $= ks(i) \oplus m(i)$
- Entschlüsselung
  - $m(i) = ks(i) \oplus c(i)$

- RC4 ist ein populärer Streamchiffrierer
  - ausführlich analysiert und als sicher angesehen
  - Schlüssellänge: von 1 bis 256 Bytes
  - wird in WEP für 802.11 verwendet
  - kann in SSL verwendet werden



```
k[]: gegebene Schlüssel-Zeichenfolge der Länge 5 bis 256 Byte
L := Länge des Schlüssels in Byte
s[]: Byte-Vektor der Länge 256
Für i = 0 bis 255
    s[i] := i
j := 0
Für i = 0 bis 255
    j := (j + s[i] + k[i mod L]) mod 256
    vertausche s[i] mit s[j]
```

```
klar[]: gegebene Klartext-Zeichenfolge der Länge X
schl[]: Vektor zum Abspeichern des Schlüsseltextes
i := 0
j := 0
Für n = 0 bis X-1
    i := (i + 1) mod 256
    j := (j + s[i]) mod 256
    vertausche s[i] mit s[j]
    zufallszahl := s[(s[i] + s[j]) mod 256]
    schl[n] := zufallszahl XOR klar[n]
```

- Aus Wikipedia
  - <http://de.wikipedia.org/wiki/Rc4>

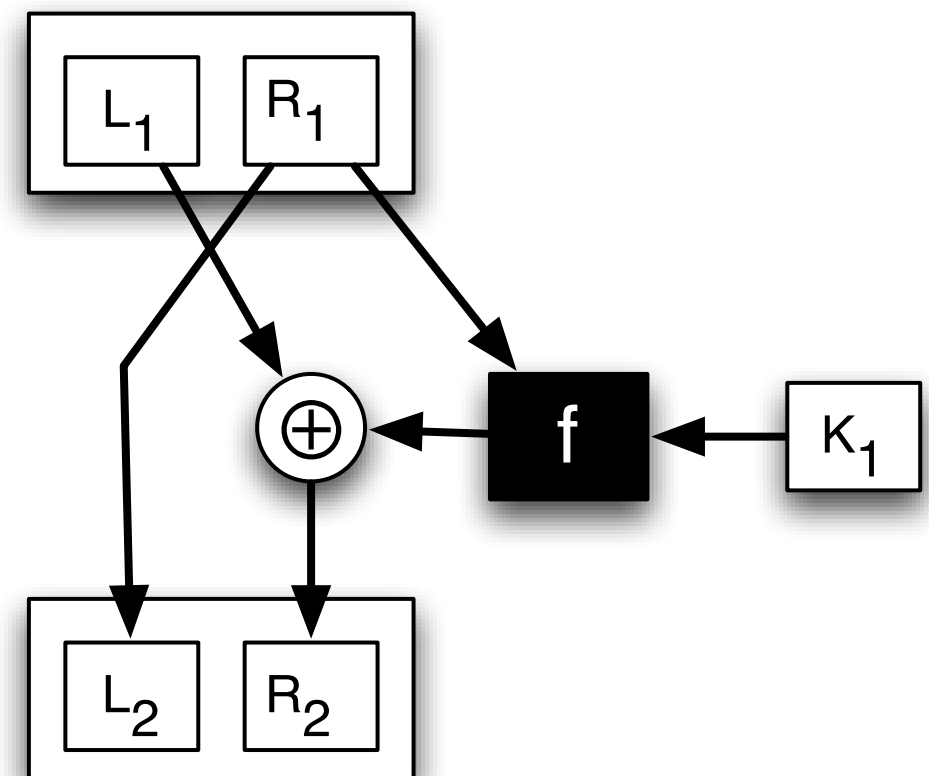
- Nachrichten werden in Blöcken von  $k$  bits verschlüsselt
  - z.B. 64-bit Blöcke
- Injektive Abbildung um den Quelltext in den  $k$ -bit verschlüsselten Text umzuwandeln
- Beispiel  $k=3$ :

<u>input</u>	<u>output</u>	<u>input</u>	<u>output</u>
000	110	100	011
001	111	101	010
010	101	110	000
011	100	111	001

- Wie viele mögliche Abbildungen gibt es für  $k$ -Bit Block-Chiffre?

- Wie viele mögliche Abbildungen gibt es für k-Bit Block-Chiffre?
  - Im allgemeinen:  $2^k!$
  - riesig für  $k=64$
  - und absolut sicher, wenn man sie zufällig auswählt
- Problem:
  - Die meisten dieser Abbildungen benötigen große Tabellen um sie zu berechnen
- Lösung
  - Statt einer Tabelle, verwendet man eine Funktion, die diese Tabelle simuliert
  - Dadurch verliert man möglicherweise wieder die Sicherhei

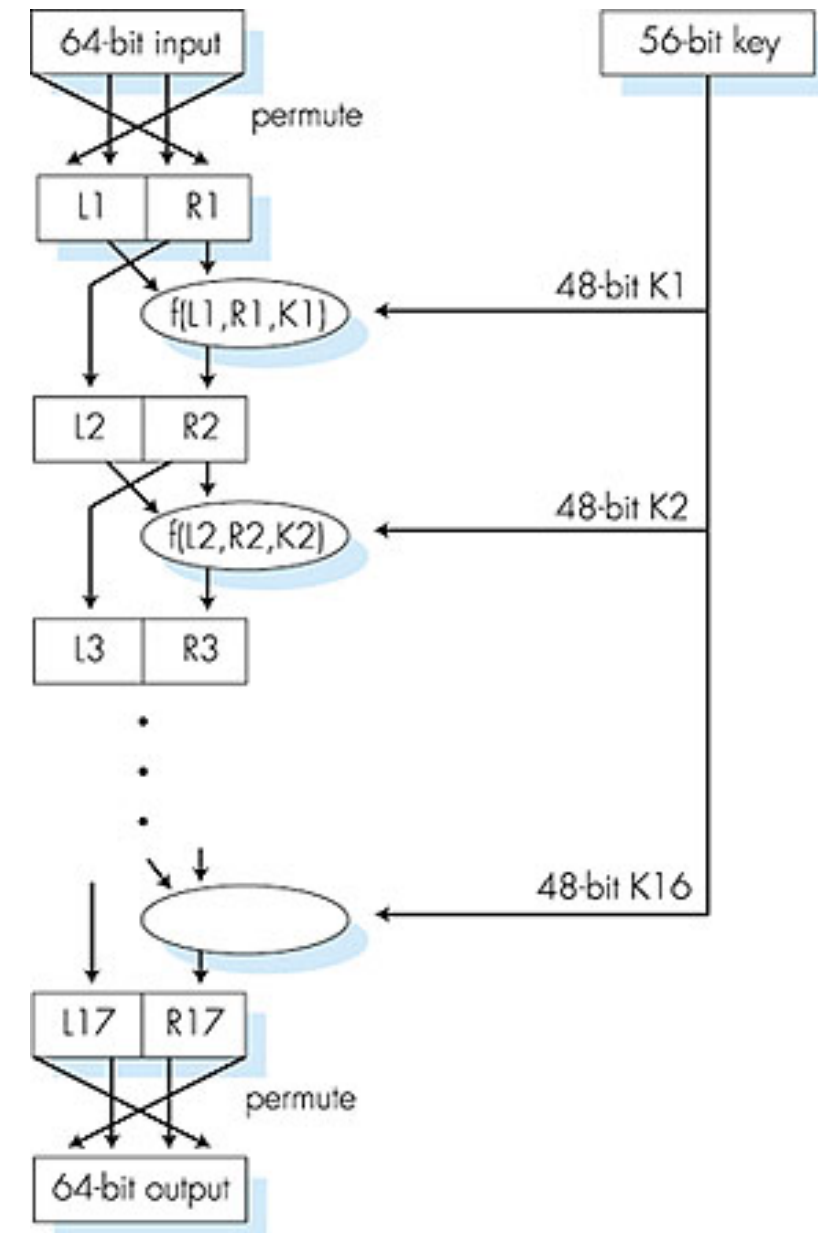
- Aufteilung der Nachricht in zwei Hälften  $L_1, R_1$ 
  - Schlüssel  $K_1, K_2, \dots$
  - Mehrere Runden: resultierender Code:  $L_n, R_n$
- Verschlüsselung
  - $L_i = R_{i-1}$
  - $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$
- Entschlüsselung
  - $R_{i-1} = L_i$
  - $L_{i-1} = R_i \oplus f(L_i, K_i)$
- $f$ : beliebige, komplexe Funktion



- Skipjack
  - 80-Bit symmetrischer Code
  - baut auf Feistel-Chiffre auf
  - wenig sicher
- RC5
  - Schlüssellänge 1-2048 Bits
  - Rivest Code 5 (1994)
  - Mehrere Runden der Feistel-Chiffre

# Digital Encryption Standard

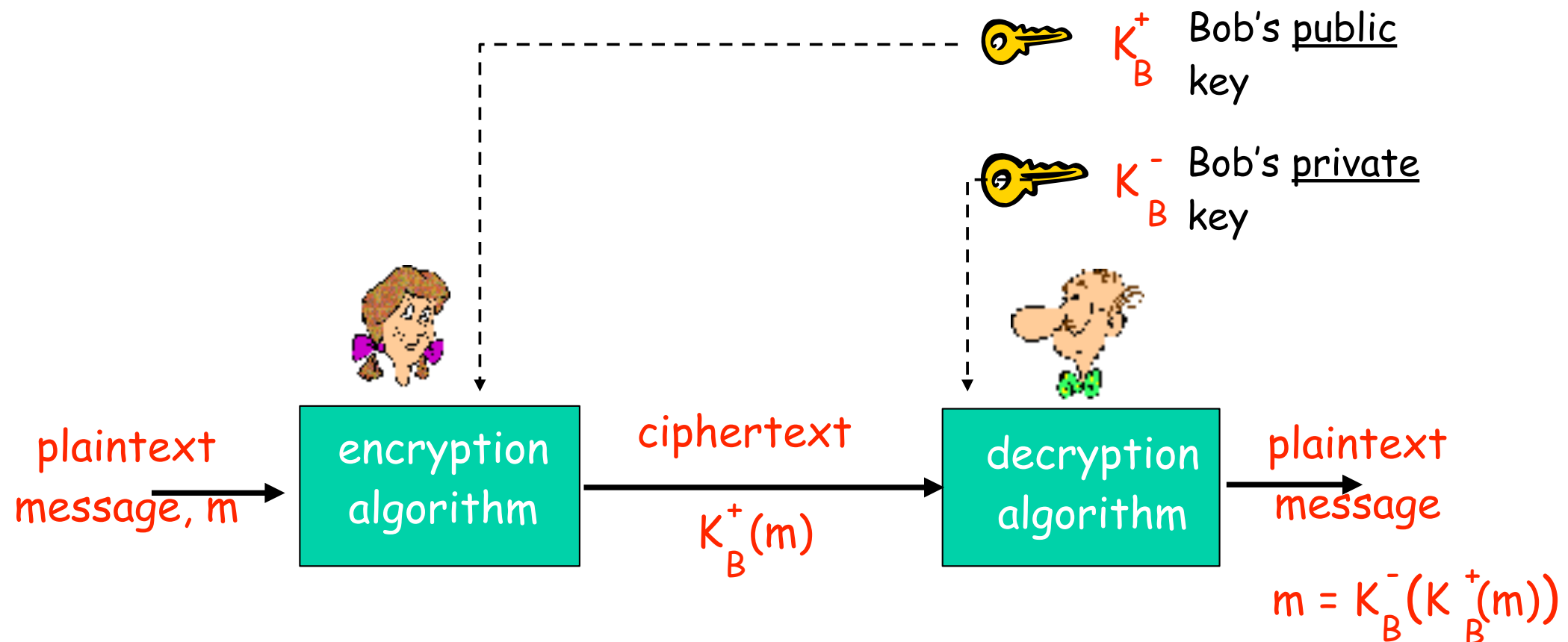
- Geschickt gewählte Kombination von
  - Xor-Operationen
  - Feistel-Chiffre
  - Permutationen
  - Table-Lookups
  - verwendet 56-Bit Schlüssel
- 1975 entwickelt von Wissenschaftlern von IBM
  - Mittlerweile nicht mehr sicher
    - leistungsfähigere Rechner
    - Erkenntnisse in Kryptologie
- Nachfolger: AES (2001)



- Geschickt gewählte Kombination von
  - Xor-Operationen
  - Feistel-Chiffre
  - Permutationen
  - Table-Lookups
  - Multiplikation in  $GF[2^8]$
  - symmetrische 128, 192 oder 256-Bit Schlüssel
- Joan Daemen und Vincent Rijmen
  - 2001 als AES unter vielen ausgewählt worden
  - bis heute als sicher erachtet



# Public key cryptography



- z.B. RSA, Ronald Rivest, Adi Shamir, Lenard Adleman, 1977
  - Diffie-Hellman, PGP
- Geheimer Schlüssel privat: kennt nur der Empfänger der Nachricht
- Öffentlichen Schlüssel offen: Ist allen Teilnehmern bekannt
- Wird erzeugt durch Funktion
  - $\text{keygen}(\text{privat}) = \text{offen}$
- Verschlüsselungsfunktion  $f$  und Entschlüsselungsfunktion  $g$ 
  - sind auch allen bekannt
- Verschlüsselung
  - $f(\text{offen}, \text{text}) = \text{code}$
  - kann jeder berechnen
- Entschlüsselung
  - $g(\text{privat}, \text{code}) = \text{text}$
  - nur vom Empfänger

- R. Rivest, A. Shamir, L. Adleman
  - On Digital Signatures and Public Key Cryptosystems, Communication of the ACM
- Verfahren beruht auf der Schwierigkeit der Primfaktorzerlegung
- 1. Beispiel:
  - $15 = ? * ?$
  - $15 = 3 * 5$
- 2. Beispiel:
  - $3865818645841127319129567277348359557444790410289933586483552047443 = 1234567890123456789012345678900209 * 3131313131313131313131313131300227$

- Bis heute ist kein effizientes Verfahren zur Primfaktorzerlegung bekannt
  - Aber das Produkt von Primzahlen kann effizient bestimmt werden
  - Primzahlen können effizient bestimmt werden
  - Primzahlen sehr häufig

- Erzeugung der Schlüssel
  - Wähle zufällig zwei Primzahlen  $p$  und  $q$  mit  $k$  bits ( $k \geq 500$ ).
  - $n = p \cdot q$
  - $e$  ist Zahl, die teilerfremd ist mit  $(p - 1) \cdot (q - 1)$ .
  - $d = e^{-1} = 1/e \bmod (p - 1)(q - 1)$ 
    - es gilt  $d \cdot e \equiv 1 \bmod (p - 1)(q - 1)$
- Public Key  $P = (e, n)$
- Secret Key  $S = (d, n)$

- Kodierung
  - Teile Nachricht in Blöcke der Größe  $2^{2k}$  auf
  - Interpretiere Block  $M$  als Zahl  $0 \leq M < 2^{2k}$
  - Chiffre:  $P(M) = M^e \bmod n$
- Dekodierung
  - $S(C) = C^d \bmod n$
- Korrektheit gilt nach dem kleinen Satz von Fermat
  - Für Primzahl  $p$  und von  $p$  teilerfremde Zahl  $a$  gilt:

$$a^p \equiv a \pmod{p}$$

- Bob wählt  $p=5$ ,  $q=7$ 
  - $n=35$ ,  $z=24$
  - $e = 5$
  - $d = 29$ 
    - $e d = 1 \bmod 24$
- Verschlüsselung von 8-Bit-Nachrichten

Bit pattern	$m$	$m$	$c=m^e \bmod n$
00001000	12	248 832	17

- Entschlüsselung

$c$	$c^d = 17^{29}$	$m=c^d \bmod n$
17	481968572106750915091411825223071697	12

- Berechnung von  $17^{29} \bmod 35$
- $29 = 11101_2$ 
  - $17^{29} = 17 \cdot (17^{14})^2 \bmod 35$
  - $17^{14} = (17^7)^2 \bmod 35$
  - $17^7 = 17 \cdot (17^3)^2 \bmod 35$
  - $17^3 = 17 \cdot (17)^2 \bmod 35$
- Einsetzen:
  - $17^3 = 4913 = 13 \bmod 35$
  - $17^7 = 17 \cdot 13^2 = 2873 = 3 \bmod 35$
  - $17^{14} = 3^2 = 9 \bmod 35$
  - $17^{29} = 17 \cdot 9^2 = 1377 = 12 \bmod 35$

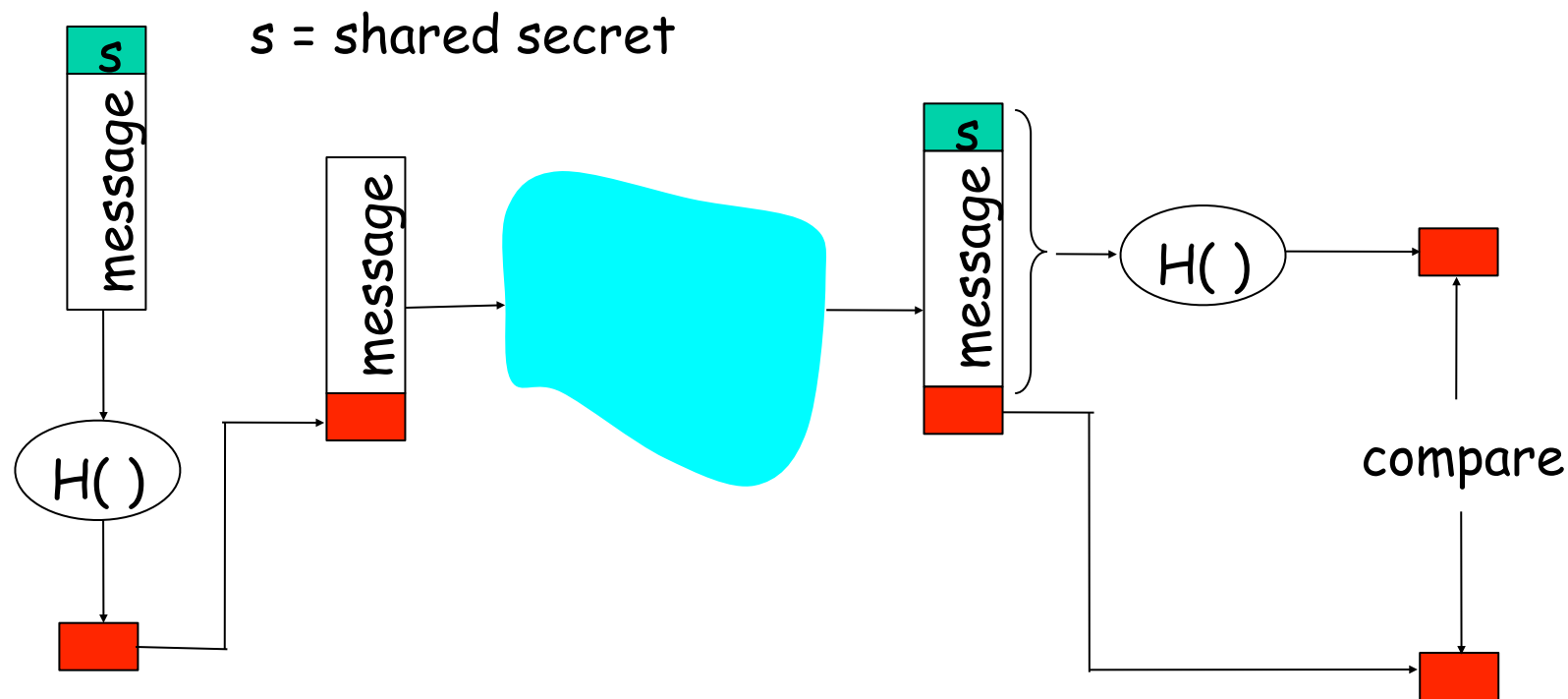


- Erlaubt den Kommunikationspartnern die Korrektheit und Authentizität der Nachricht zu überprüfen
  - Inhalt ist unverändert
  - Urheber ist korrekt
  - Nachricht ist keine Wiederholung
  - Reihenfolge der Nachrichten ist korrekt
- Message Digests

- z.B. SHA-1, SHA-2, MD5
- Ein kryptographische Hash-Funktion  $h$  bildet einen Text auf einen Code fester Länge ab, so dass
  - $h(\text{text}) = \text{code}$
  - es unmöglich ist einen anderen Text zu finden mit:
    - $h(\text{text}') = h(\text{text})$  und  $\text{text} \neq \text{text}'$
- Mögliche Lösung:
  - Verwendung einer symmetrischen Kodierung

- MD5 ist sehr verbreitet (RFC 1321)
  - berechnet 128-bit Nachricht
  - unsicher
- SHA-1 auch gebräuchlich
  - US standard [NIST, FIPS PUB 180-1]
  - 160-bit Message Digest
  - nicht mehr als sicher angesehen
- SHA-2
  - SHA-256/224
  - SHA-512/384
  - bis jetzt (2012) als sicher angesehen
- SHA-3
  - 2011 veröffentlicht

# Message Authentication Code (MAC)



- Authentifiziert Absender
- Überprüft Nachrichtenintegrität
- Keine Verschlüsselung
- “keyed hash”
- Notation:  $MDm = H(s \parallel m)$  ; sende  $m \parallel MDm$

# HMAC (Keyed-Hash Message Authentication Code)

---

- Populärer MAC-Standard
- Sicher gegen Anhängen von Nachrichten

$$HMAC_K(N) = H\left((K \oplus opad) \parallel H((K \oplus ipad) \parallel N)\right)$$

- Nachricht  $N$
- geheimer Schlüssel  $K$
- Konstante  $opad$  und  $ipad$
- Erhöht Sicherheit gegen angreifbare Hash-Codes
  - wird in TLS und IPsec verwendet

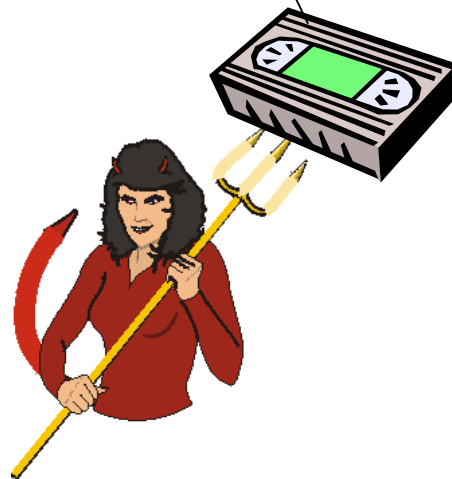
- Versicherung, dass der Kommunikationspartner korrekt ist
- Angenommen Alice und Bob haben ein gemeinsames Geheimnis, dann gibt MAC eine Authentifizierung der Endpunkte
  - (end-point authentication)
  - Wir wissen, dass Alice die Nachricht erzeugt hat
  - Aber hat sie sie auch abgesendet?

# Playback-Attacke

MAC =  
 $f(\text{msg}, s)$

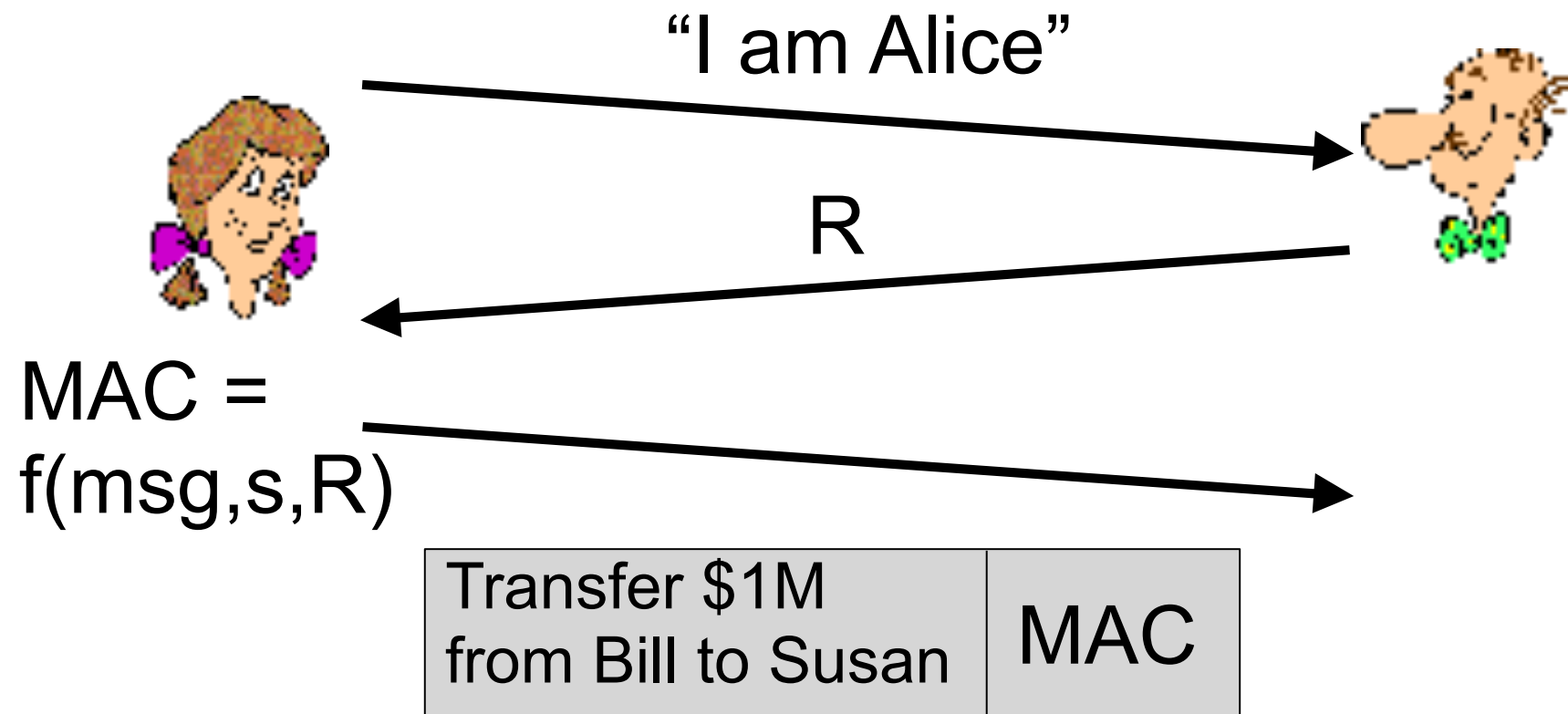


Transfer \$1M from Bill to Trudy	MAC
-------------------------------------	-----



Transfer \$1M from Bill to Trudy	MAC
-------------------------------------	-----

# Verteidigung gegen die Playback- Attacke: nonce (use only once)






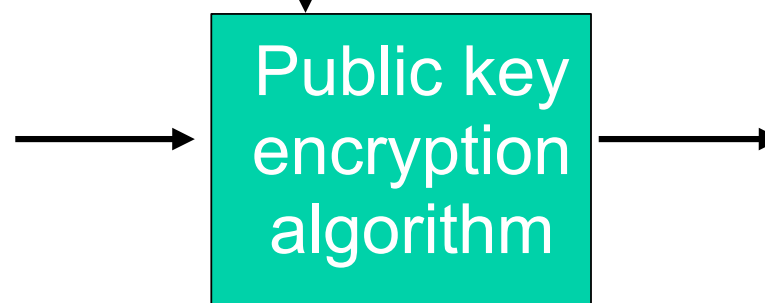
- Kryptographischer Algorithmus analog zu handgeschriebenen Unterschriften
  - nur sicherer
- Absender (Bob) unterschreibt digital das Dokument
  - bestätigt seine Urheberschaft
- Ziel ist ähnlich wie MAC
  - aber mit Hilfe von Public-Key-Kryptographie
  - verifizierbar, nicht fälschbar:
    - Empfänger (Alice) kann anderen beweisen, dass Bob und sonst niemand das Dokument unterschrieben hat

- Digitale Signaturen
  - Unterzeichner besitzt einen geheimen Schlüssel
  - Dokument wird mit geheimen Schlüssel unterschrieben
  - und kann mit einem öffentlichen Schlüssel verifiziert werden
  - Öffentlicher Schlüssel ist allen bekannt
- Beispiel eines Signaturschemas
  - m: Nachricht
  - Unterzeichner
    - berechnet  $h(\text{text})$  mit kryptographischer Hashfunktion
    - und veröffentlicht m und  
signatur =  $g(\text{privat}, h(\text{text}))$ , für die Entschlüsselungsfunktion g
  - Kontrolleur
    - berechnet  $h(\text{text})$
    - und überprüft  $f(\text{offen}, \text{signatur}) = h(\text{text})$ , für die asymmetrische Verschlüsselungsfunktion g

Bob's message,  $m$

Dear Alice  
Oh, how I have missed you. I  
think of you all the time! ...  
(blah blah blah)  
Bob

  $K_B^-$  Bob's private  
key

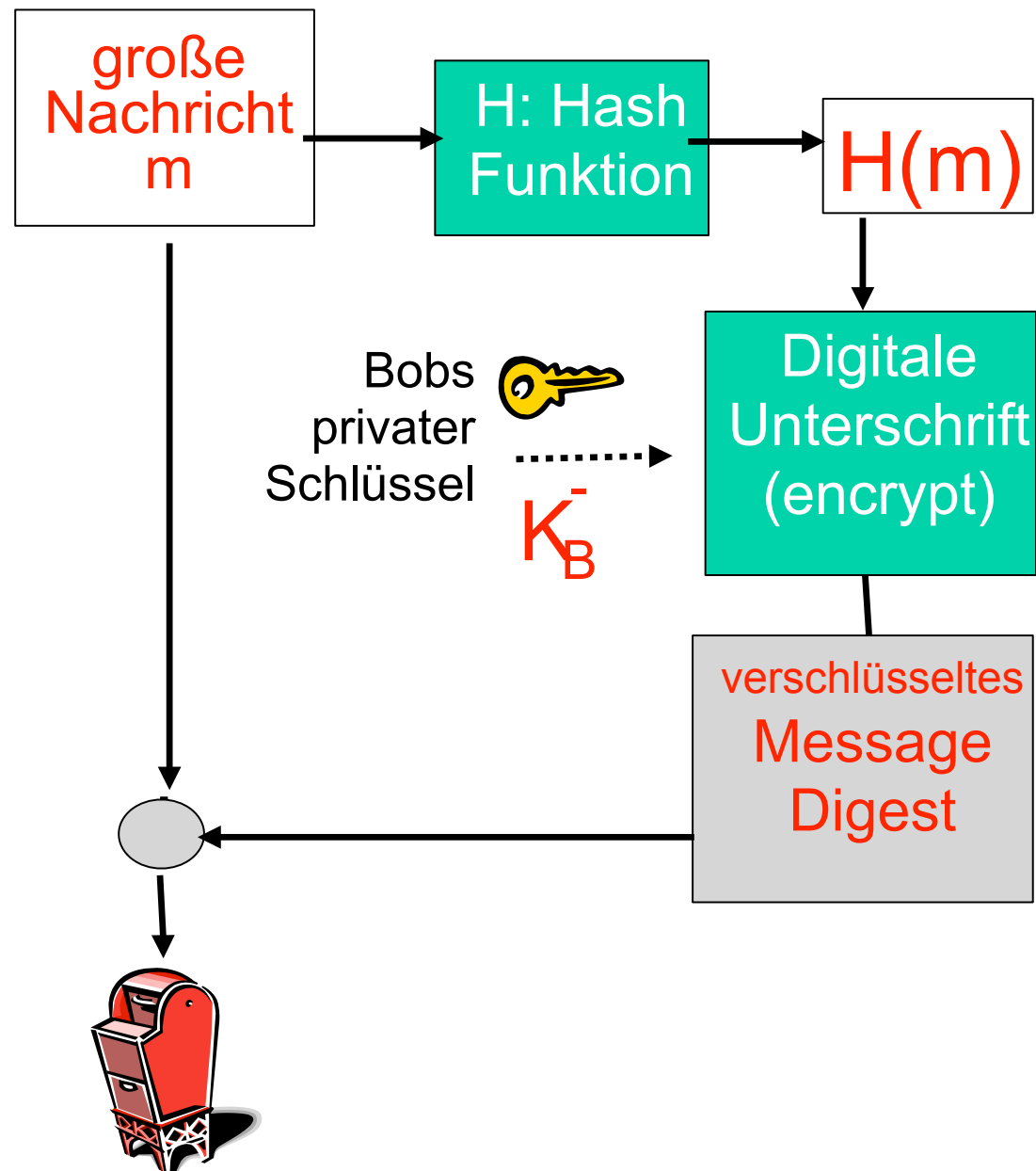


$K_B^-(m)$

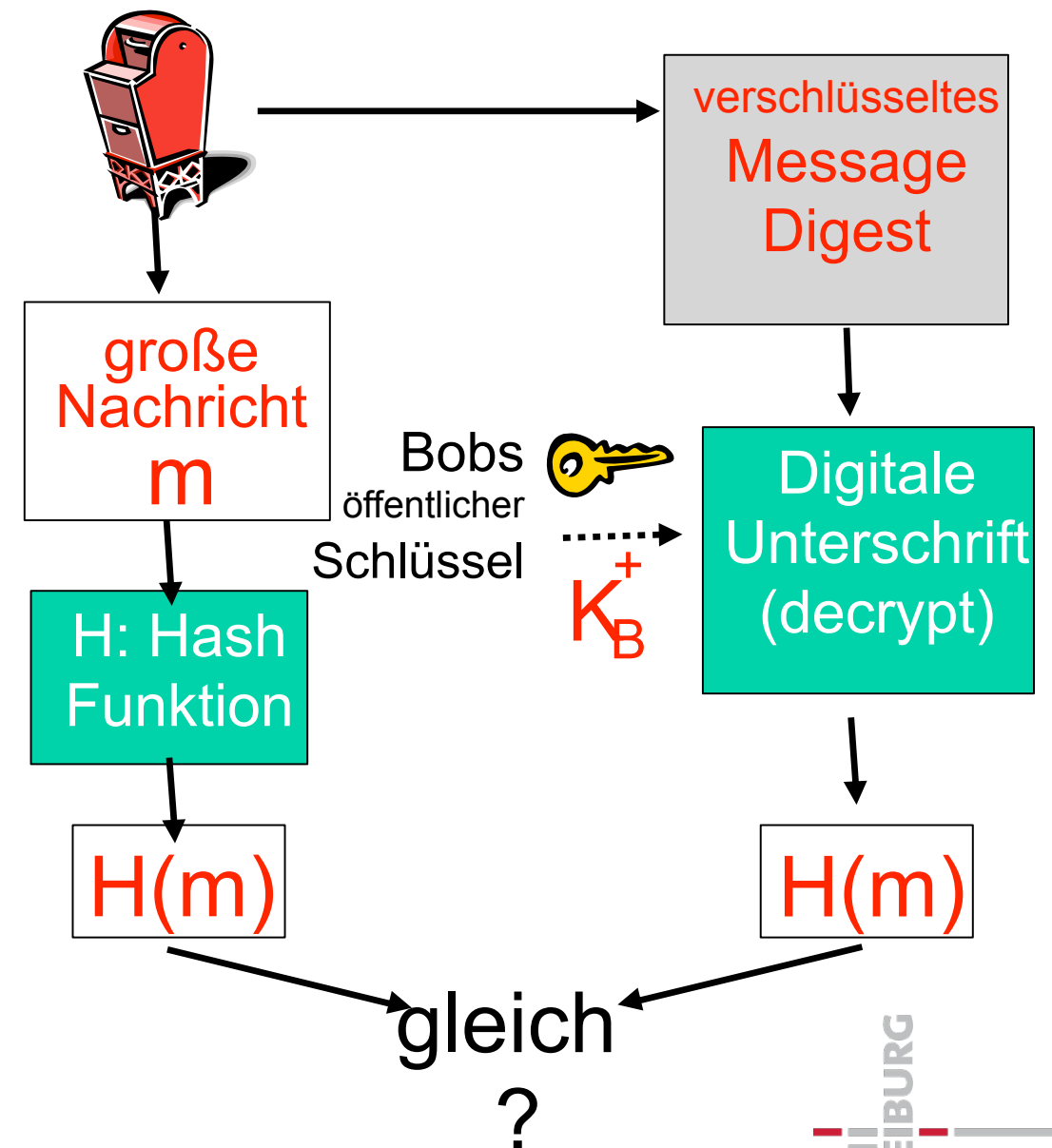
Bob's message,  $m$ ,  
signed (encrypted)  
with his private key

# Digitale signature = signiertes Message Digest

Bob sendet eine digital unterschriebene Nachricht



Alice überprüft die Unterschrift und die Korrektheit der Nachricht



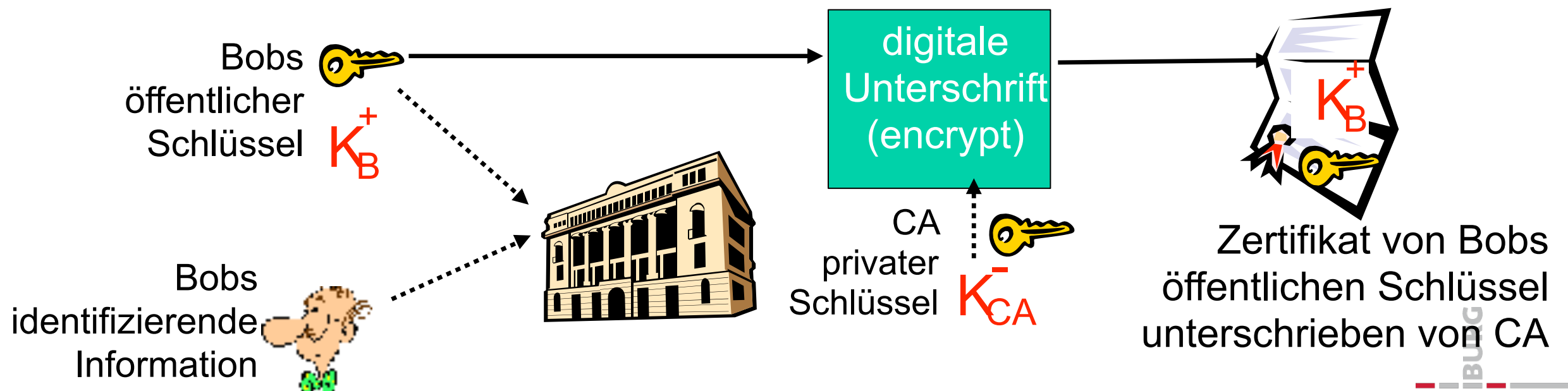
- Angenommen Alice erhält
  - die Nachricht  $m$
  - mit digitaler Unterschrift  $K_B^-(m)$
- Alice überprüft  $m$ 
  - mit den öffentlichen Schlüssel von Bob
  - Ist  $K_B^+(K_B^-(m)) = m$ ?
- Falls  $K_B^+(K_B^-(m)) = m$ 
  - dann hat jemand Bobs geheimen Schlüssel
- Alice verifiziert daher, dass
  - Bob hat  $m$  unterschrieben
  - Niemand anders hat  $m$  unterschrieben
  - Bob hat  $m$  und nicht  $m' \neq m$  unterschrieben
- Unleugbarkeit
  - Alice kann mit  $m$  und der Unterschrift vor Gericht gehen und beweisen, dass Bob  $m$  unterschrieben hat

- Motivation: Trudy spielt Bob einen Pizza-Streich
- Trudy bestellt per e-mail order:
  - „Liebe Pizzeria, schick mir bitte vier Pepperoni-Pizza.  
vielen Dank Bob“
- Trudy unterschreibt mit ihrem privaten Schlüssel
- Trudy sendet die Bestellung zur Pizzeria
- Trudy sendet der Pizzeria ihren öffentlichen Schlüssel
  - behauptet aber er gehöre Bob
- Die Pizzeria überprüft die Unterschrift
  - Bob mag gar keine Pepperoni

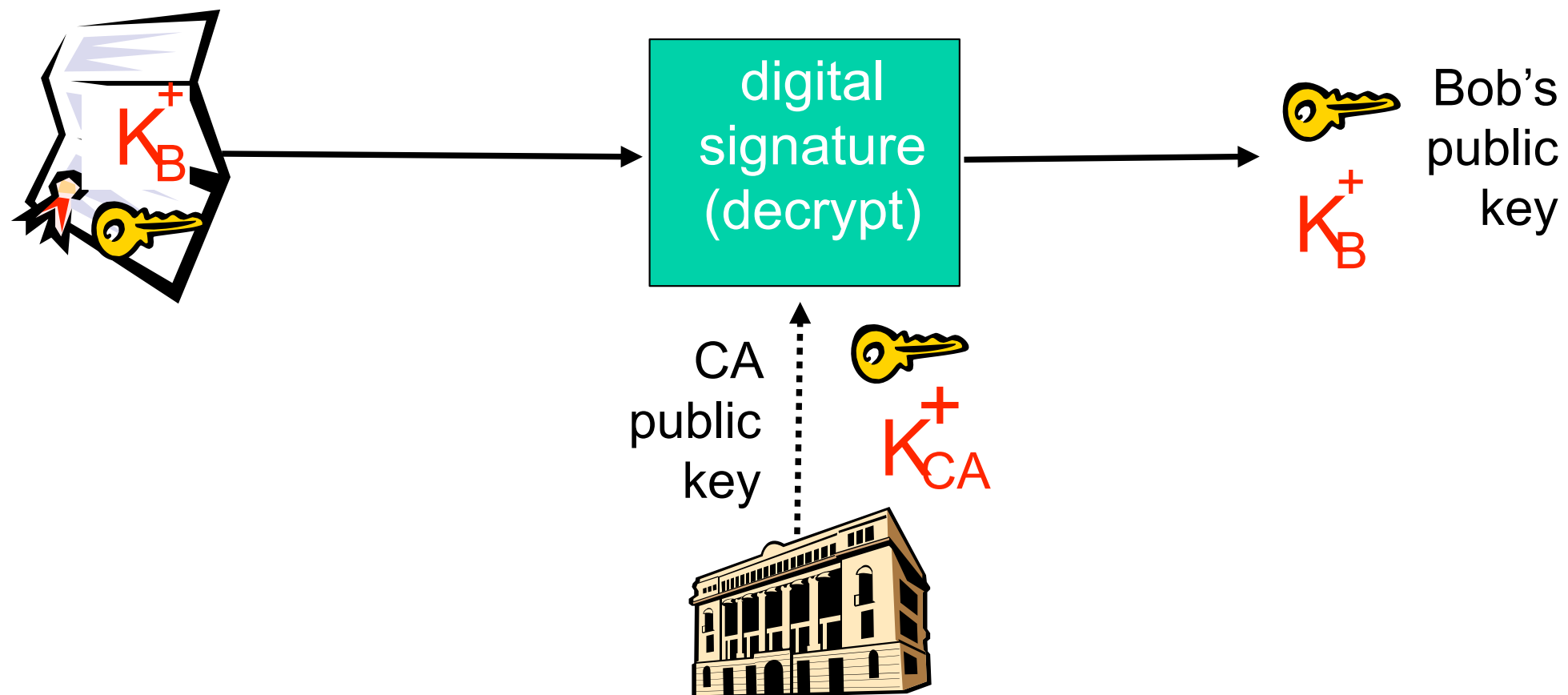
# Zertifizierungsstelle

## Certification Authorities (CA)

- Zertifizierungsstelle (Certification authority – CA): verknüpft öffentlichen Schlüssel mit der Entität (Person, Service, Router) E
- E registriert seinen öffentlichen Schlüssel mit CA
  - E „beweist seine Identität“ der Zertifizierungsstelle
  - CA erzeugt eine Zertifizierungsverknüpfung von E mit seinem öffentlichen Schlüssel
  - Zertifikat mit E's öffentlichen Schlüssel wird von der CA digital unterschrieben:
    - „Das ist der öffentliche Schlüssel von E“



- Wenn Alice Bobs öffentlichen Schlüssel möchte
  - erhält Bobs Zertifikat
  - wendet CA's öffentlichen Schlüssel auf Bobs Zertifikat an
    - Alice erhält Bobs öffentlichen Schlüssel





- Hauptstandard X.509 (RFC 2459)
- Zertifikat enthält
  - Name des Ausstellers (Issuer name)
  - Name der Entität, Adresse, Domain-Name, etc.
  - Öffentlicher Schlüssel der Entität
  - Digitale Unterschrift (unterschrieben mit dem geheimen Schlüssel des Ausstellers)
- Public-Key Infrastruktur (PKI)
  - Zertifikate und Zertifizierungsstellen

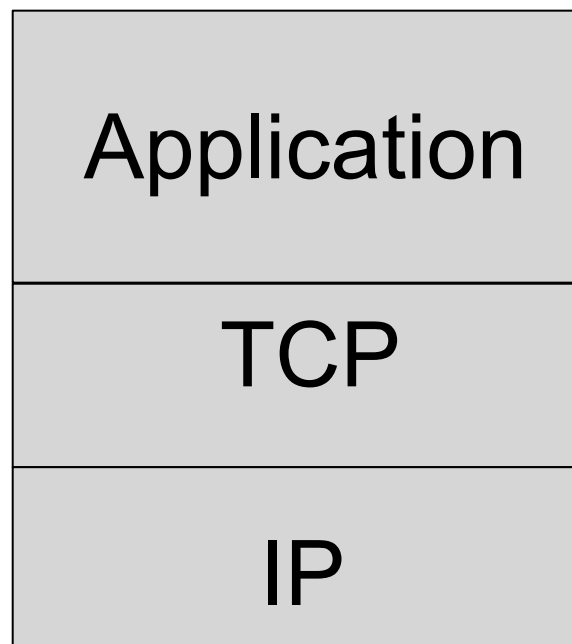
- Weit verbreitetes Sicherheitsprotokoll
  - Unterstützt durch alle Browser und Web-Server
  - https
  - Jährlich Transaktionen im Wert von Zigmilliarden Euro über SSL
- 1993 entworfen von Netscape
- Aktueller Name
  - TLS: transport layer security, RFC 2246
- Gewährleistet
  - Vertraulichkeit (Confidentiality)
  - Nachrichtenintegrität (Integrity)
  - Authentifizierung

- Ursprüngliche Motivation
  - Web E-Commerce Transaktionen
  - Verschlüsselung (Credit-Karte)
- Web-server Authentifizierung
  - Optional Client Authentifizierung
- Kleinstmöglicher Aufwand für Einsteiger
- In allen TCP Anwendungen verfügbar
  - Secure socket interface

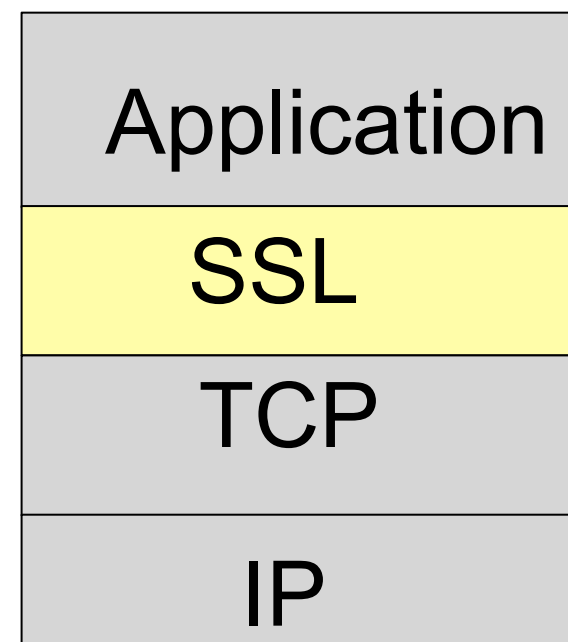
- DES – Data Encryption Standard: Block
- 3DES – Triple strength: Block
- RC2 – Rivest Cipher 2: Block
- RC4 – Rivest Cipher 4: Stream
  
- Auch Public-Key-Verschlüsselung
  - RSA

- Cipher Suite
  - Public-key Algorithmus
  - Symmetrische Verschlüsselungsalgorithmus
  - MAC Algorithmus
- SSL unterstützt mehrere Kodierungsverfahren
- Verbindungsvereinbarung (Negotiation)
  - Client und Server einigen sich auf ein Kodierungsverfahren
- Client bietet eine Auswahl an
  - Server wählt davon eines

- SSL stellt eine Programm-Interface für Anwendungen zur Verfügung
- C and Java SSL Bibliotheken/Klassen verfügbar



Normale Anwendung



Anwendung  
mit SSL

- Ziel
  - Server Authentifizierung
  - Verbindungsvereinbarung:
    - Einigung auf gemeinsames kryptographische Verfahren
  - Schlüsselaustausch
  - Client Authentifizierung (optional)

# SSL: Handshake (2)

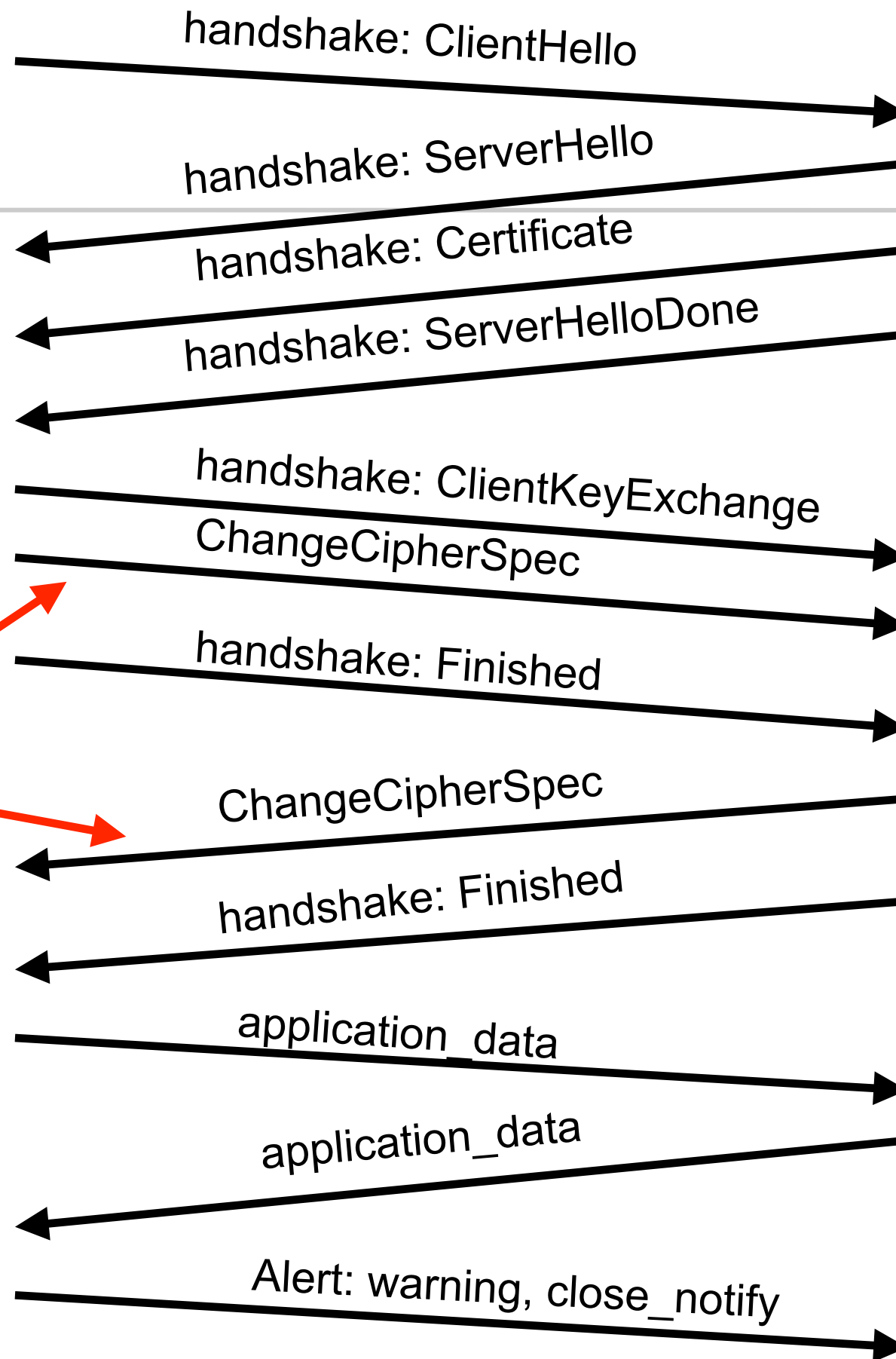
- Client sendet
  - Liste unterstützter Krypto-Algorithmen
  - Client nonce (salt)
- Server
  - wählt Algorithmen von der Liste
  - sendet zurück: Wahl + Zertifikat + Server Nonce
- Client
  - verifiziert Zertifikat
  - extrahiert Servers öffentlichen Schlüssel
  - erzeugt pre\_master\_secret verschlüsselt mit Servers öffentlichen Schlüssel
  - sendet pre\_master\_secret zum Server
- Client und Server
  - berechnen unabhängig die Verschlüsselungs- und MAC-Schlüssel aus pre\_master\_secret und Nonces
- Client sendet ein MAC von allen Handshake-Nachrichten
- Server sendet ein MAC von allen Handshake-Nachrichten





Ab hier ist  
alles verschlüsselt

TCP Fin folgt



- Client Nonce, Server Nonce und pre-master secret werden in Pseudozufallsgenerator gegeben
  - Ausgabe: Master Secret
- Master Secret und neue Nonces werden in anderen Pseudozufallsgenerator mit Ausgabe: “key block”
- Key block:
  - Client MAC key
  - Server MAC key
  - Client encryption key
  - Server encryption key
  - Client initialization vector (IV)
  - Server initialization vector (IV)

- Spielt eine Rolle in den Schichten
  - Bitübertragungsschicht
  - Sicherungsschicht
  - Vermittlungsschicht
  - Transportschicht
  - Anwendungsschicht
- Was ist eine Bedrohung (oder ein Angriff)?
- Welche Methoden gibt es?
  - Kryptographie
- Wie wehrt man Angriffe ab?
  - Beispiel: Firewalls

# Was ist eine Bedrohung?

## ■ Definition:

- Eine Bedrohung eines Rechnernetzwerks ist jedes mögliche Ereignis oder eine Folge von Aktionen, die zu einer Verletzung von Sicherheitszielen führen kann
- Die Realisierung einer Bedrohung ist ein Angriff

## ■ Beispiel:

- Ein Hacker erhält Zugang zu einem geschlossenen Netzwerk
- Veröffentlichung von durchlaufenden E-Mails
- Fremder Zugriff zu einem Online-Bankkonto
- Ein Hacker bringt ein System zum Absturz
- Jemand agiert unautorisiert im Namen anderer (Identity Theft)

- Vertraulichkeit:
  - Übertragene oder gespeicherte Daten können nur vom vorbestimmten Publikum gelesen oder geschrieben werden
  - Vertraulichkeit der Identität der Teilnehmer: Anonymität
- Datenintegrität
  - Veränderungen von Daten sollten entdeckt werden
  - Der Autor von Daten sollte erkennbar sein
- Verantwortlichkeit
  - Jedem Kommunikationsereignis muss ein Verursacher zugeordnet werden können
- Verfügbarkeit
  - Dienste sollten verfügbar sein und korrekt arbeiten
- Zugriffskontrolle
  - Dienste und Informationen sollten nur autorisierten Benutzern zugänglich sein

- Maskierung (Masquerade)
  - Jemand gibt sich als ein anderer aus
- Abhören (Eavesdropping)
  - Jemand liest Informationen, die nicht für ihn bestimmt sind
- Zugriffsverletzung (Authorization Violation)
  - Jemand benutzt einen Dienst oder eine Resource, die nicht für ihn bestimmt ist
- Verlust oder Veränderung (übertragener) Information
  - Daten werden verändert oder zerstört
- Verleugnung der Kommunikation
  - Jemand behauptet (fälschlicherweise) nicht der Verursacher von Kommunikation zu sein
- Fälschen von Information
  - Jemand erzeugt (verändert) Nachrichten im Namen anderer
- Sabotage
  - Jede Aktion, die die Verfügbarkeit oder das korrekte Funktionieren der Dienste oder des Systems reduziert

# Bedrohungen und Sicherheitsziele

Sicherheits- ziele	Bedrohungen						
	Mas- kierung	Abhören	Zugriffs- ver- letzung	Verlust oder Verän- derung (über- tragener) information	Verleug- nung der Kommuni- kation	Fäl- schen von Infor- mation	Sabotage (z.B. Überlast)
Vertraulichkeit	x	x	x				
Datenintegrität	x		x	x		x	
Verantwort- lichkeit	x		x		x	x	
Verfügbarkeit	x		x	x			x
Zugriffs- kontrolle	x		x			x	

## ■ Sicherheitsdienst

- Ein abstrakter Dienst, der eine Sicherheitseigenschaft zur erreichen sucht
- Kann mit (oder ohne) Hilfe kryptografischer Algorithmen und Protokolle realisiert werden, z.B.
  - Verschlüsselung von Daten auf einer Festplatte
  - CD im Safe

## ■ Kryptografischer Algorithmus

- Mathematische Transformationen
- werden in kryptografischen Protokollen verwendet

## ■ Kryptografisches Protokoll

- Folge von Schritten und auszutauschenden Nachrichten um ein Sicherheitsziel zu erreichen



- Authentisierung
  - Digitale Unterschrift: Das Datum ist nachweislich vom Verursacher
- Integrität
  - Sichert ab, dass ein Datum nicht unbemerkt verändert wird
- Vertraulichkeit
  - Das Datum kann nur vom Empfänger verstanden werden
- Zugriffskontrolle
  - kontrolliert, dass nur Berechtigte Zugang zu Diensten und Information besitzen
- Unleugbarkeit
  - beweist, dass die Nachricht unleugbar vom Verursacher ist

# Systeme II

## 7. Sicherheit

Christian Schindelhauer

Technische Fakultät

Rechnernetze und Telematik

Albert-Ludwigs-Universität Freiburg