Rechnernetze I

SoSe 2020

Roland Wismüller
Universität Siegen
roland.wismueller@uni-siegen.de
Tel.: 0271/740-4050, Büro: H-B 8404

Stand: 8. Juli 2020

Rechnernetze I

SoSe 2020

10 Netzwerksicherheit

Inhalt

- Sicherheitsanforderungen
- Sicherheitsprobleme der Internet-Protokolle
- Kryptographische Grundlagen
- Sicherheitsmechanismen für Protokolle
- Beispiele sicherer Protokolle
- Firewalls

- Peterson, Kap. 8.1, 8.2, 8.3.1, 8.3.3, 8.4
- CCNA, Kap. 11.2

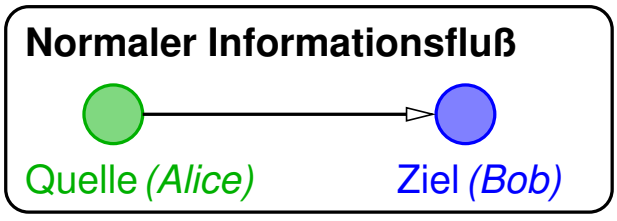
10.1 Sicherheitsanforderungen

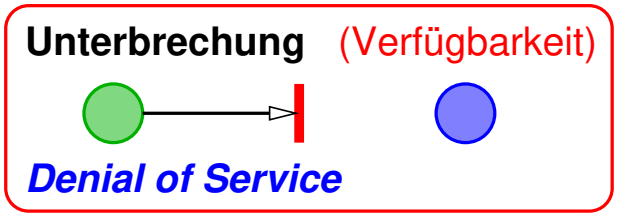
- In Netzwerken wird persönliche / wertvolle / vertrauliche Information übermittelt
  - Information sollte nur Berechtigen bekannt werden!
  - Authentizität der Information?
- Wachsende Bedeutung der Netzwerksicherheit wegen
  - steigender Vernetzung
    - höheres Angriffspotential
  - neuer Einsatzgebiete
    - z.B. e-Business: elektronische Zahlung / Verträge

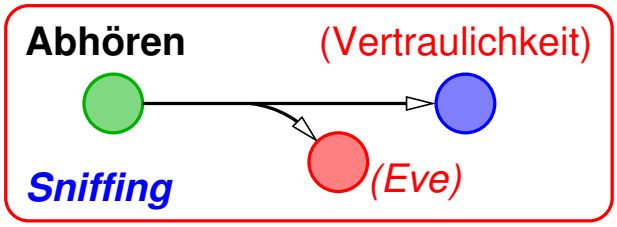
Allgemeine Sicherheitsanforderungen

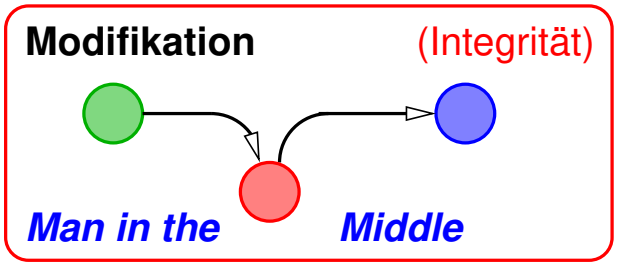
- **(Informations-)Vertraulichkeit (*confidentiality*)**
  - Schutz vor unautorisierter Informationsgewinnung
- **(Daten-)Integrität (*integrity*)**
  - Schutz vor unautorisierter Veränderung von Daten
- **(Nachrichten-)Authentizität (*message authenticity*)**
  - Urheber der Daten kann korrekt identifiziert werden
- **Verbindlichkeit (*nonrepudiation*)**
  - Handlungen können nicht abgestritten werden
- **Verfügbarkeit (*availability*)** von Diensten
- **Anonymität** der Kommunikationspartner

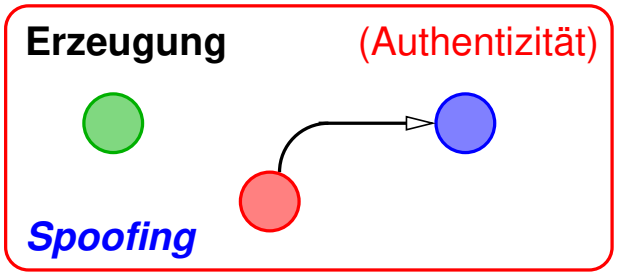
Angriffe auf die Netzwerksicherheit

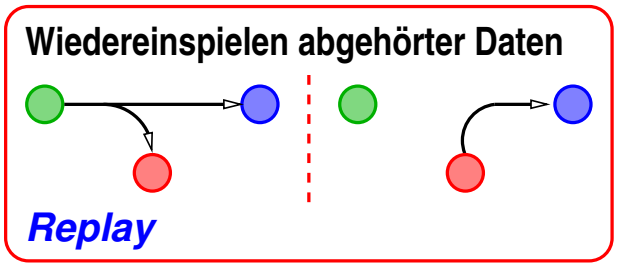












10.1 Sicherheitsanforderungen ...

Konkret: Alice sendet eine Nachricht an Bob

- **Vertraulichkeit**: niemand außer Alice und Bob erfahren den Inhalt der Nachricht
- **Integrität**: Bob kann sich (nach entsprechender Prüfung!) sicher sein, daß die Nachricht während der Übertragung nicht (absichtlich) verfälscht wurde
- **Authentizität**: Bob kann sich (nach entsprechender Prüfung!) sicher sein, daß die Nachricht von Alice gesendet wurde
- **Verbindlichkeit**: Alice kann nicht bestreiten, die Nachricht verfaßt zu haben
  D.h. Bob kann Dritten gegenüber **beweisen**, daß die Nachricht von Alice gesendet wurde
- Im Folgenden: Beschränkung auf diese vier Anforderungen

10.2 Sicherheitsprobleme des Internets

- Heutiger Stand im Internet:
  - Die Standard-Internet-Protokolle (u.a. IP, TCP, DNS, ARP, NFS, HTTP, SMTP) erfüllen **keine** der in 10.1 genannten Sicherheitsanforderungen
- Hauptprobleme:
  - öffentliche Netze prinzipiell abhörbar
  - fehlende / unzureichende Authentifizierung
- Abhilfe:
  - sichere Protokolle in der Anwendungsschicht:
    - SSL/TLS (HTTPS, FTPS), S/MIME, PGP, SSH, ...
  - sicheres IP-Protokoll (IPsec, siehe Rechnernetze II)
- Basis: kryptographische Verfahren

Anmerkungen zu Folie 268:

Ein Grundproblem der Internet-(Un)sicherheit ist die Tatsache, daß der Absenderadresse von IP-Paketen nicht vertraut werden kann (der Sender kann die Quell-IP-Adresse Aufwand beliebig angeben, dazu ist praktisch kein Aufwand notwendig).

Daher ist es wichtig, in Protokollen nicht auf die Authentizität der Absenderadresse zu vertrauen.

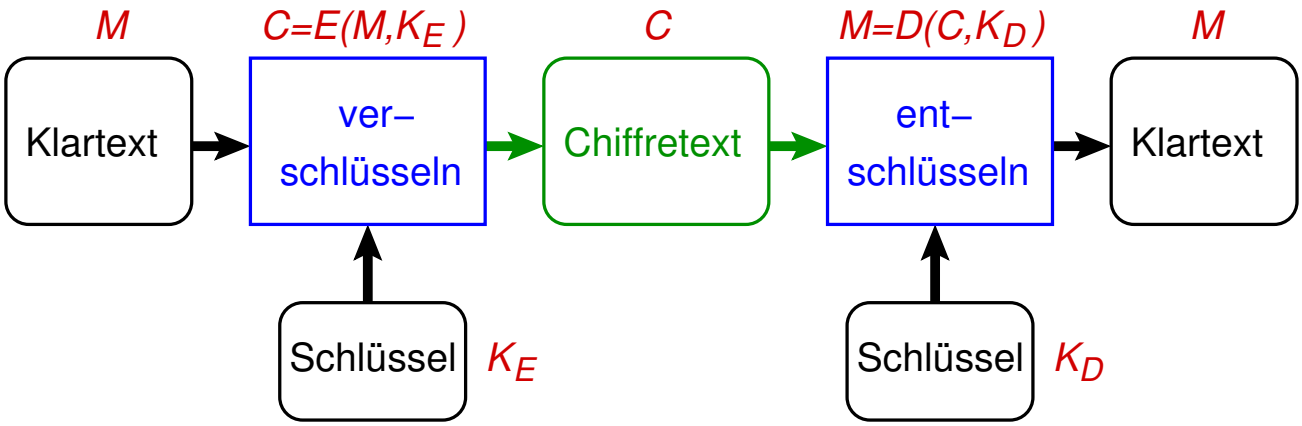
Eine weitere Maßnahme, um die Sicherheit zu erhöhen ist die sog. *Source Address Verification* in Routern. Dabei prüfen die Router, ob ein IP-Paket, das sie über eine bestimmte Schnittstelle empfangen, tatsächlich aus diesem Netz kommen kann. Empfängt der Router beispielsweise über eine Schnittstelle, an die ein LAN mit der Adresse 141.99.0.0/16 angeschlossen ist, ein Paket mit Quelladresse 131.159.79.3, so weiß er, daß dieses eine falsche Quelladresse besitzt und verwirft das Paket.

Eine wirkliche Authentifizierung von IP-Paketen ist mit *Secure IP* (IPsec) möglich.

Meist wird heute die Sicherheit aber nicht auf Ebene von IP realisiert, sondern in den darüberliegenden Protokollen.

10.3 Kryptographische Grundlagen

Grundprinzip der Verschlüsselung:



- Symmetrische Verschlüsselungsverfahren
  - $K_E = K_D = K$ = gemeinsamer geheimer Schlüssel
- Asymmetrische Verschlüsselungsverfahren
  - K_E = öffentlicher, K_D = privater Schlüssel

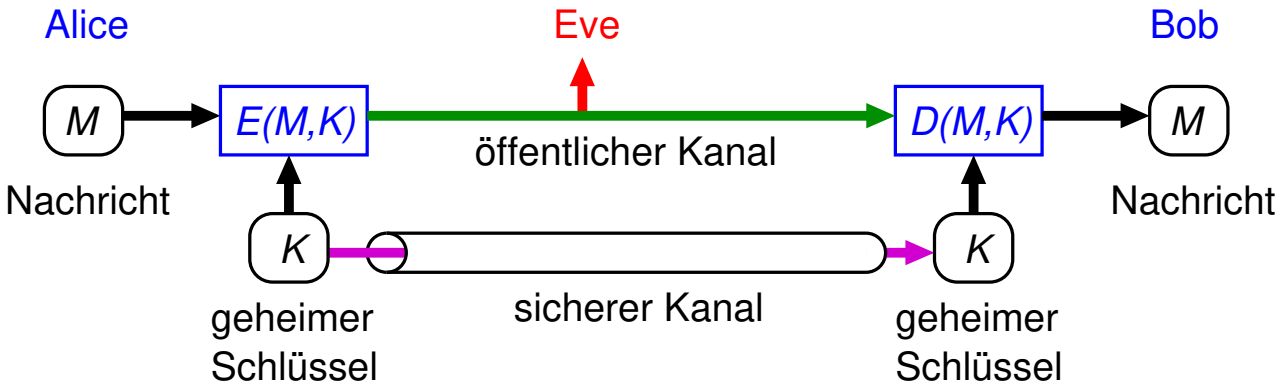
Anforderungen an Verschlüsselungsverfahren:

- Nur der Besitzer des geheimen bzw. privaten Schlüssels kann den Chiffretext entschlüsseln
- Sicherheit basiert nicht auf Geheimhaltung der Algorithmen

Mögliche Angriffe:

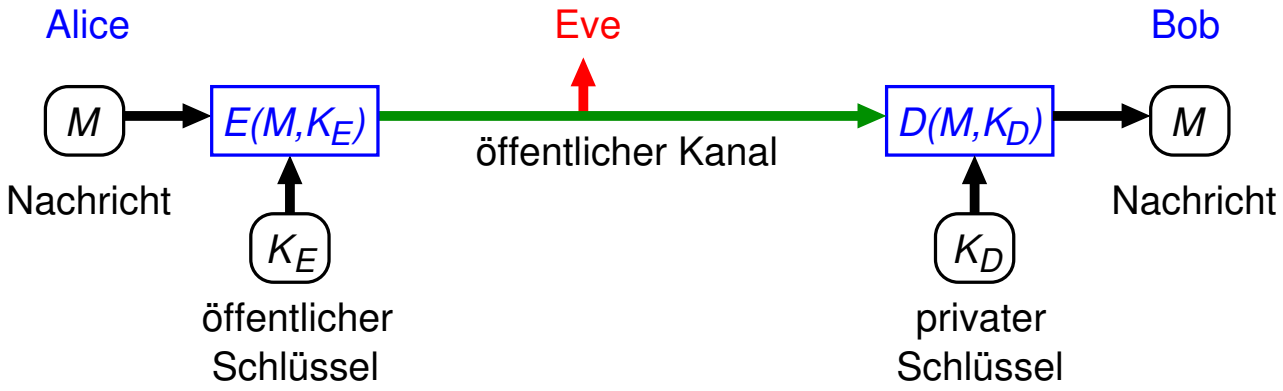
- Klartext-Angriff: Klartext + Chiffretext $\Rightarrow$ Schlüssel
- Im Idealfall: alle Schlüssel müssen durchprobiert werden
  - Schlüssel müssen lang genug sein!
- Bei asymmetrischen Verfahren auch effizientere Angriffe
  - Berechnung von K_D aus K_E ($\Rightarrow$ längere Schlüssel nötig)

Symmetrische Verschlüsselung:



- Symmetrische Verschlüsselung ist sehr effizient realisierbar
- Schlüssel sind relativ kurz (heute typisch 128-256 Bit)
- Problem: Austausch des Schlüssels K

Asymmetrische Verschlüsselung:



- Bob berechnet K_E aus K_D und veröffentlicht K_E
  - Problem: Authentizität von K_E
- Weniger effizient als symmetrische Verfahren
- Längere Schlüssel nötig (heute typisch 2048-4096 Bit)

Beispiele symmetrischer Verschlüsselungsverfahren:

- **DES**: veraltet, Schlüssel nur 56 Bit lang
- **Triple-DES**: veraltet, dreifache Anwendung von DES
- **AES**: Nachfolger von DES, 128-256 Bit Schlüssel
- **IDEA**: z.B. in PGP benutzt, 128 Bit Schlüssel

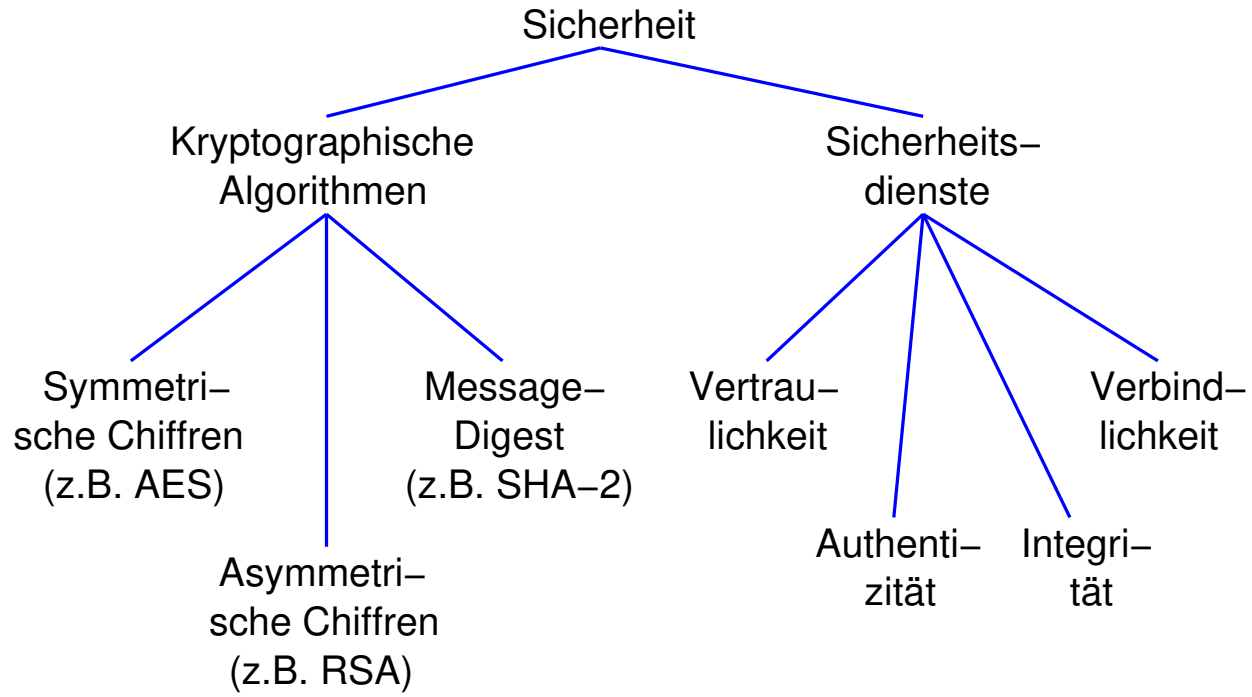
Beispiele für asymmetrische Verschlüsselungsverfahren:

- **RSA**: Standardverfahren, Sicherheit basiert auf der Schwierigkeit der Primfaktorzerlegung großer Zahlen
- Diffie-Hellman-Schlüsselaustausch: basiert auf diskretem Logarithmus
- *Elliptic Curve Cryptography*: erlaubt deutlich kürzere Schlüssel

Kryptographische Hashes (*Message Digest*)

- Analog einer normalen Hashfunktion:
  - Nachricht wird auf einen Wert fester Größe abgebildet
- Zusätzliche Eigenschaft: **Kollisionsresistenz**
  - zu Nachricht *x* kann (in vernünftiger Zeit) keine andere Nachricht *y* mit gleichem Hashwert gefunden werden
- Einsatz zur Sicherung der Integrität
  - „kryptographische Prüfsumme“
- Beispiele
  - MD5 (*Message Digest, Version 5*): 128 Bit Hashwert, unsicher
  - SHA-1 (*Secure Hash Algorithm 1*): 160 Bit Hashwert, unsicher
  - SHA-2 / SHA-3: 224 - 512 Bit Hashwert

Begriffe



Was leistet die reine Verschlüsselung von Nachrichten?

- Vertraulichkeit: **ja**
- Integrität: **bedingt**
  - nur, wenn Klartext genügend Redundanz aufweist
  - $\Rightarrow$ Verwendung von *Message Digests*
- Nachrichtenauthentizität:
  - **nein** bei asymmetrischen Verfahren: K_E öffentlich!
  - **bedingt** bei symmetrischer Verschlüsselung
    - nur mit gesicherter Integrität und Schutz vor *Replay*
- Verbindlichkeit: **nein**
- Schutz vor Replay: **nein**
  - $\Rightarrow$ Transaktionszähler im Klartext + Integrität sichern

10.4 Sicherheitsmechanismen

- Kryptographische Algorithmen sind nur Bausteine für die Netzwerksicherheit
- Zusätzlich benötigt: Mechanismen und Protokolle
- Einige Sicherheitsaufgaben:
  - Authentifizierung
    - von Kommunikationspartnern
      - „wer ist mein Gegenüber?“
    - von Nachrichten
      - „stammt die Nachricht wirklich vom Absender?“
  - Sicherung der Integrität von Nachrichten
  - Verbindlichkeit
  - Verteilung öffentlicher Schlüssel

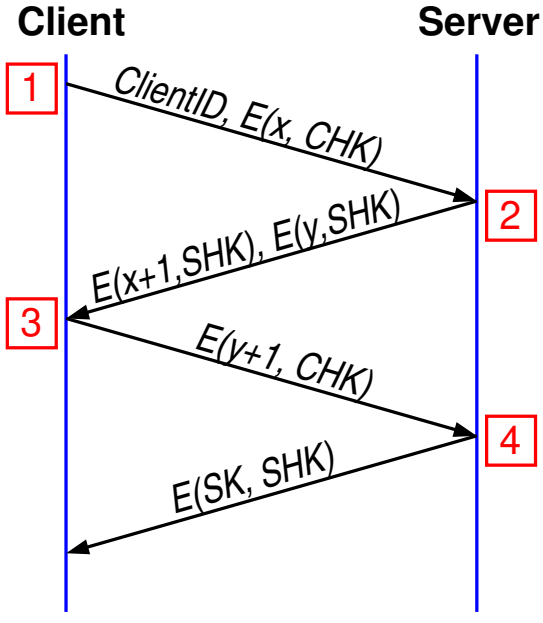
Partner-Authentifizierung

- Kommunikationspartner identifizieren sich gegenseitig
  - Beispiel: File-Server
    - Server verlangt ID des Clients zur Prüfung der Schreib-/Leserechte
    - Client verlangt ID des Servers zum Lesen/Schreiben sensibler Daten
- Manchmal auch nur einseitige Authentifizierung
  - Beispiel: WWW-Server
    - Client verlangt ID des Servers zur Übertragung wichtiger / vertraulicher Daten

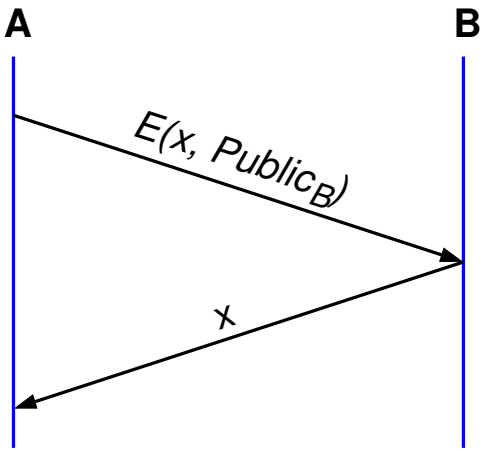
Partner-Authentifizierung mit Drei-Wege-Handshake

- Server kennt Schlüssel des Clients (z.B. Paßwort bei login)

1. Client sendet *ClientID* und verschlüsselte Zufallszahl x (CHK: *Client Handshake Key*)
2. Server sucht den zu *ClientID* gehörigen Schlüssel *SHK*, sendet *x+1* und Zufallszahl *y*
3. Server ist authentifiziert (*x+1*)
4. Client ist authentifiziert (*y+1*), Server sendet *Session Key SK* für weitere Kommunikation



Partner-Authentifizierung über asymmetrische Chiffre



- Einseitige Authentifizierung von *B*
  - ggf. authentifiziert sich *A* ebenso (≈ 3-Wege-Handshake)
- $Public_B$ nicht zum Verschlüsseln verwenden!

Sicherung der Nachrichtenintegrität und -authentizität

- Integrität: Kein Dritter soll Nachricht verfälschen können
  - setzt sinnvollerweise Nachrichten-Authentizität voraus
- Bei Übertragung mit symmetrischer Verschlüsselung:
  - kryptographischen Hashwert *H(M)* an Klartext *M* anfügen und verschlüsseln
  - bei Modifikation des Chiffretexts paßt die Nachricht nicht mehr zum Hashwert
  - kein Angreifer kann neuen Hashwert berechnen / verschlüsseln
  - Nachrichten-Authentizität (bis auf Replay) durch symmetrische Chiffre sichergestellt
    - Replay-Schutz: Transaktionszähler / Zeitstempel in *M*

Sicherung der Nachrichtenintegrität und -authentizität ...

- Bei asymmetrischer Verschlüsselung:
  - Hash-Wert allein nützt nichts, da Nachrichten-Authentizität nicht sichergestellt ist
- Bei unverschlüsselter Übertragung (oft sind Daten nicht vertraulich, aber ihre Integrität wichtig):
  - Hash-Wert stellt Integrität nicht sicher, da jeder nach einer Modifikation der Nachricht den neuen Hash-Wert berechnen kann
- Lösungen:
  - kryptographischer Hashwert mit geheimem Schlüssel
  - digitale Signatur

Hashwert mit geheimem Schlüssel

- Einbeziehen eines (gemeinsamen) geheimen Schlüssels *K* in den Hashwert:
  - füge *H(M + K)* an Nachricht M an (*+* = Konkatenation)
- Sichert auch Nachrichten-Authentizität (bis auf Replay)
  - kein Dritter kann *H(M + K)* korrekt berechnen
  - Replay-Schutz: Transaktionszähler / Zeitstempel in *M*
- Sichert nicht Verbindlichkeit
  - Empfänger kann *H(M + K)* berechnen
- Beispiel: *HMAC-SHA-256*

Anmerkungen zu Folie 283:

Tatsächlich wird der Schlüssel nicht einfach an die Nachricht angefügt, sondern komplexer mit dieser verknüpft, um Kryptoanalyse zu erschweren:

$$HMAC(M, K) = H(K \oplus \texttt{opad} + H(K \oplus \texttt{ipad} + M))$$

ipad = $0011\ 0110\ ...\ 0011\ 0110_2$
opad = $0101\ 1010\ ...\ 0101\ 1010_2$

Dabei ist $\oplus$ die bitweise Exklusiv-Oder-Verknüpfung und $+$ die Konkatenation.

10.4 Sicherheitsmechanismen ...

Digitale Signatur mit asymmetrischer Chiffre

- Sender *A* sendet *M* und *E(M, Private$_A$)* an Empfänger *B*
- *B* entschlüsselt mit *Public$_A$* und prüft, ob Ergebnis gleich *M* ist
- Problem: asymmetrische Verschlüsselung ist langsam
- Daher: Kombination mit kryptographisher Hashfunktion
  - digitale Signatur von *A* auf *M* dann: *E(H(M), Private$_A$)*
- Digitale Signatur sichert Integrität, Nachrichten-Authentizität (bis auf Replay) und Verbindlichkeit
  - nur *A* besitzt *Private$_A$*
  - Replay-Schutz: Transaktionszähler in *M*

Anmerkungen zu Folie 284:

Bei der Signatur mit Hilfe einer kryptographishen Hashfunktion ist die exakte Vorgehensweise wie folgt:

- Alice sendet *M* und *S* := *E(H(M), Private$_A$)* an Bob
- Bob entschlüsselt *S* und erhält einen „Soll"-Hashwert *H'* := *D(S, Public$_A$)*
- Bob berechnet nun selbst den Hashwert über die Nachricht *M*:
  H := *H(M)*
- Nun vergleicht Bob *H'* mit *H*. Sind beide gleich, akzeptiert er die Signatur.

Dieses Signaturschema ist nur sicher, wenn die verwendete Hashfunktion kollisionsresistent ist. Sonst könnte ein Angreifer zu dem Hashwert
H' := *D(S, Public$_A$)* einer Signatur *S* eine neue Nachricht *M'* berechnen, die dieselbe Signatur hat, und somit behaupten, Alice hätte *M'* signiert.

10.4 Sicherheitsmechanismen ...

Verteilung öffentlicher Schlüssel

- Problem: Übertragung des öffentlichen Schlüssels *Public$_A$* von *A* zu *B*
- Woher weiß *B*, daß *Public$_A$* authentisch ist?
  - zur Authentifizierung bräuchte *B* den Schlüssel von *A* ...
- Lösungen:
  - Übertragung über andere Medien (persönlich, Post, ...)
  - Zertifikate (*Certificates*)

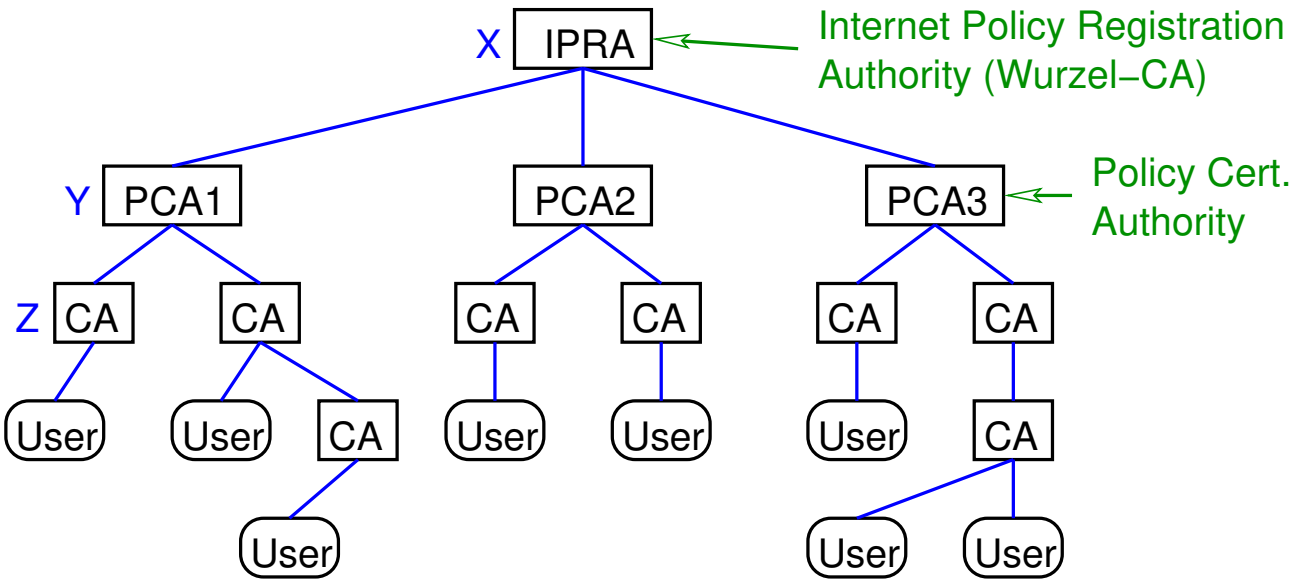
Zertifikat

> *Ich bestätige, daß der in diesem Dokument stehende öffentliche Schlüssel dem angegebenen Eigentümer gehört.*
>
> Gezeichnet: $\mathcal{CA}$

- Die Zertifizierungsstelle (CA, *Certification Authority*) beglaubigt die Zuordnung zwischen einem öffentlichem Schlüssel und seinem Besitzer
  - durch digitale Signatur
- Nur noch der öffentliche Schlüssel der CA muß separat veröffentlicht werden

Zertifizierungshierarchie (z.B. bei HTTPS)

- Vertrauenskette: *X* zertifiziert, daß Schlüssel von *Y* authentisch ist, *Y* zertifiziert Schlüssel von *Z*, ...



X.509 Zertifikate

- X.509: wichtiger Standard für Zertifikate
- Komponenten des Zertifikats:
  - Name der Person/Institution oder eines Rechners
    - ggf. auch Email-Adresse oder Domain-Name
  - öffentlicher Schlüssel der Person/Institution bzw. des Rechners
  - Name der CA
  - Ablaufdatum des Zertifikats (optional)
  - digitale Signatur der CA
    - über alle obigen Felder

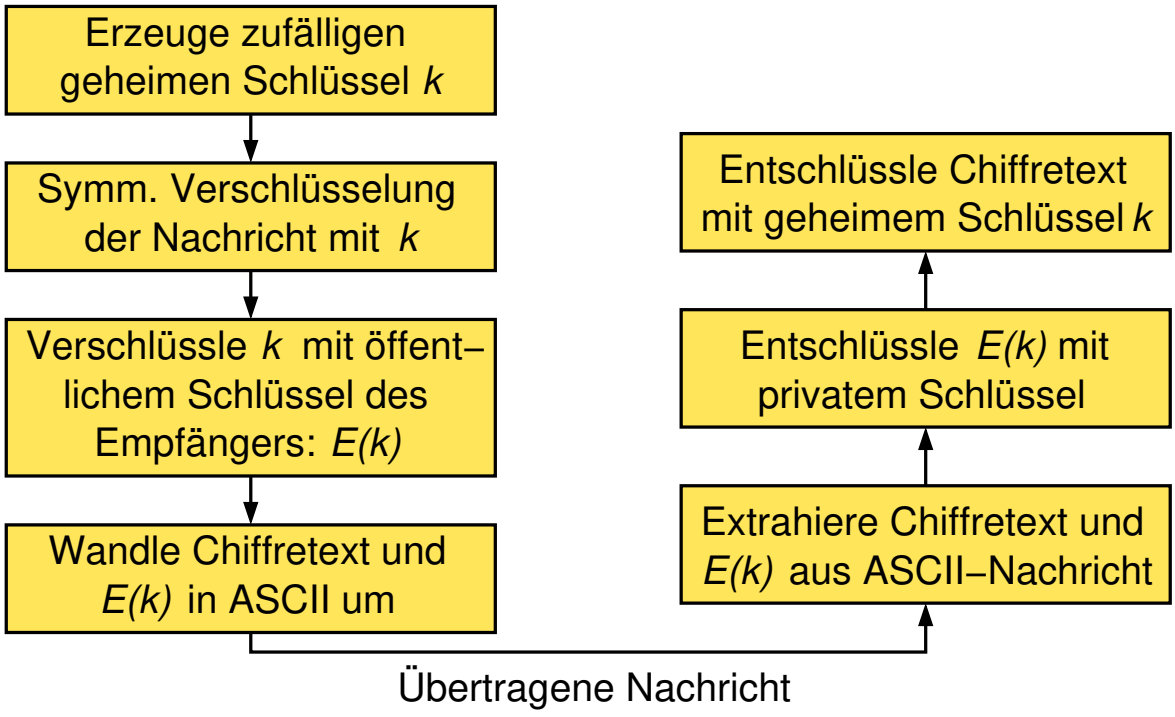
Invalidierung von Zertifikaten

- Zertifikate können beliebig kopiert und verbreitet werden
- Identität wird durch ein Zertifikat nur in Verbindung mit dem Besitz des privaten Schlüssels belegt
- Falls privater Schlüssel ausgespäht wurde:
  - Widerruf des Zertifikats nötig
- Einfache Möglichkeit:
  - *Certificate Revocation List* (CRL)
    Liste widerrufener Zertifikate, signiert von CA
  - Ablaufdatum begrenzt Länge der Liste

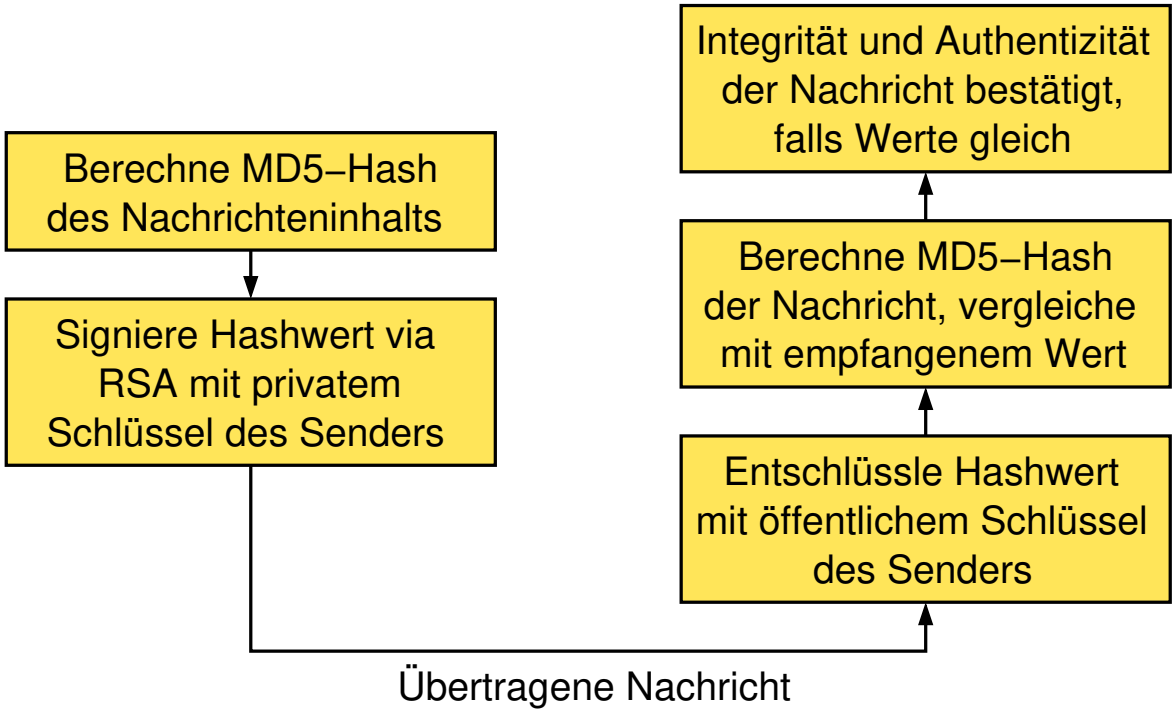
PGP (Pretty Good Privacy)

- Realisiert Vertraulichkeit, Integrität, Authentifizierung und Verbindlichkeit für Email
- Mechanismen: Verschlüsselung und digitale Signatur
  - einzeln oder kombiniert verwendbar
- Keine Zertifizierungsstellen bzw. –hierarchie
  - PGP-Benutzer zertifizieren die öffentlichen Schlüssel gegenseitig
    - mehrere Zertifikate möglich (höheres Vertrauen)
  - Vertrauensstufe des Schlüssels wird bei Email-Empfang angezeigt

PGP: Verschlüsselte Übertragung von Emails



PGP: Signierung von Emails



TLS (*Transport Layer Security*)

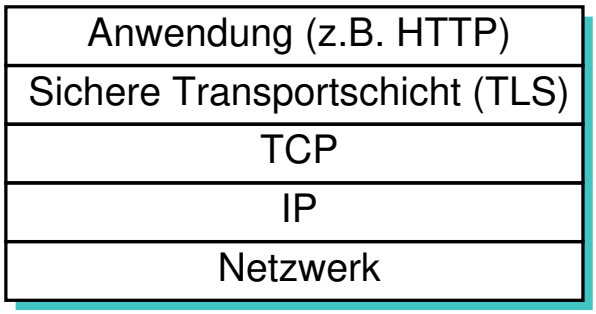
- Motivation: Sicherheit im WWW, z.B. für Kreditkartenzahlung
  - Vertraulichkeit (der Kreditkarteninformation)
  - Authentizität (des WWW-Servers)
  - Integrität (der Bestelldaten)
  - (Verbindlichkeit wird von TLS nicht gewährleistet)
- TLS ist ein Internet-Standard der IETF
  - Basis: ältere Realisierung SSL (*Secure Socket Layer*)
- TLS ist die Grundlage vieler sicherer Protokolle im WWW:
  - z.B. HTTPS, FTPS, ...
  - realisiert durch eine zusätzliche Schicht

Anmerkungen zu Folie 293:

SFTP ist eine weitere sichere Variante des Dateitransfer-Protokolls FTP, das nicht TLS, sondern das SSH-Protokoll als Basis für die Sicherheit verwendet.

10.5 Beispiele sicherer Protokolle ...

TLS: sichere Transportschicht



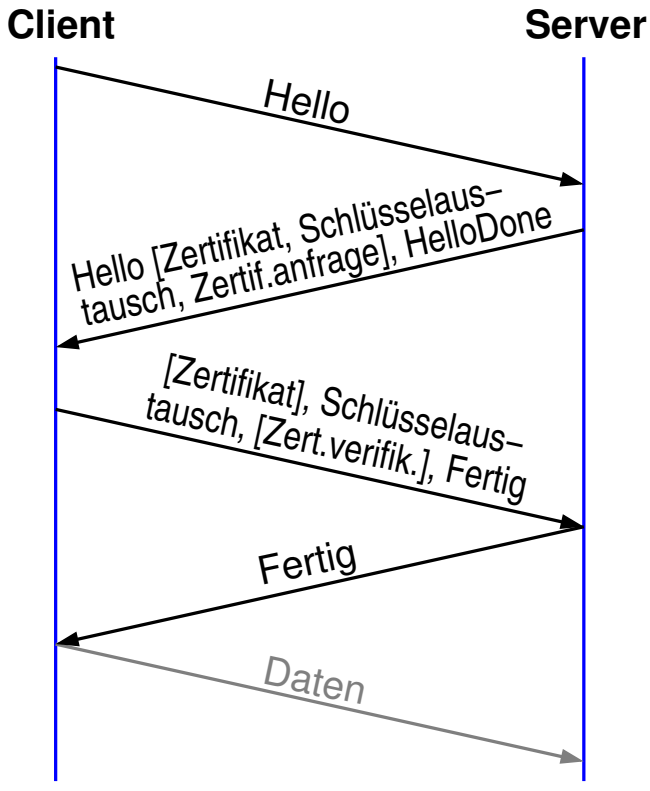
- Vorteil: unveränderte Anwendungsprotokolle
- Spezielle Ports, z.B. 443 für HTTPS
  - TLS gibt Daten von TCP an HTTP-Protokoll weiter (bzw. umgekehrt)

Wichtige TLS Teil-Protokolle:

- Handshake-Protokoll
  - beim Verbindungsaufbau
  - Aushandeln der kryptographischen Parameter:
    - Verfahren, Schlüssellänge, Sitzungsschlüssel, Zertifikate, Kompression
- Record-Protokoll
  - für die eigentlichen Daten
  - Fragmentierung, Kompression, Message Digests, Verschlüsselung, Transport (TCP)

TLS Handshake-Protokoll

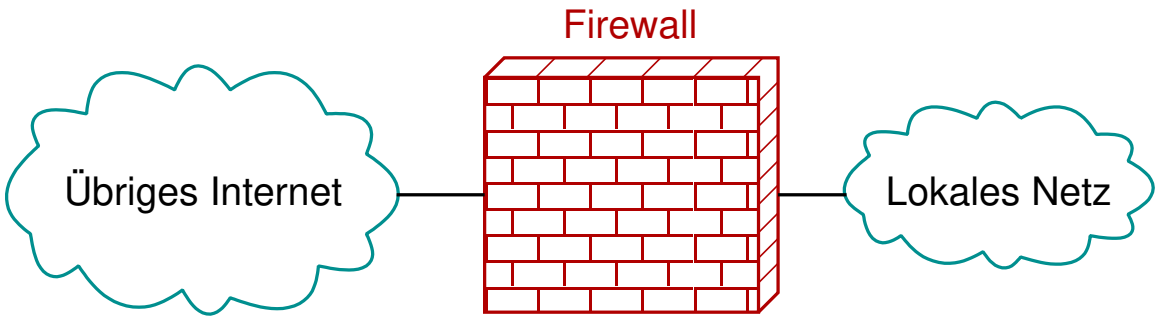
- Bis zu 12 Nachrichten
- Aushandeln der kryptographischen Parameter notwendigerweise unverschlüsselt
- Man-in-the-Middle kann schwache Verschlüsselung aushandeln
- Anwendungen müssen auf Mindestanforderungen bestehen, ggf. Verbindungsabbruch



Anmerkungen zu Folie 296:

- Die Hello-Nachrichten dienen zum Aushandeln von Parametern:
  - Version, Zufallszahl, SessionID, Verfahren für Schlüsselaustausch, Verschlüsselung und Kompression, ...
- Für den Schlüsselaustausch können unterschiedliche Verfahren verwendet werden, die dann auch Anzahl und Inhalt der Nachrichten bestimmen.
- Ein Client-Zertifikat wird nur auf Anfrage durch Server übertragen
- Die Zertifikatsverifikations-Nachricht enthält die digitale Signatur aller bisher gesendeten und empfangenen Nachrichten
- Der Server verifiziert sein Zertifikat beim Schlüsselaustausch durch eine Signatur über die Zufallszahlen aus den Hello-Nachrichten und die Server-Schlüsselaustausch-Parameter

10.6 Firewalls



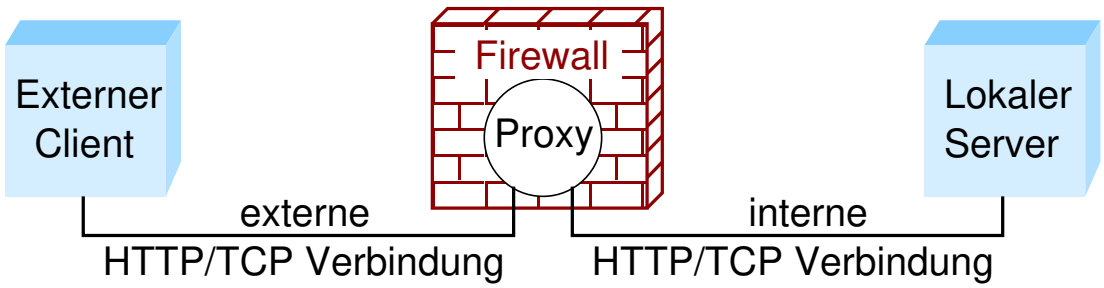
- **Firewall**: Router mit Filterfunktion
  - kann bestimmte Pakete ausfiltern (verwerfen) und somit Zugriff auf bestimmte Hosts / Dienste unterbinden
  - wäre i.W. überflüssig, wenn alle Dienste sicher wären!
- Zwei Typen:
  - Filter-basierte Firewalls
  - Proxy-basierte Firewalls

Filter-basierte Firewalls

- Filtern nur aufgrund von Quell- und Ziel-IP-Adressen, Quell- und Ziel-Ports, sowie übertragenem Protokoll
- Filterregeln z.B.
  - `deny tcp 192.12.0.0/16 host 128.7.6.5 eq 80`
  - `permit tcp any host 128.7.6.5 eq 25`
- Frage: alles erlaubt, was nicht verboten ist, oder umgekehrt?
- Statische oder dynamische Regeln
  - z.B. FTP: neue Ports für jede übertragene Datei
- „Level-4-Switch“: Firewall kennt Transport-Protokolle

Proxy-basierte Firewalls

- Proxy: Mittler zwischen Client und Server
  - für Client: Proxy ist Server, für Server: Proxy ist Client



- Proxy arbeitet auf Anwendungsschicht
  - kann auf der Basis des Nachrichteninhalts filtern
  - z.B. HTTP-Anfragen nach bestimmten Seiten nur von speziellen Hosts akzeptieren

Grenzen von Firewalls

- Kein Schutz interner Benutzer untereinander
- Nur begrenzter Schutz gegen mobilen Code (z.B. Email Wurm)
- Schutz von Teilen eines Netzes schwierig
- Angreifer kann sich in privilegiertes Netz „einschleichen“
  - z.B. bei drahtlosen Netzen
- Filterung über Sender-IP-Adresse/Port ist unsicher

Vorteil von Firewalls

- Umsetzung einer Sicherheitsstrategie an zentraler Stelle

10.7 Zusammenfassung

- Sicherheitsanforderungen:
  - Vertraulichkeit, Integrität, Authentizität,Verbindlichkeit
  - Verfügbarkeit, Anonymität, ...
- IP, TCP, UDP erfüllen keine Sicherheitsanforderungen
  - Vertraulichkeit, Integrität, Authentizität
- Kryptographische Verfahren:
  - symmetrische und asymmetrische Chiffren
  - Kryptographische Hashes (Message Digest)
- Sicherheitsmechanismen
  - Authentifizierung (Kommunikationspartner, Nachrichten)
  - Integrität: Hashwerte mit Schlüssel, digitale Signatur
  - Verteilung öffentlicher Schlüssel: Zertifikate

- Sichere Protokolle, z.B. PGP, TLS (HTTPS), IPsec
- Firewalls

Fortsetzung:

- Rechnernetze-Praktikum (WiSe, B.Sc., Vertiefungsprakt., 5 LP)
  - Aufbau von Netzen, Routing und Switching
- Rechnernetze II (SoSe, M.Sc./B.Sc., 5 LP)
  - weitere Netzwerktechnologien (Fast Ethernet, WLAN, ...)
  - Vertiefung (Routing, QoS, IPsec, ...)