Informatik II

Dr. Henrik Brosenne
Georg-August-Universität Göttingen
Institut für Informatik

Sommersemester 2015

Inhalt

Literatur

J. F. Kurose, K. W. Ross.
Computernetze: Ein Top-Down-Ansatz mit Schwerpunkt Internet,
Pearson Studium (Addison-Wesley), August 2008.

A. S. Tanenbaum.
Computernetzwerke,
Pearson Studium (Prentice-Hall), 2003.

William Stallings.
Data & Computer Communications,
Prentice Hall, 2007.
`http://williamstallings.com/DCC6e.html`

Warriors of the net movie.
`http://www.warriorsofthe.net/movie.html`

Begriffe

Telematik = **TELE**kommunikation und Infor**MATIK**, geprägt von Simon Nora und Alain Minc in Ihrem Bericht an den französischen Präsidenten, 1978.

Drückt die Verflechtung von Rechnern und Telekommunikationsmitteln aus.

Andere Bezeichnungen für Telematik.

- Computernetze
- Computernetzwerke
- Rechnernetze
- Kommunikationsnetze

Aufgabengebiete

- Praktische, Angewandte und Technische Informatik.
- Technische Infrastruktur verteilter Systeme (Netze).
- Netz-Dienste und darauf aufbauende Anwendungen.
- Regeln für Nachrichtenaustausch (Protokolle).
- Werkzeuge zum Entwickeln verteilter Anwendungen.

Was ist das *Internet* (1/3)

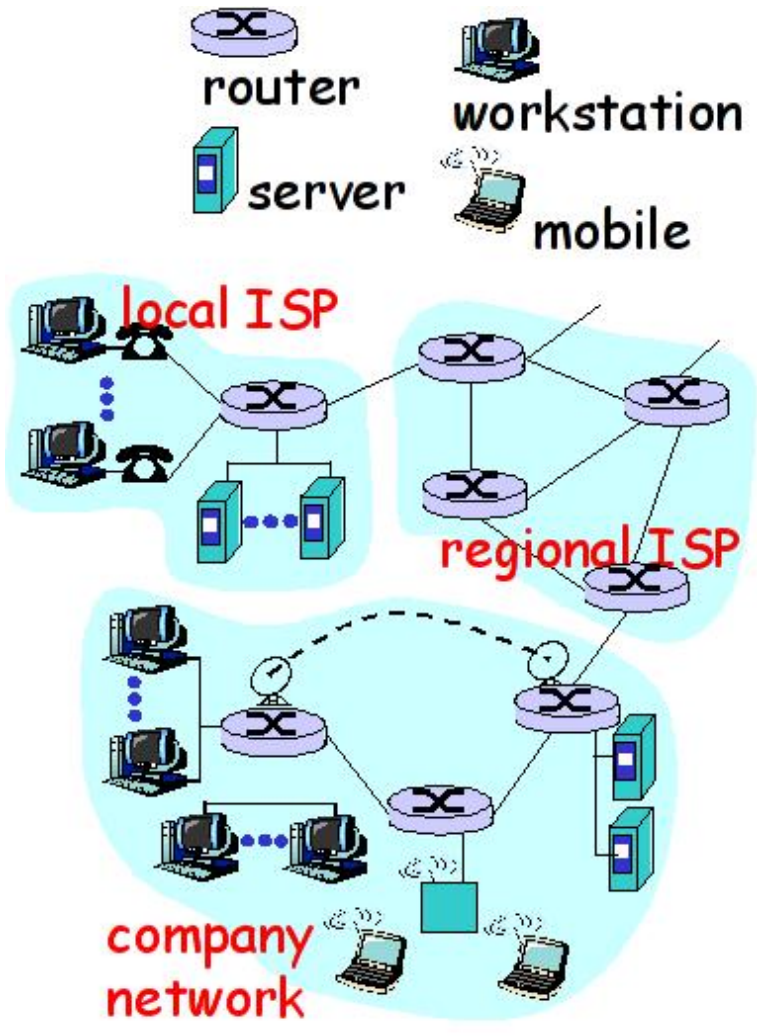
Sehr viele (Millionen) von verbundenen Rechnern **End-Geräte** (*hosts*) sind Geräte auf denen **Netzwerkanwendung** laufen.

- PCs, Workstations, Server.
- PDAs, Telephone, Toaster

Kommunikationsverbindungen

- Kabel (Kupfer, Fiberglas, etc.), Funk (Hochfrequenz, Satellit, etc.).
- Übertragunsrate = Bandbreite (*bandwidth*).

Vermittlungseinheiten (*switching nodes*) schalten die Kommunikationsverbindungen zusammen.



Quelle: Kurose & Ross *Computernetze: Ein Top-Down-Ansatz mit Schwerpunkt Internet*

Was ist das *Internet* (2/3)

Protokolle regeln Senden und Empfangen von Nachrichten.

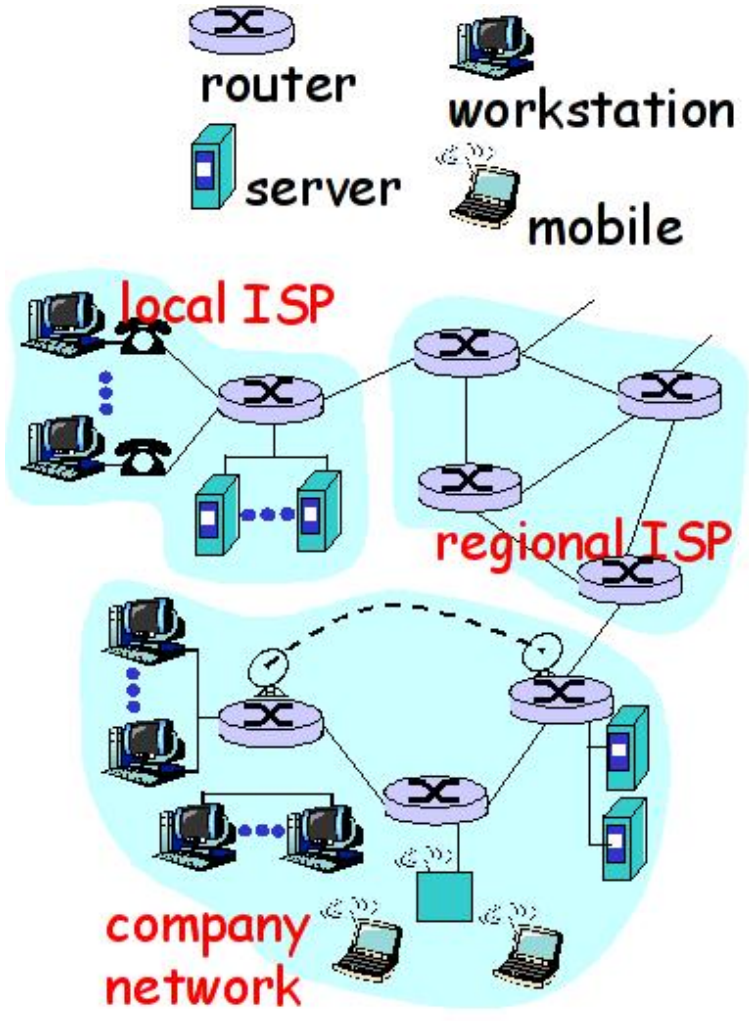
- PPP (*Point-to-Point Protocol*)
- PPPoE (*PPP over Ethernet*)
- IP (*Internet Protocol*)
- TCP (*Transmission Control Protocol*)
- HTTP (*Hypertext Transport Protocol*)
- FTP (*File Transfer Protocol*)

Internet, *Netzwerk von Netzwerken*

- Schwach hierarchisch Strukturiert.
- Öffentliches Internet steht dem privaten Intranet gegenüber.

Internet **Standards**

- IETF (*Internet Engineering Task Force*) veröffentlicht RFCs (*Request For Comments*).
- IEEE (*Institute of Electrical and Electronics Engineers*).



Was ist das *Internet* (3/3)

Kommunikationsinfrastruktur ermöglicht verteilte Anwendungen

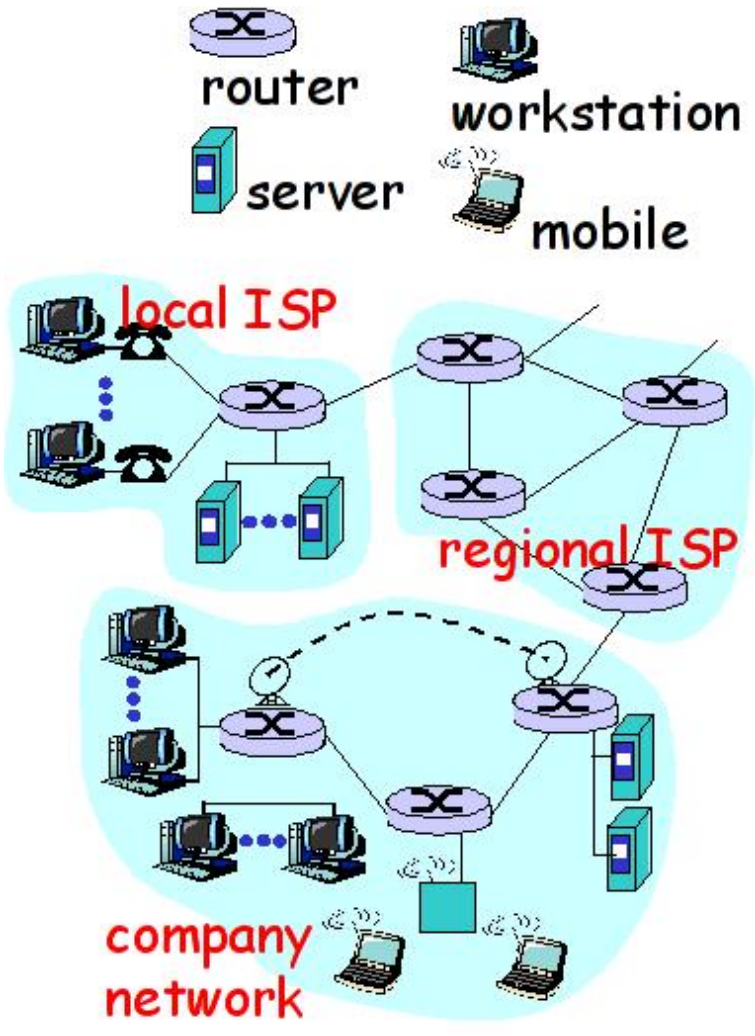
- Web, EMail, Spiele, E-Commerce, File-Sharing (z.B. MP3).

Kommunikationsdienste stehen den Netzwerkanwendungen zur Verfügung.

- Verbindungslos (*connectionless*).
- Verbindungsorientiert (*connection-oriented*).

Cyberspace von William Gibson (Neuromancer, 1984).

- *a consensual hallucination experienced daily by billions of operators, in every nation, . . .*

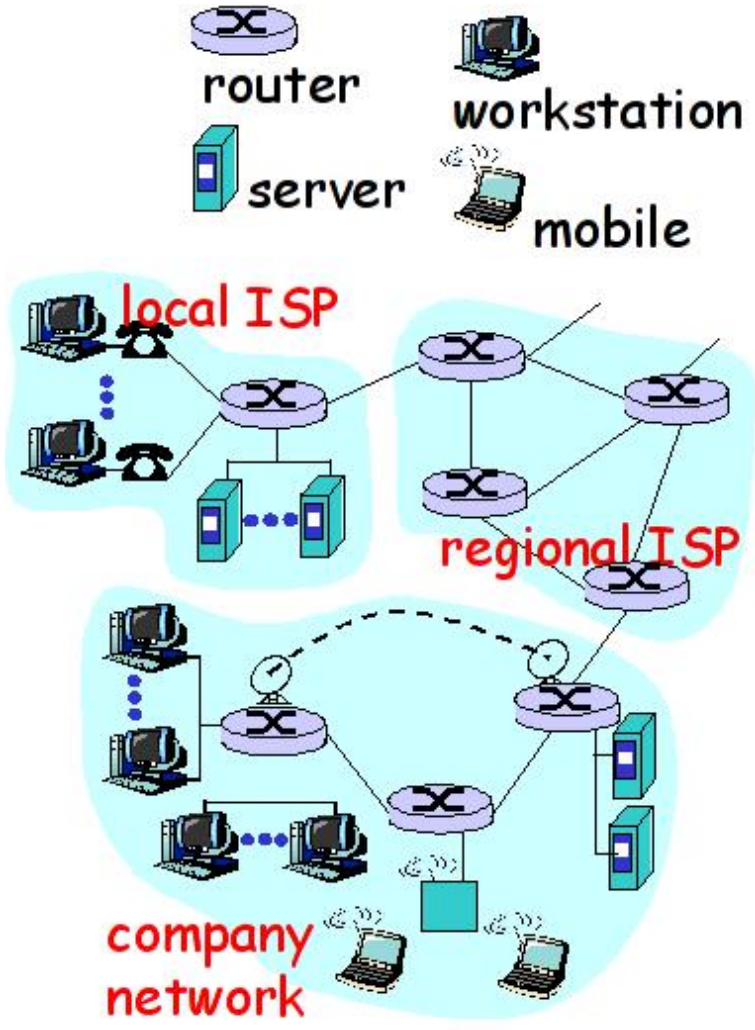


Quelle: Kurose & Ross *Computernetze: Ein Top-Down-Ansatz mit Schwerpunkt Internet*

Inhalt

Vermittlung ist der Mechanismus zum Zusammenschalten der Kommunikationsverbindungen (Übertragungskanäle).

- Leitungsvermittlung
- Paketvermittlung



Quelle: Kurose & Ross *Computernetze: Ein Top-Down-Ansatz mit Schwerpunkt Internet*

Leitungsvermittlung

Leitungsvermittlung (*circuit switching*) ist das Einrichtung einer festen und exklusiven Verbindung der Kommunikationspartner durch das Kommunikationsnetz für die Dauer der Verbindung.

Phasen einer vermittelten Verbindung.

- **Verbindungsaufbau.** Auswahl der Kommunikationspartner, Reservierung der benötigten Betriebsmittel (Kanäle, Bandbreiten, usw.), Feststellen der Kommunikationsbereitschaft der Endeinrichtung, Herstellen der Verbindung.
- **Informationsübertragung.** Transparent basierend auf den Anforderungen beim Verbindungsaufbau und den Möglichkeiten des Netzes.
- **Verbindungsabbau.** Freigabe der verwendeten Betriebsmittel.

Signalisierung. Austausch von Informationen zum Auf- und Abbau der Verbindungen.

Dienstgüte ist sichergestellt. Qualität der Sprachübermittlung, Verfügbarkeit des Netzes, Zuverlässigkeit der Übertragung, Bitrate, Laufzeit, etc.

Paketvermittlung (1/2)

Bei Datenkommunikation sind Pausen typisch, Leitungsvermittlung ist daher ineffizient.

Paketvermittlung (*packet switching*) bedeutet, dass die Daten in Datenpakete zerlegt und einzeln übertragen werden.

Dateneinheiten beinhalten Kontrollinformationen (*header*).

Statistisches Multiplexen. Pakete werden in den Vermittlungseinheiten in Puffern zwischengespeichert.

- Vor dem Weiterleiten können sich Wartezeiten ergeben.
- Puffer können überlaufen.

Best Effort. Dienstgüte ist zunächst nicht sichergestellt, hierfür sind weitere Maßnahmen notwendig.

Paketvermittlung (2/2)

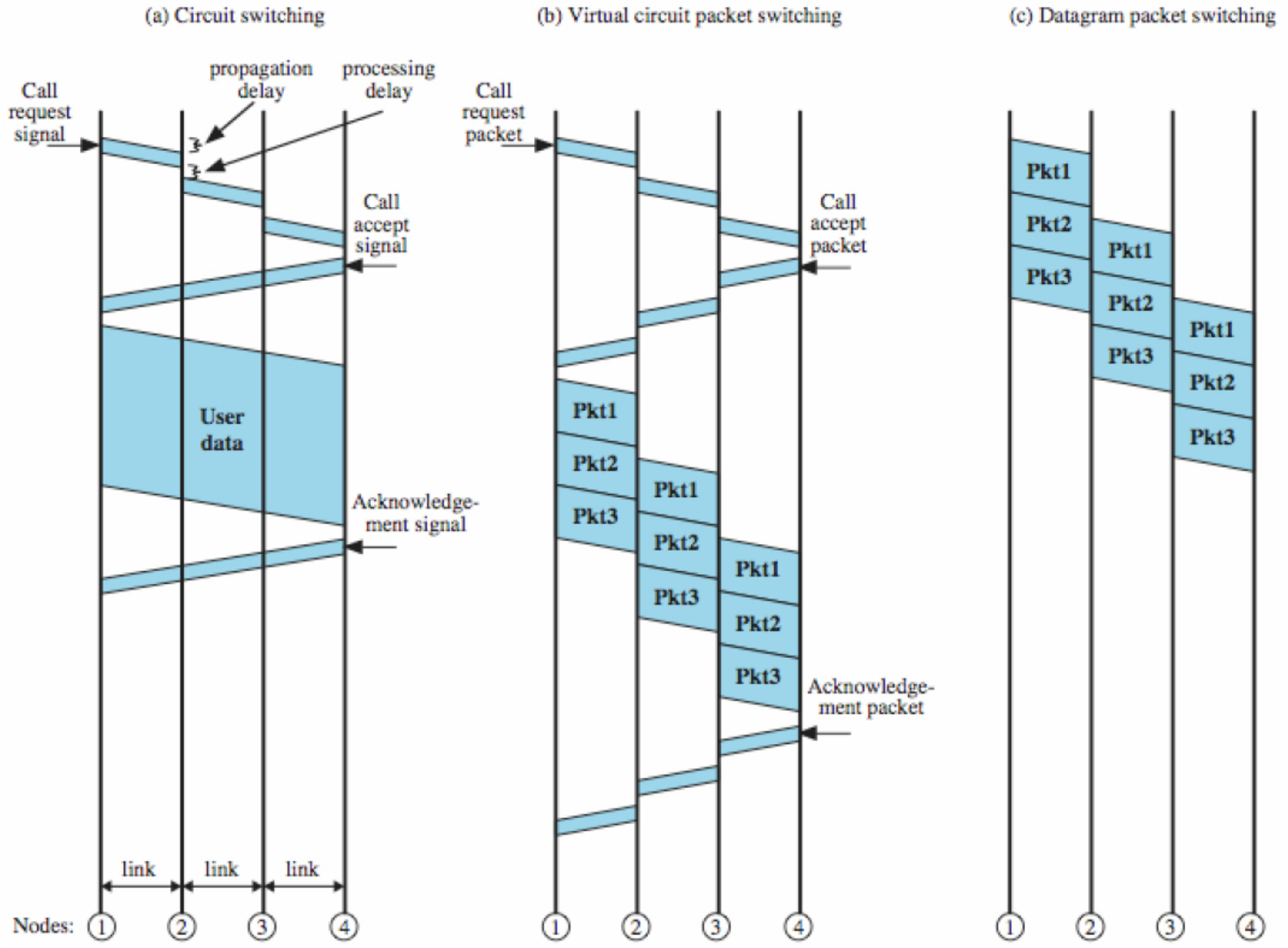
Datagramm-orientierte Paketvermittlung

- Jedes Paket enthält eindeutige Zieladresse, aufgrund der die Vermittlungseinheit es weiterleitet.
- Pakete können unterschiedliche Pfade durchlaufen (ist aber untypisch).

Virtuelle Leitungsvermittlung (*virtual circuit switching*)

- Vor dem Übertragen von Paketen wird ein Pfad bestehend aus Vermittlungseinheiten als **virtuelle Verbindung** festgelegt.
- Jedes Paket wird über diesen Pfad geleitet, aber immer noch in den Vermittlungseinheiten zwischengespeichert.

Beispiel


(a) Circuit switching
(b) Virtual circuit packet switching
(c) Datagram packet switching
Call request signal
propagation delay
processing delay
Call accept signal
User data
Acknowledge-ment signal
link
link
link
Nodes:
1
2
3
4
Call request packet
Call accept packet
Acknowledge-ment packet
Pkt1
Pkt2
Pkt3

Inhalt

Schichtenarchitektur

ISO-OSI Referenzmodell

- *International Organization for Standardization* (ISO).
- *Open Systems Interconnection* (OSI).
- Liefert gute begriffliche Abstraktionen

Dienst

Dienste (*services*) werden von Kommunikationssystem bereitgestellt, um Daten, Programme, und Dokumente auszutauschen bzw. abzurufen und Programme entfernt oder in Kooperation auszuführen.

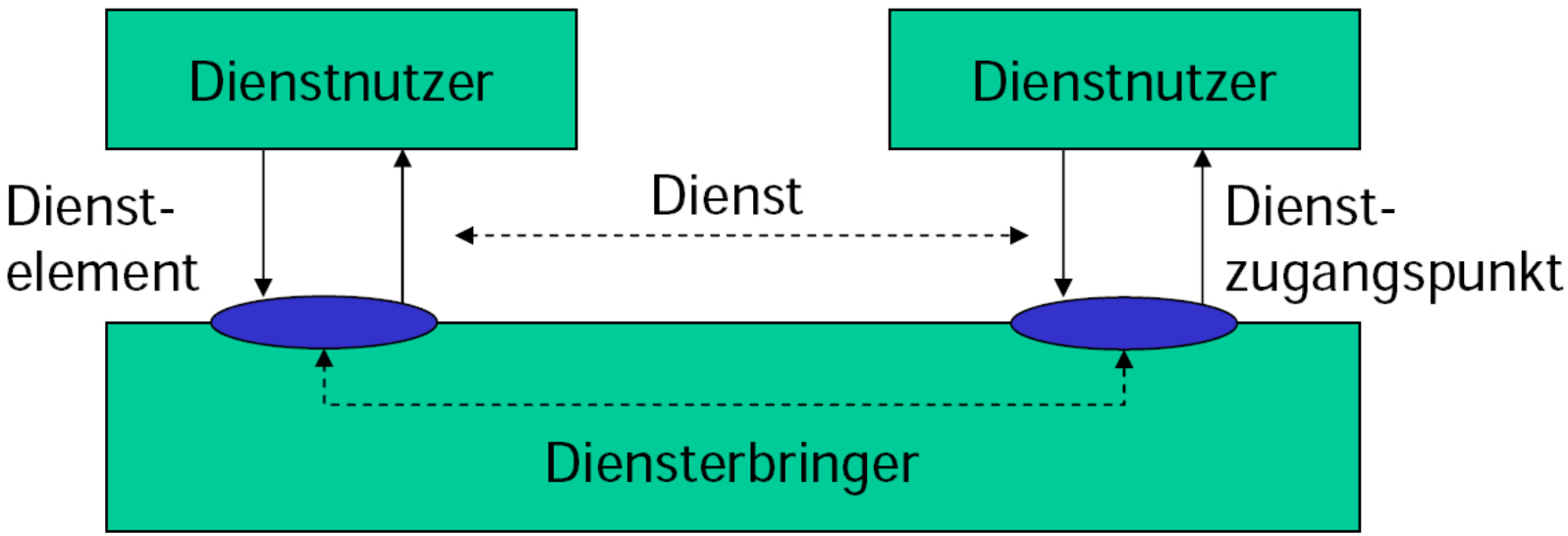
Der Dienst beschreibt **was** zu Verfügung gestellt wird.

Unterscheidung nach verschiedenen Dienstarten.

- Symmetrisch. Dienste werden an zwei oder mehreren Zugangspunkten gleichzeitig angeboten, z.B. Dienste zum Austausch von Daten.
- Asymmetrisch. Z.B. beim Client/Server-Modell.
- Bestätigt/unbestätigt.
- Verbindungsorientiert/verbindungslos.

Dienstmodell

Dienstnutzer (*service user*)
Diensterbringer (*service provider*)
Dienstschnittstelle (*service interface*)
Dienstzugangspunkt (*service access point, SAP*)
Dienstdateneinheiten (*service data units, SDU*)



Dienstelemente

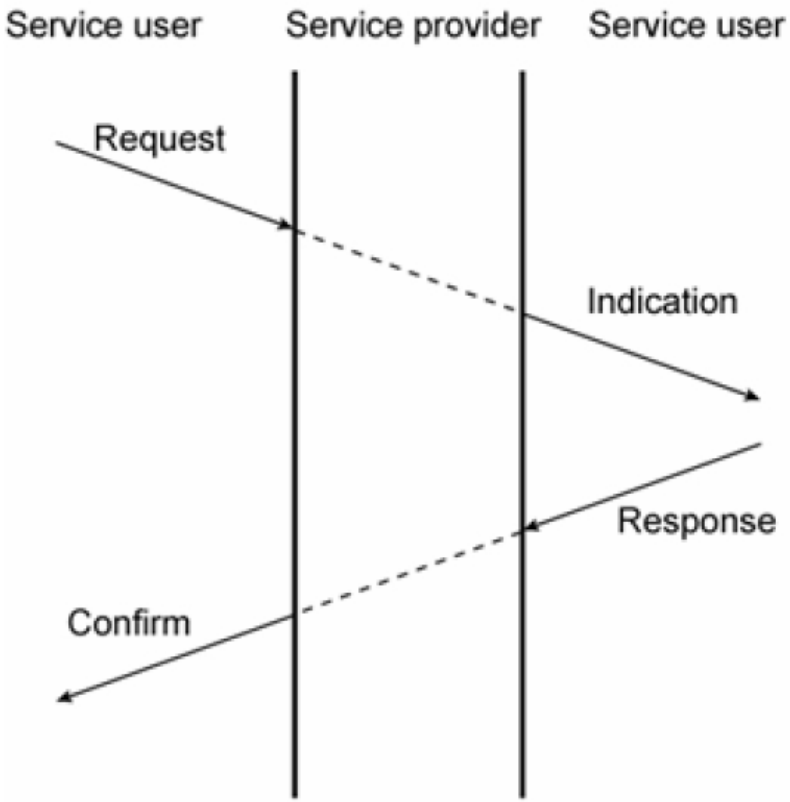
Dienstelemente (*service primitives*) sind die Abstraktionen für die Beschreibung der Wechselwirkung zwischen Dienstnutzer und Diensterbringer.

Der Dienstnutzer kennt die Dienstschnittstelle und tauscht, geregelt durch die Dienstelemente, Dienstdateneinheiten mit dem Dienstzugangspunkt des Diensterbringers aus.

Beispiel

OSI-Notation *Name, Typ, Parameter*

- `CONNECT request(CalleeAddr, CallerAddr, QoSParam, UserData)`
- `DATA indication(UserData)`
- Namen. `CONNECT, DISCONNECT, DATA, ABORT.`
- Typen. `request, indication, respond, confirm.`



Protokoll

Instanzen (*entities*). Aktive Objekte des Diensterbringers, die mit ihrer Umgebung durch den Austausch von Nachrichten interagieren.

Protokoll. Verhaltenskonvention, die die zeitliche Abfolge der Interaktionen zwischen den diensterbringenden Instanzen vorschreibt und die Formate (Syntax und Semantik) der austauschenden Nachrichten definiert.

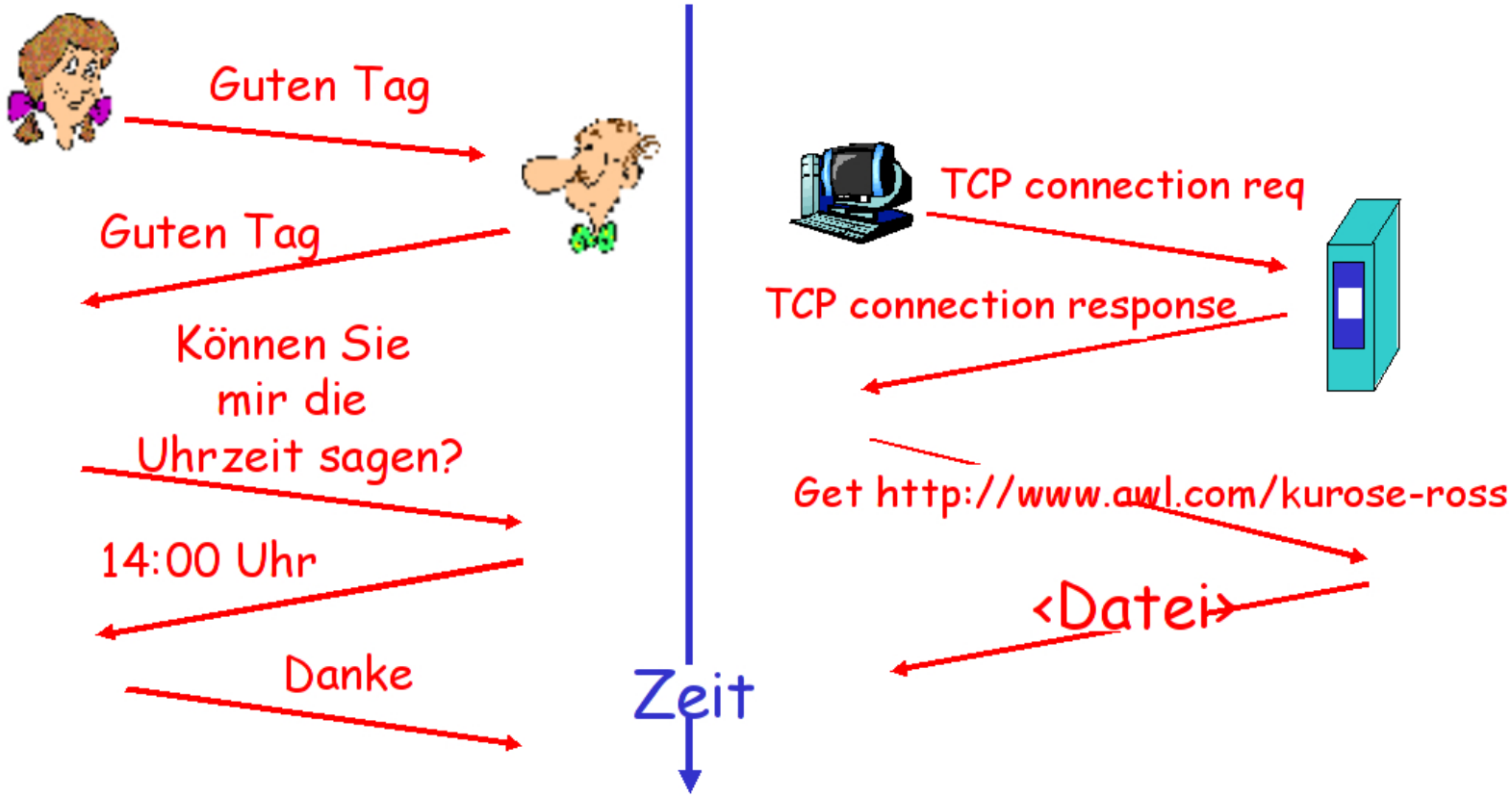
Ein Protokoll beschriebt **wie** ein Dienst zur Verfügung gestellt wird.

Ein Protokoll legt Folgendes fest.

- Beschreibung des bereitgestellten **Dienstes**.
- **Vokabular** und **Kodierungsformat** der Nachrichten.
- **Bedeutung** der Nachrichtenfelder.
- **Verhalten** der Instanzen beim Eintreffen von Nachrichten (funktional und zeitlich).
- Annahmen über die **Umgebung**, in der das Protokoll abläuft.

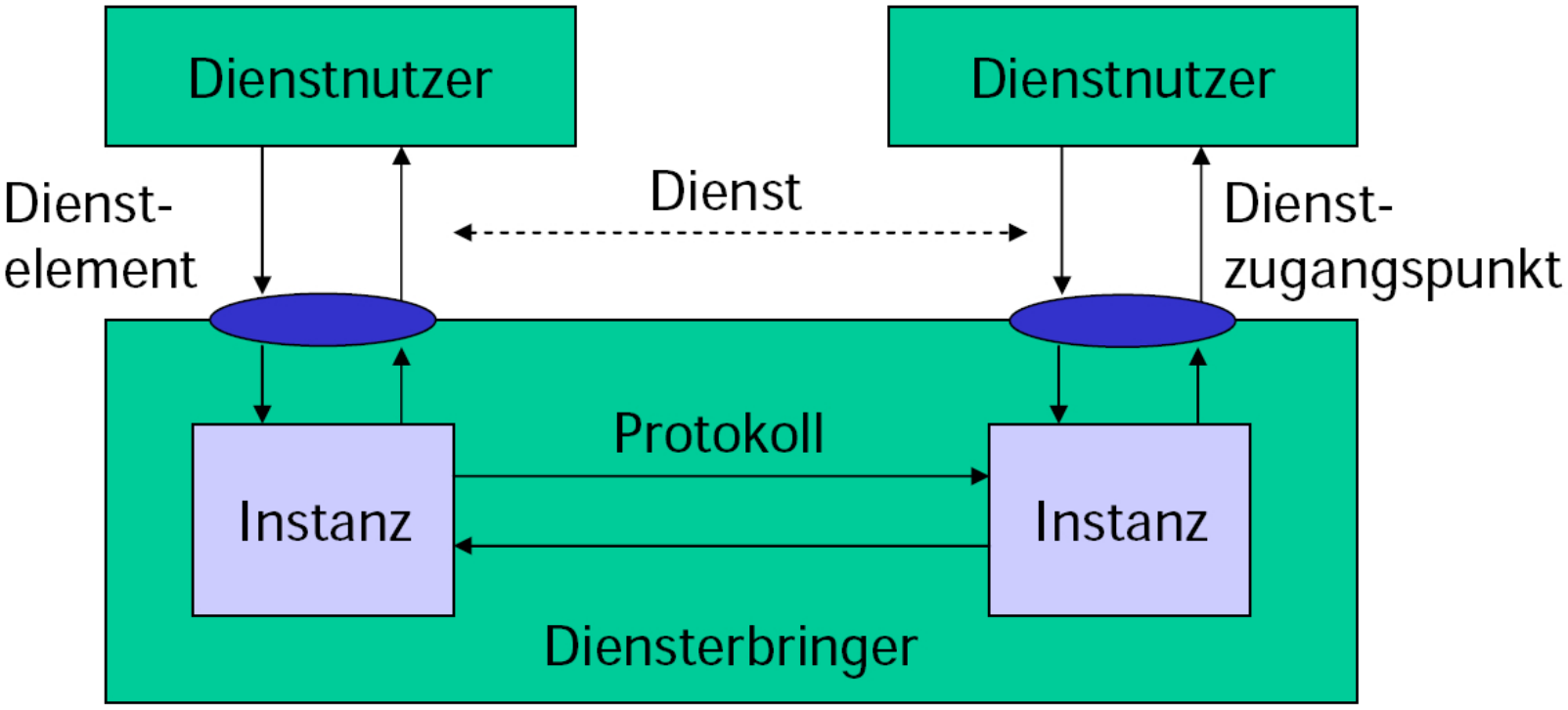
Beispiel

Menschliches und Computernetzwerk Protokoll.



Quelle: Kurose & Ross *Computernetze: Ein Top-Down-Ansatz mit Schwerpunkt Internet*

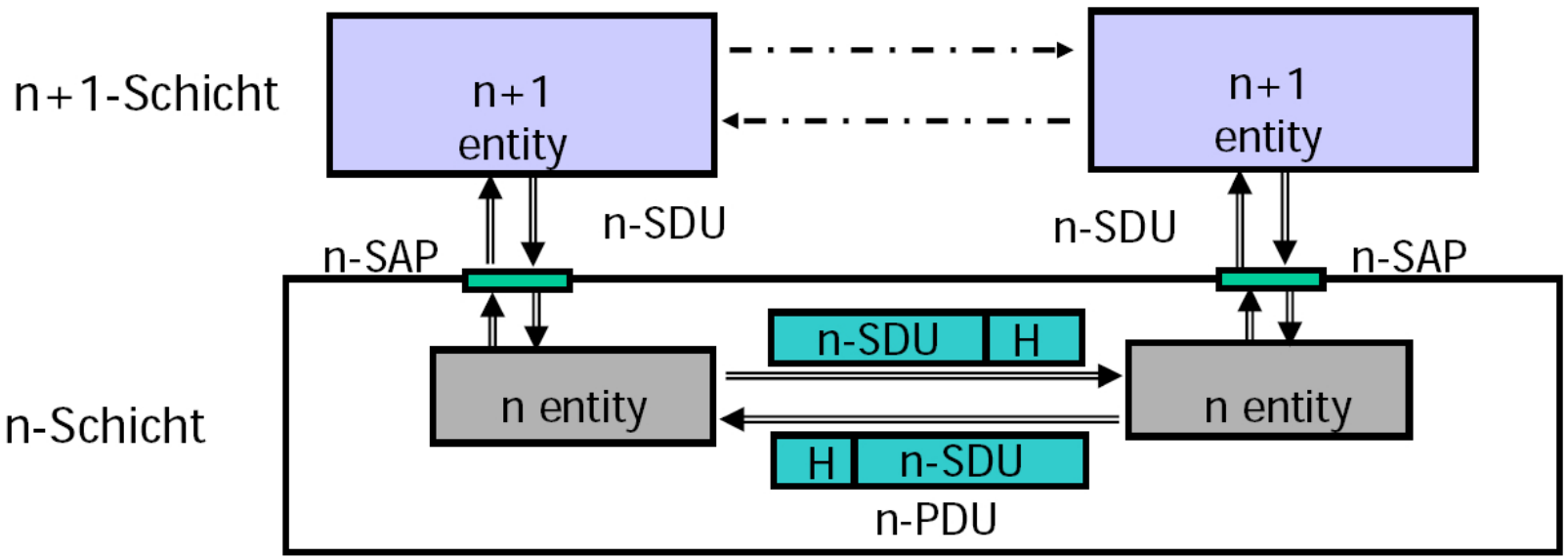
Protokoll im Dienstmodell



Schicht

Schicht (*layer*)

- Umfasst alle Instanzen, die die gleichen Dienste bereitstellen.
- Für die Kommunikation der Instanzen untereinander (d.h. zur Protokollausführung) nutzen die Instanzen die Dienste der darunter liegenden Schicht.
- Eine Instanz ist zugleich Diensterbringer und Dienstnutzer.
- Das Schichtenprinzip ist ein wichtiges Gestaltungsprinzip.
- **Partner-Instanzen** (*peer entities*). Instanzen der gleichen Schicht mit Kommunikationsbeziehung.



Schichtenarchitektur

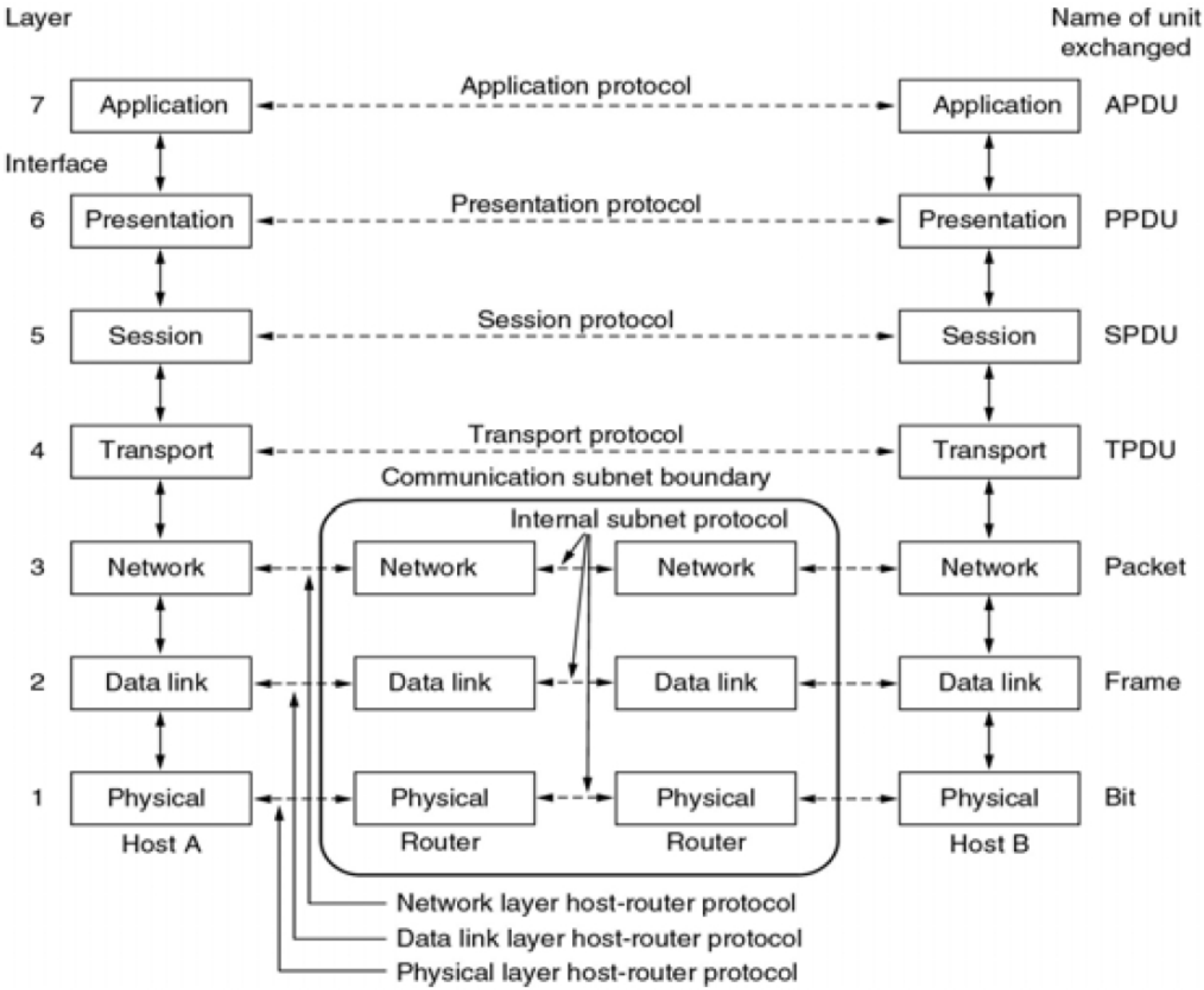
Schichtenarchitektur

- Abgestimmte Architektur von Schichten, in die die einzelnen Kommunikationsprotokolle eingebettet werden.
- Definiert Funktionalität der einzelnen Schichten und legt die Prinzipien der Interaktion zwischen ihnen fest.
- In der Regel durch Standardisierungsgremien oder Firmenkonsortien festgelegt.

ISO-OSI Referenzmodell

- Aufteilung in 7 Schichten.
- Gedankenmodell, Implementierungen haben sich nicht durchgesetzt.

ISO-OSI Referenzmodell



Bitübertragungsschicht

Bitübertragungsschicht (*Physical Layer*, *Layer 1*).

Mechanische, elektrische und prozedurale Eigenschaften zur Übertragung von **Bits**.

- Zeitsynchronisation
- Kodierung
- Modulation
- etc.

Verbindungsschicht

Verbindungs- oder **Sicherungssicht** (*Data Link Layer, Layer 2*)

Gesicherte (weitgehend fehlerfreie) Übertragung von **Rahmen** (*frames*) durch Folgenummern, Prüfsummen, Quittungs- und Wiederholungsmechanismen.

- Rahmensynchronisation
- Fehlerkontolle
- Flußkontolle
- etc.

Vermittlungs- und Transsportschicht

Vermittlungsschicht (*Network Layer*, *Layer 3*)

Übertragung von Paketen bzw. Datagrammen.

- Verbindungsaufbau
- Wegewahl
- Vermittlung
- Betriebsmittelverwaltung
- etc.

Transportschicht (*Transport Layer*, *Layer 4*)

Ende-zu-Ende Transport von Segmenten.

Festlegen der virtuelle Verbindung, wenn virtuelle Leitungsverbindung angeboten wird.

Sitzungs-, Darstellungs- und Anwendungsschicht

Sitzungsschicht (*Session Layer, Layer 5*)

Kommunikation zwischen Anwendungen.

Darstellungsschicht (*Presentation Layer, Layer 6*)

Syntax und Semantik der ausgetauschten Informationen, z.B. mit Abstract Syntax Number One (ASN.1) oder XML.

Anwendungsschicht (*Application Layer, Layer 7*)

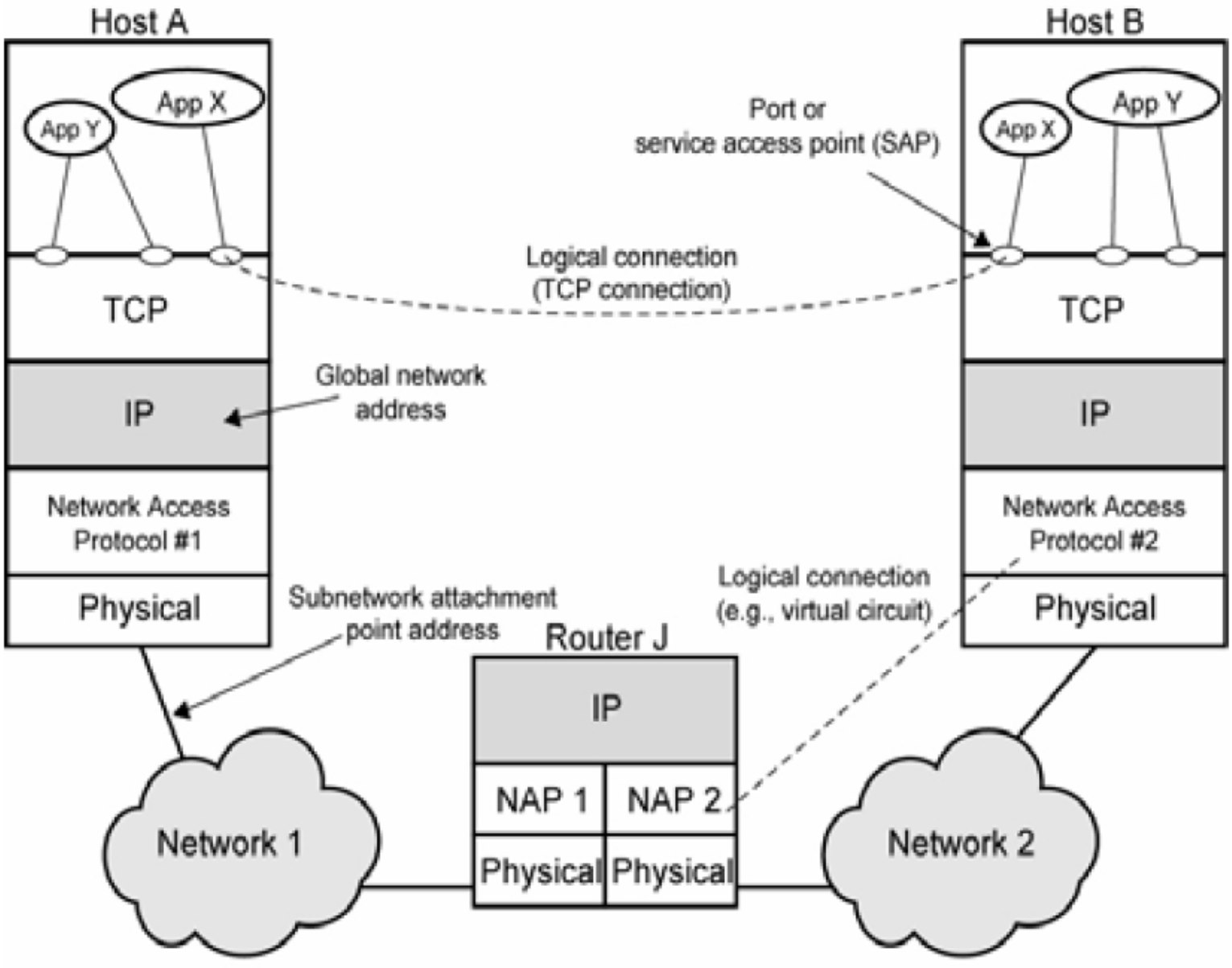
Kommunikation der Anwendungsprozesse mit anwendungsspezifischen Informationen.

ISO-OSI Modell

7	Anwendungsschicht	*Application Layer*
6	Darstellungsschicht	*Presentation Layer*
5	Sitzungsschicht	*Session Layer*
4	Transportschicht	*Transport Layer*
3	Vermittlungsschicht	*Network Layer*
2	Verbindungsschicht	*Data Link Layer*
1	Bitübertragungsschicht	*Physical Layer*

TCP-IP Modell

Im Internet hat sich eine vereinfachte Schichtenarchitektur durchgesetzt.



Transportschicht

TCP (*Transmission Control Protocol*)

- zuverlässig
- verbindungsorientiert
- paketvermittelt

UDP (*User Datagram Protocol*)

- nicht zuverlässig
- verbindungslos
- paketvermittelt

Vermittlungsschicht

IP (*Internet Protocol*)

- netzübergreifende Adressierung
- nicht zuverlässig
- verbindungslos
- paketvermittelt

ARP (*Address Resolution Protocol*)

- Zuordnung von Netzwerkadressen zu Hardwareadressen

Verbindungsschicht und Bitübertragungsschicht

Ethernet (IEEE 802.3)

- paketvermittelt
- Hardwareadressen
- Protokoll CSMA/CD *Carrier Sense Multiple Access with Collision Detect* (Verbindungsschicht)
- Pakete = Rahmen (Frames) (Verbindungsschicht)
- beschreibt die Signalisierung (physikalische Sicht = Bitübertragungsschicht)

Inhalt

Ethernet

Idee. *Funksystem* für Kabel

- Kommunikationsteilnehmer teilen ein Kabel.
- Jeder Teilnehmer kann alles mithören.

Namensgebung an Äther (*ether*) angelehnt.

Eindeutige Kennung der Teilnehmer.

Basisband-Übertragung. Gesamtes Frequenzspektrum des Mediums wird für eine Übertragung genutzt.

Zeit-Multiplex-Verfahren. In bestimmten Zeitabschnitten (Zeitschlitzen) werden die Frames verschiedener Sender auf einem Kanal übertragen.

Kollisionen können auftreten.

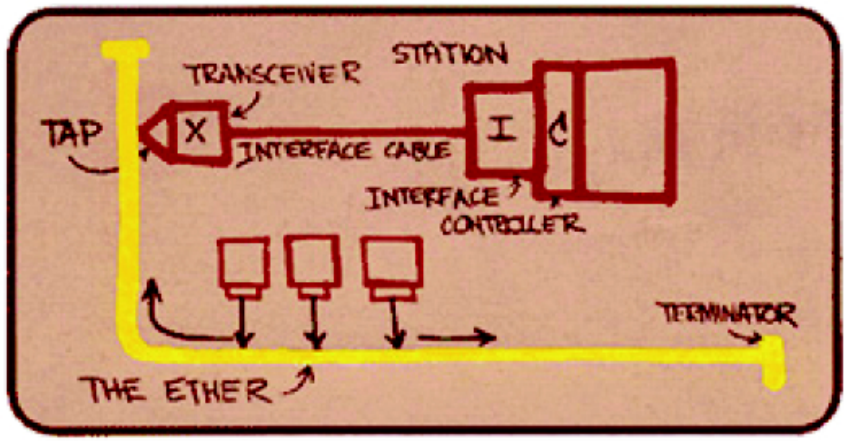


Foto © Metcalfe

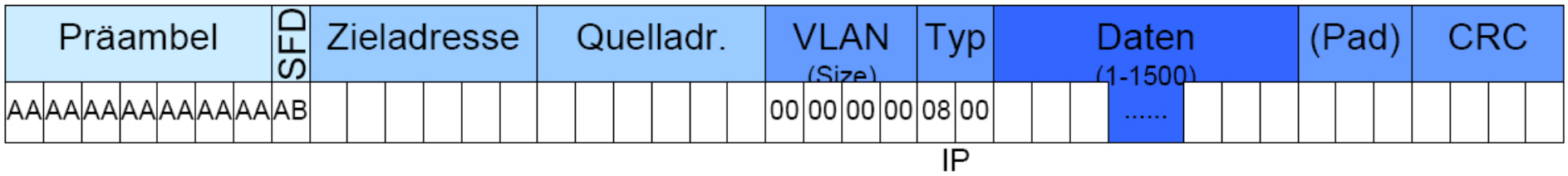
Ethernet-Frame

Pakete/Frames allgemein

- kleine Datenmenge (*payload*)
- plus Zusatzinformation (*Header*)

Ethernet-Frame

- Mindestlänge von 64 Byte
- Präambel 56 Bit. Abwechselnd 0 und 1 zur Synchronisation
- SFD *Start of Frame Delimiter*
- Ziel- und Quell-Adresse.
  - 48-Bit MAC-Adresse (*Media Access Control-*)
  - jede Netzwerkkarte hat eindeutige MAC-Adresse (Hersteller-Prefix)
- Längen oder VLAN-Feld
- Paket-Typ (z.B. 0x0800 = IP)
- PAD (padding). Wenn nötig Auffüllen auf Mindestlänge.
- Prüfsumme CRC (*Cyclic Redundancy Check*)



Netzwerkkarte

Network Interface Card (NIC)

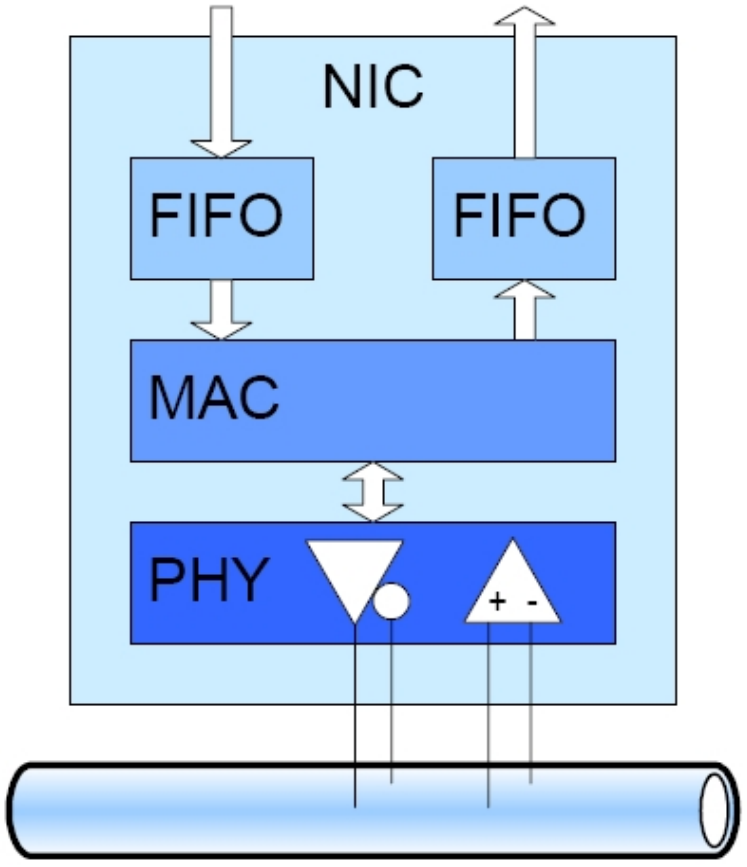
- Schnelle serielle Schnittstelle (10-1000 Mbit/s).
- FIFO-Buffer für Ein- und Ausgabe.
- Paket-orientierter Datentransfer, DMA für Datentransfer von und ins Memory.

Media-Access Controller

- Steuert Zugriff auf Physical Layer.
- Framing des Pakets.
- Prüfsumme für Daten.

Physical Layer

- Elektrischer Standard für Übertragung.



Routing

Next-Hop Routing

- Ethernet ist ein Protokoll der Verbindungsschicht (*data link layer*) und der Bitübertragungsschicht (*physical layer*).
- Frames werden nur von einem Interface zum nächsten geleitet.

Ethernet kann keine Pakete routen, das passiert auf anderer Netzwerke-Ebene.

Ziel-(Ethernet)-Adresse wird z.B. durch das ARP (*Address Resolution Protocol*) ausfindig gemacht.

ARP

Zum Ausfindigmachen der MAC-Adresse

- Netzwerk-Daten-Verkehr auf IP-Ebene benötigt Routing-Information auf Ethernet-Ebene.
- An welche MAC-Adresse muss das Paket geschickt werden?

ARP-Broadcast

- Wer kennt die IP-Adresse `www.xxx.yyy.zzz`?
- ARP-Protokoll. Ethernet-Frame-Typ 0x0806
- ARP-Broadcast an alle Subnetz-Rechner
  Zieladresse `FF:FF:FF:FF:FF:FF` (=Broadcast).

ARP-Antwort

- Ein (oder mehrere) Host antworten.
- Adresse des Antwort-Hosts ist gesuchte Next-Hop-Adresse.
- Host muss nicht Zielrechner sein, sondern kann auch Gateway (Router) sein.

Beispiel

Windows und Linux

- Speichern MAC-Adressen in ARP-Cache.
- Einträge sind eine Zeit lang (z.B. 20 Minuten) gültig.

Windows

```
C: \> arp -a
Interface: 192.168.1.101 --- 0x80004
  Internet Address      Physical Address    Typ
  192.168.1.1           00-0d-6d-bc-a8-6b   dynamic
  192.168.1.2           00-0e-1c-2b-e5-3c   dynamic
```

Linux

```
$ sudo arp
Address                  HWtype  HWaddress           Iface
192.168.1.1              ether   00:0D:6D:BC:A8:6B   eth0
192.168.1.2              ether   00:0E:1C:2B:E5:3C   eth1
```

IP

IP (*Internet Protokoll*)

- Routet Daten-Pakete durch das Internet.
  - Globale Zustellung von Daten-Paketen.
- Benötigt eindeutige Netzwerkadresse
  - 32-Bit IP-Adresse (IPv4)
  - 128-Bit IP-Adresse (IPv6)
- Pakete werden einzeln geroutet (verbindugslos).
- Unabhängig von Übertragungstechnologie.
  - Protokoll der Verbindungsschicht (z.B. Ethernet) hat andere Adressierung.
- IP ist kein verlässliches Service (*best effort delivery*).
  - Paketverlust kann auftreten.
  - Reihenfolge der Pakete kann geändert sein.
  - Paketdaten können verändert sein.

IPv4-Adresse

Jedes End-Gerät (*host*) und jede Vermittlungseinheit (*switching node*) im Internet hat eine **IP-Adresse** der Länge 32-Bit, die die **Netz- und Hostnummer** kodiert.

Die Kodierung ist eindeutig, zwei unterschiedlichen Geräte in einem Netz haben unterschiedliche IP-Adressen.

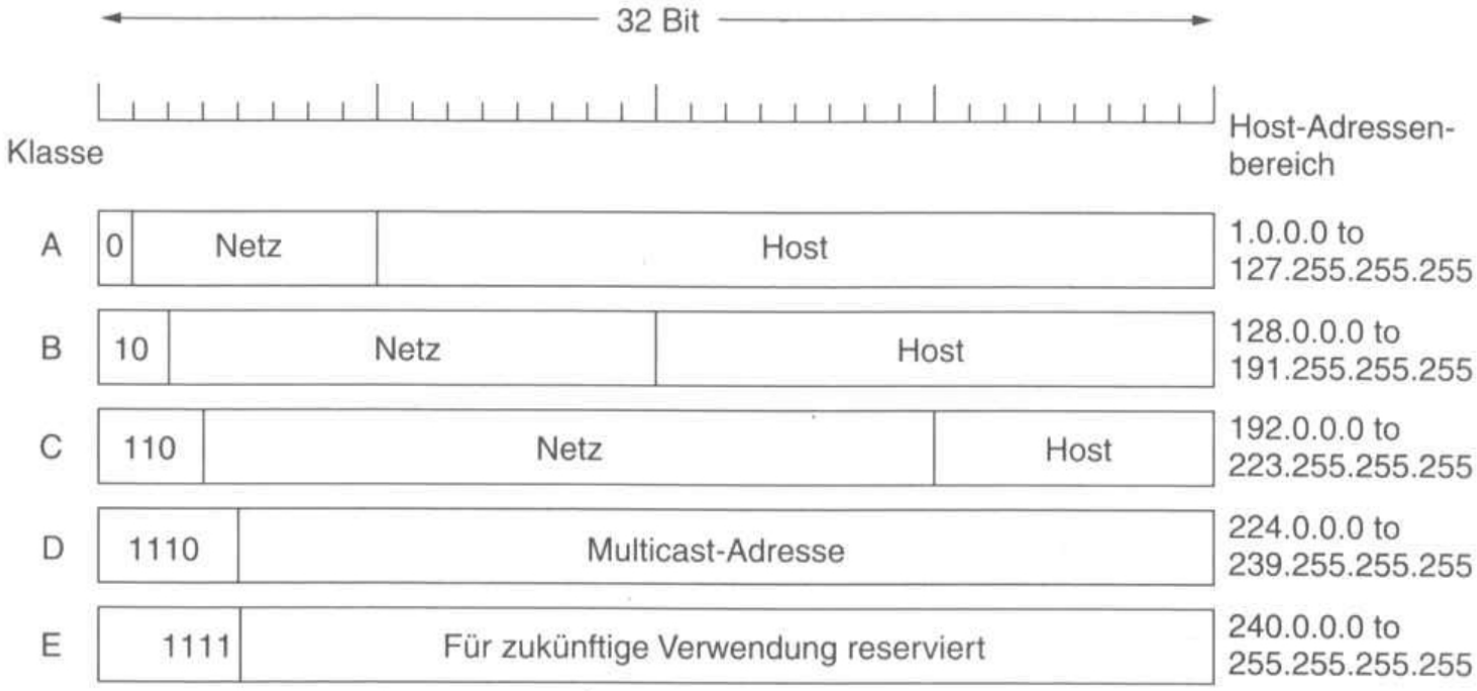
Die IP-Adresse bezieht sich nicht auf das Gerät, sondern auf die Netzwerkschnittstelle.

Ist ein Gerät in zwei Netzen verfügbar, muss es zwei IP-Adressen haben.

IP-Adressen werden in fünf Klassen eingeteilt, diese Zuordnung wird als klassenbasierte IP-Adressierung *Classful Adressing* bezeichnet.

Ausblick
Klassenbasierte IP-Adressierung wird abgelöst durch *Classless InterDomain Routing (CIDR)*.

Klassenbasierte IP-Adressierung

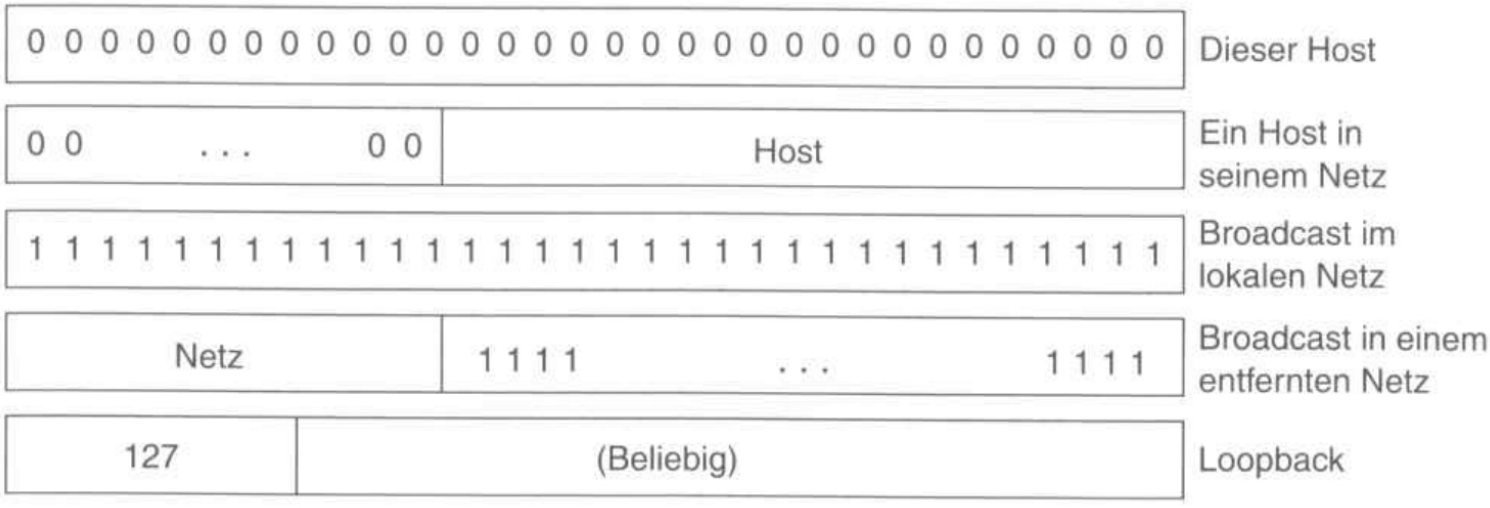


IP-Adressen sind 32-Bit Zahlen, die in Dezimalzahldarstellung mit Punkten (*Dotted Decimal Notation)* geschrieben werden. In diesem Format werden jeweils 4-Byte dezimal geschrieben (0-255) und durch einen Punkt getrennt.

Die niedrigste Adresse ist 0.0.0.0, die höchste 255.255.255.255. Diese beiden Adressen haben jeweils eine besondere Bedeutung.

Um Konflikte bei der Wahl/Vergabe von Netznummern zu vermeiden, werden diese von der gemeinnützigen Organisation *Internet Corporation for Assigned Names and Numbers (ICANN)* verwaltet.

Spezielle IPv4-Adressen



Die IP-Adresse 0.0.0.0 wird von einem Host benutzt, wenn er gebootet wird.

IP-Adressen mit 0 als Netzwerknummer beziehen sich auf das aktuelle Netz. So kann sich ein Rechner auf sein Netz beziehen ohne dessen Nummer zu kennen. Allerdings muss er die Klasse kennen um zu wissen wieviele Nullen nötig sind.

Mit der Adresse 255.255.255.255 (nur Einsen) ist Broadcast (das Senden einer Nachricht an alle Rechner) im lokalen Netz möglich.

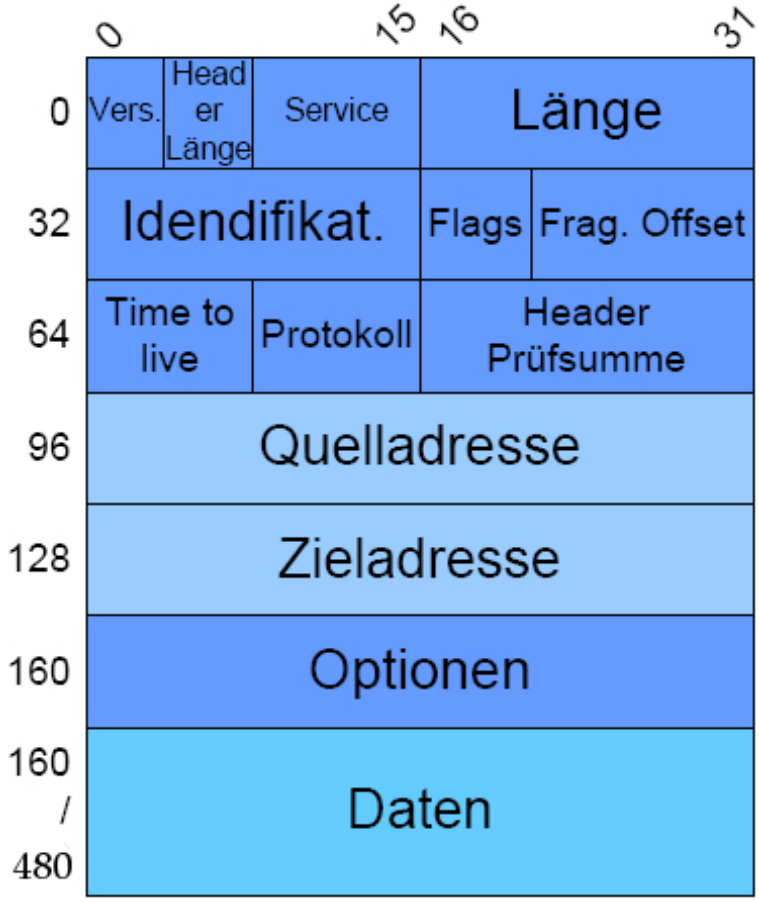
Adressen mit einer gültigen Netznummer gefolgt von Einsen erlauben Broadcast in dieses Netz.

Pakete, die an Adressen der Form 127.x.y.z gesendet werden, werden nicht ins Netz weitergegeben, sondern als Eingangspakete behandelt (**Loopback**).

IPv4-Paket (IPv4-Datagram)

Header (20-60 Byte) mit Big-Endian Darstellung von 32-Bit Wörtern.

- *Version*, 4=IPv4
- *Header Length* in 32-Bit Wörtern.
- *Type of Service* für Quality-of-Sevices.
- *Total Length*, Header + Daten in Bytes
  20 Byte bis 65535 Bytes
- *Identification, Flags, Fragment Offset*.
  Zur Erkennung von Fragmenten.
- *Time to Live (TTL).*
- *Protokoll*, 6=TCP, 17=UDP.
- *Quell-* und *Zieladresse* (32 Bit).
- *Header-Prüfsumme*, 16-Bit Summe.



Daten

- Maximale Größe $2^{16} - 1 = 65535$ Byte - Header.

Time to Live (TTL)

Time to Live (TTL) ist die Lebensdauer des Pakets, bei Wert 0 wird das Paket verworfen.

Verhindert, dass Pakete im Fall von Daten- oder Routing-Fehlern fortlaufend (z.B. auf einer Kreisroute) weitergeleitet werden.

Der Absender initialsiert dieses Feld (anhängig von Implementierung/Konfiguration) üblicherweise mit 64, 128 oder 255.

Jeder Router, den das Paket passiert, verringert den TTL-Wert mindestens um Eins.

Beispiel

Der TTL-Wert kann dazu genutzt werden, grob abzuschätzen, über wie viele Router die Ping-Pakete gelaufen sind.

```
ping -c 1 -n localhost
PING 127.0.0.1 56(84) bytes of data.
64 bytes from 127.0.0.1: ttl=64 time=0.025 ms
...
```

Beispiel

```
> ping -c 3 www.gwdg.de
PING 134.76.10.47 56(84) bytes of data.
64 bytes from 134.76.10.47: ttl=61 time=0.794 ms
64 bytes from 134.76.10.47: ttl=61 time=0.833 ms
64 bytes from 134.76.10.47: ttl=61 time=0.873 ms

--- ping statistics ---
3 packets, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.794/0.833/0.873 ms
```

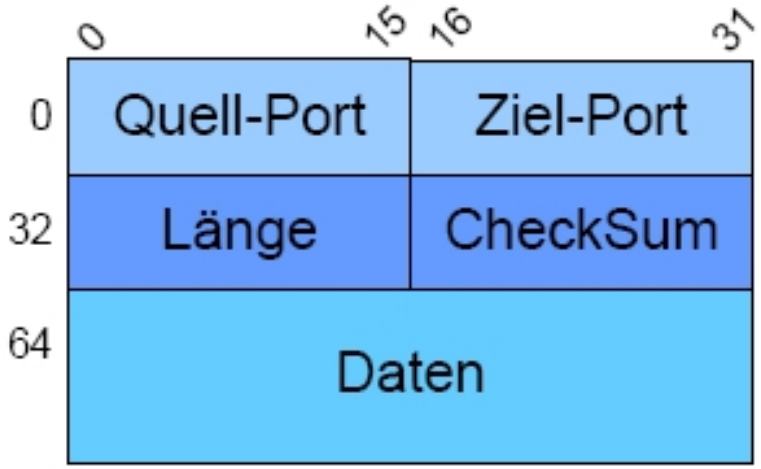
```
> ping -c 3 www.google.de
PING 72.14.221.99 56(84) bytes of data.
64 bytes from 72.14.221.99: ttl=53 time=17.7 ms
64 bytes from 72.14.221.99: ttl=53 time=17.0 ms
64 bytes from 72.14.221.99: ttl=53 time=15.3 ms

--- ping statistics ---
3 packets, 3 received, 0% packet loss, time 2008ms
rtt min/avg/max = 15.375/16.716/17.736 ms
```

UDP

UDP (*User Datagram Protocol*)

- Nicht verbindungsorientiert, d.h. kein Verbindungsaufbau nötig.
- *Unreliable datagram protocol*
- Keine Fehlerkontrolle, deshalb nicht für Anwendungen geeignet, die fehlerfreie Übertragung benötigen.
- Wenig Overhead, deshalb gut für Anwendungen geeignet, die schnelle Übertragung benötigen.
- ca. 20% des Internet-Verkehrs



TCP (1/2)

TCP (*Transmission Control Protocol*)

- Verbindungsorientiert
  1. Aufbau einer Ende-zu-Ende Verbindung.
  2. Nutzen der Verbindung, d.h. Austausch von Paketen.
  3. Abbau der Verbindung.
- Garantierte Datenverbindung
  - Fehlerkontrolle.
  - Fehlerkorrektur durch erneutes Senden von Daten.
- Paketgröße
  - Größer als bei IP.
  - TCP segmentiert große Daten.
- Rund 75% des Internet-Verkehrs
  - WWW und Email baut auf TCP auf.

TCP (2/2)

Protokoll

- Empfänger bestätigt
  - Vollständigen Paketempfang
  - Acknowledge (ACK)
- Sender sendet erneut
  - Falls ACK ausbleibt.
  - Nach Ablauf einer Timeout-Periode, die von der der Round-Trip-Time (RTT) abhängt.

TCP-Zustände

- Verbindungsorientierung, d.h. Zustände (*states*) werden benötigt.
- Zustände `LISTEN`, `SYN-SENT`, `SYN-RECEIVED`, `ESTABLISHED`, `FIN-WAIT-1`, `FIN-WAIT-2`, `CLOSE-WAIT`, `CLOSING`, `LAST-ACK`, `TIME-WAIT`, `CLOSED`.

Round-Trip-Time (RTT)

Die **Round-Trip-Time (RTT)** ist die Zeit, die vom Senden einer Anfrage bis zum Empfangen der Bestätigtigung vergeht.

Beispiel

```
> ping -c 3 www.gwdg.de
PING 134.76.10.47 56(84) bytes of data.
64 bytes from 134.76.10.47: ttl=61 time=0.794 ms
64 bytes from 134.76.10.47: ttl=61 time=0.833 ms
64 bytes from 134.76.10.47: ttl=61 time=0.873 ms

--- ping statistics ---
3 packets, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.794/0.833/0.873 ms
```

```
> ping -c 3 www.google.de
PING 72.14.221.99 56(84) bytes of data.
64 bytes from 72.14.221.99: ttl=53 time=17.7 ms
64 bytes from 72.14.221.99: ttl=53 time=17.0 ms
64 bytes from 72.14.221.99: ttl=53 time=15.3 ms

--- ping statistics ---
3 packets, 3 received, 0% packet loss, time 2008ms
rtt min/avg/max = 15.375/16.716/17.736 ms
```

TCP-Verbindung

Verbindungsaufbau

- 3-fach Handshake
  1. Client sendet SYN zu Server.
  2. Server antwortet mit SYN-ACK.
  3. Client schickt ACK.
- Client und Server tauschen Sequenznummern aus

Datenverbindung

- Jedes Byte Daten erhöht die Sequenznummer, so wird die Reihenfolge der Segmente festgelegt.
- Empfänger schickt ACK mit Sequenznummer.

Verbindungsabbau

- 4-fach Handshake
  - FIN und ACK in beide Richtungen.

TCP-Segment

Header

- Quell- und Ziel-Port
  - IP-Adressen stehen im IP-Paket-Header.
- Sequenz-Nummer
  - Fortlaufende Nummer für Segmente.
  - Zum Ordnen empfangener Segmente.
- Data Offset
  - Länge des TCP-Headers.
- Flags
  - SYN, ACK, RESET, FIN, ...
- Window
  - Gewünschte Antwortlänge.
- Prüfsumme
  - Einfach: 16-Bit Summe (kein CRC).
  - Header und Payload.

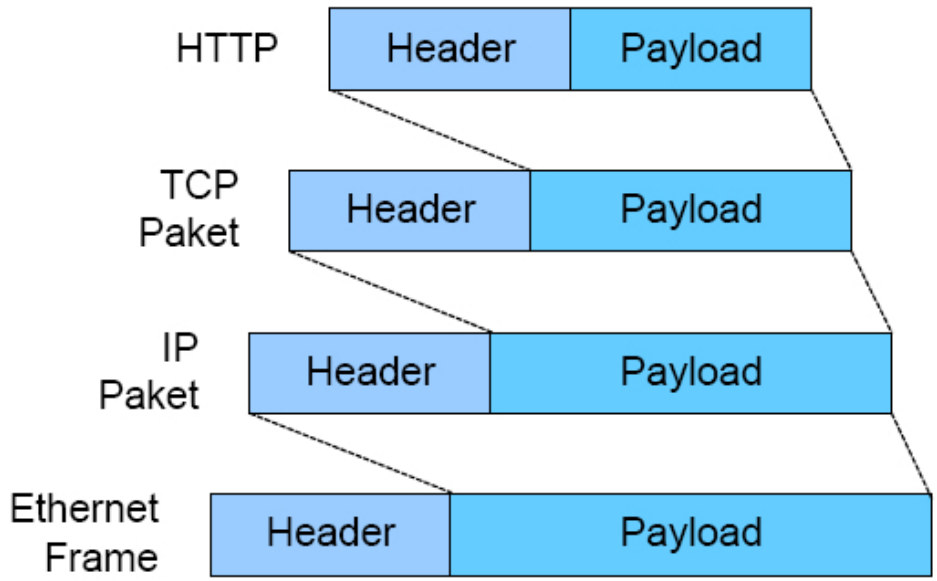
Daten

- Die eigentlichen Daten.

TCP in IP in Ethernet

Kapselung der *protocol data units* (Segmente, Pakete, etc.) als *payload* (Daten).

- Header und Daten des höherem Protokolls werden Daten des darunter liegenden Protokolls.
- Tiefere Protokollschichten kapseln höhere Protokollschichten.



Umgedrehte Reihenfolge, höchstes Protokoll ist am tiefsten vergraben.

Inhalt

Literatur

A. Beutelspacher, J. Schwenk, K.-D. Wolfenstetter
Moderne Verfahren der Kryptographie,
Vieweg+Teubner, Mai 2010.

C. Damm
Kryptographie,
Skript zur Vorlesung, 2004.

B. Schneier, N. Ferguson
Practical Cryptography,
John Wiley & Sons, 2003.

A. J. Menezes, P. C. Van Oorschot, S. A. Vanstone
Handbook of Applied Cryptography,
Crc Press, Oktober 1996.

Geheimhaltung

Grundproblem
Wie kann man mit jemanden vertraulich kommunizieren, d.h. kein Unbeteiligter soll Kenntnis von der übermittelten Nachricht erhalten?

Das Problem der Übermittlung (und Speicherung) geheimer Nachrichten kann man durch verschiedene Maßnahmen lösen.

- Organisatorische Maßnahmen
- Physikalische Maßnahmen
- Kryptographische Maßnahmen

Organisatorische Maßnahmen

Beispiel

- Ein Gespräch während eines einsamen Waldspaziergangs.
- Übermittlung einer Nachricht durch einen vertrauenswürdigen Boten.
- Einstufung vertraulicher Dokumente als Verschlusssache.

Physikalische Maßnahmen

Beispiel

- Verstecken der Informationen in einem Tresor.
- Übermitteln der Nachricht in einem versiegelten Brief.
- Verheimlichen der Existenz der Nachricht, z.B. durch Geheimtinte.

Kryptographische Maßnahmen

Kryptographische Maßnahmen verändern bzw. entstellen (**verschlüsseln**, **chiffrieren**) die Nachricht (den **Klartext**).

Dadurch ist die Nachricht für einen Außenstehenden nicht mehr erkennbar und die übertragene Information (der **Geheimtext**) erscheint diesem (meist) völlig unsinnig.

Ein berechtigter Empfänger kann die Nachricht aber (leicht) wieder herstellen (**entschlüsseln**, **dechiffrieren**).

Der älteste Zweig der klassischen Kryptographie beschäftigt sich mit der **Geheimhaltung** von Nachrichten **durch Verschlüsselung**.

Inhalt

Symmetrische Verschlüsselung

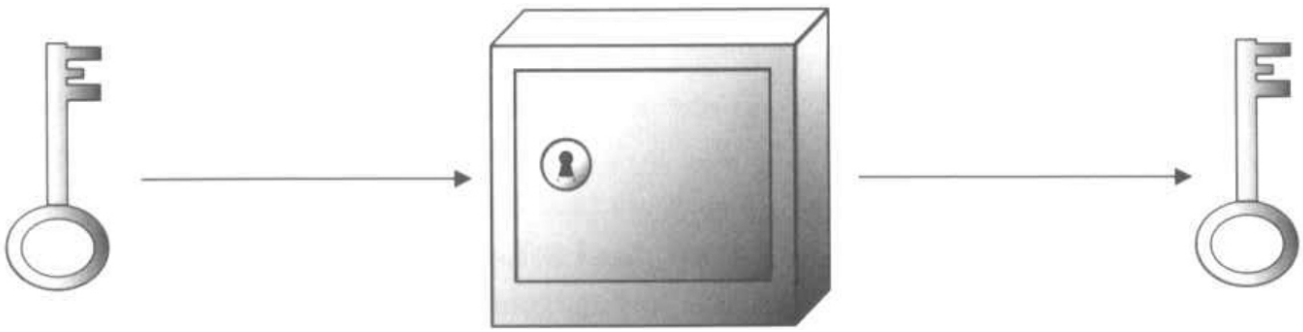
Bei der **symmetrischen Verschlüsselung** besitzen der Sender und die berechtigten Empfänger eine gemeinsame geheime Zusatzinformation (den **Schlüssel**), darin unterscheiden sie sich von den Außenstehenden.

Derselbe Schlüssel wird sowohl vom Sender zum Verschlüsseln des Klartext verwendet, als auch vom Empfänger für das Entschlüsseln des Geheimtext benötigt.

Beim Sender ist das kein Problem, da er den Schlüssel gewählt/erzeugt hat.

Dem Empfänger fehlt dieser Schlüssel erstmal, deswegen ist es bei der symmetrischen Verschlüsselung sehr wichtig, dass der Schlüssel auf einem sicheren Übertragungsweg an den Empfänger weitervermittelt wird.

Anschauung



Verschlüsseln schützt die Nachricht davor gelesen zu werden.

Man kann sich vorstellen, dass der Sender die Nachricht in einen Tresor legt und mit Hilfe seines Schlüssels abschließt.

Der Tresor wird samt Inhalt an den Empfänger geschickt. Dieser hat einen identischen Schlüssel, um den Tresor zu öffnen und die Nachricht zu lesen.

In der Kryptographie werden die Nachrichten nicht durch physikalische Maßnahmen geschützt, sondern durch mathematische Methoden.

Verschlüsselungsalgorithmus (1/2)

Definition

Ein **symmetrischer Verschlüsselungsalgorithmus** besteht aus einer Funktion f mit zwei Eingabewerten, dem **Schlüssel** k und dem **Klartext** m, die Ausgabe ist der **Geheimtext** c, der sich aus k und m ergibt.

Verschlüsselungsalgorithmus (2/2)

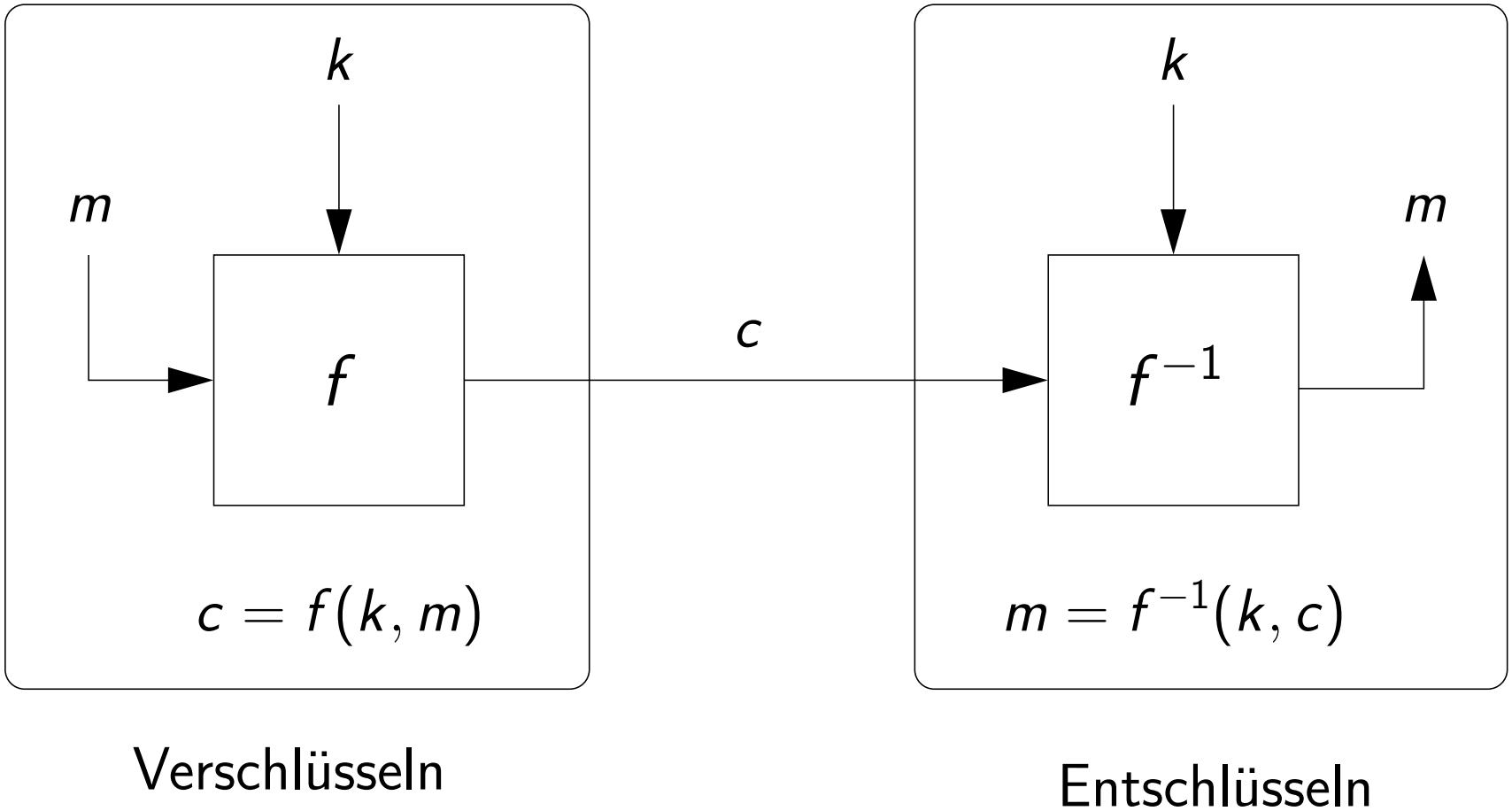
Die Verschlüsselungsfunktion f muss **umkehrbar** sein, d.h. es muss eine Funktion f^{-1} geben, die die Wirkung von f rückgängig macht.

Mit dem Schlüssel k und dem Gemeintext c kann man mit f^{-1} den Klartext m rekonstruieren.

Angenommen Sender und Empfänger benutzen den gemeinsamen (geheimen) Schlüssel k.

- Der Sender verschlüsselt einen Klartext m, indem er den Gemeintext c berechnet, $c = f(k, m)$ (oft auch $f_k(m)$).
- Der Empfänger rekonstruiert den Klartext m, indem er den Geheimtext c entschlüsselt, $m = f^{-1}(k, c) = f_k^{-1}(c)$

Funktionsschema



Verschlüsseln und Entschlüsseln müssen in einer sicheren Umgebung stattfinden, das wird im Bild durch die Kästen symbolisiert.

Inhalt

Verschlüsselungsverfahren

In der Regel sollen große Nachrichten oder Nachrichtenströme, d.h. eine kontinuierliche Abfolge von Zeichen, verschlüsselt werden, was mit einer einmalige Anwendung der Verschlüsselungsfunktion nicht umsetztbar ist.

Blockchiffren und **Stromchiffren** sind Verfahren für die Anwendung von Verschlüsselungsalgorithmen auf große Nachrichten oder Nachrichtenströme.

Beide Verschlüsselungsverfahren teilen den Klartext in Blöcke gleicher Größe $m_1, m_2, \ldots$ auf.

Durch die Verschlüsselung entstehen Geheimtextblöcke $c_1, c_2, \ldots$ gleicher Größe.

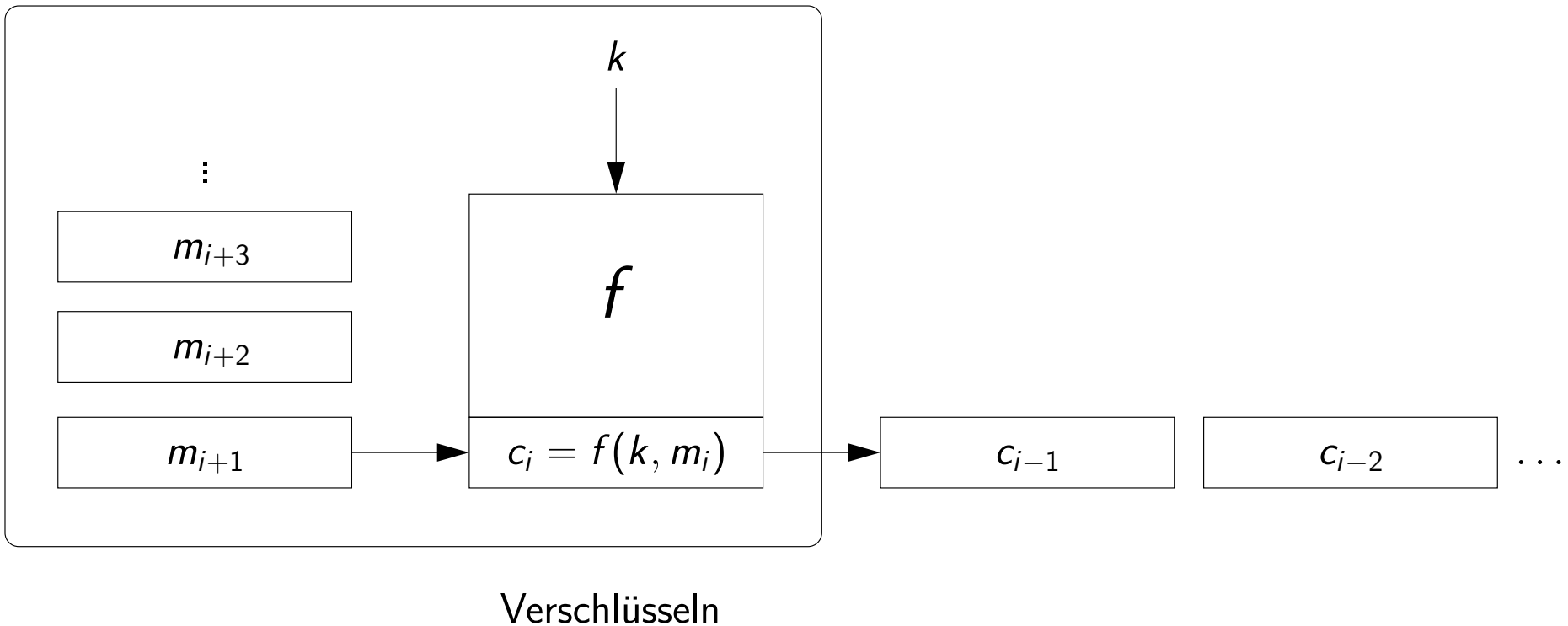
Blockchiffren (1/2)

Blockchiffre

- Die Klartextblöcke werden unabhängig voreinander mit der Verschlüsselungsfunktion und **demselben Schlüssel** einzeln zu Geheimtextblöcken chiffriert.
- Die Geheimtextblöcke werden entsprechend einzeln dechiffriert.

Blockchiffren (2/2)

Blockchiffre mit Verschlüsselungsfunktion f und Schlüssel k.



Verschlüsseln

Caesar-Verschlüsselung

Die **Caesar-Verschlüsselung** ist eine Blockchiffren mit symmetrischem Verschlüsselungsalgorithmus.

Ein Klartext m kann eine beliebige Zeichenfolge über dem Alphabet $\Sigma = \{A, B, \ldots, Z\}$ der 26 Großbuchstaben sein, $m \in \Sigma^*$.

Ein Klartext wird in Blöcke der Größe ein Zeichen aufgeteilt.

Der Schlüssel k ist ein Zeichen aus dem Alphabet Σ.

Die symmetrische Verschlüsselungsfunktion $f_k : \Sigma \to \Sigma$ verschiebt das übergebene Zeichen im Alphabet zyklisch nach **rechts**, dabei wir die Anzahl der zu verschiebenden Stellen von der Position des Schlüssels k im Alphabet bestimmt.

Die symmetrische Entschlüsselungsfunktion $f_k^{-1} : \Sigma \to \Sigma$ verschiebt entsprechend, abhängig von k, das übergebene Zeichen im Alphabet zyklisch nach **links**.

Beispiel

Caesar-Verschlüsselung mit Schlüssel C.

Die Position von Schlüssel C im Alphabt ist 3, d.h f_C (f_C^{-1}) verschiebt ein übergebenes Zeichen um 3 Stellen im Alphabet zyklisch nach rechts (links).

$$
\begin{aligned}
f_C(A) &= D & f_C^{-1}(D) &= A \\
f_C(B) &= E & f_C^{-1}(E) &= B \\
&\dots & &\dots \\
f_C(W) &= Z & f_C^{-1}(Z) &= W \\
f_C(X) &= A & f_C^{-1}(A) &= X \\
f_C(Y) &= B & f_C^{-1}(B) &= Y \\
f_C(Z) &= C & f_C^{-1}(C) &= Z
\end{aligned}
$$

CAESAR wird zeichenweise mit C verschlüsselt zu *FDHVDU*, das wird zeichenweise mit C entschlüsselt wieder zu *CAESAR*.

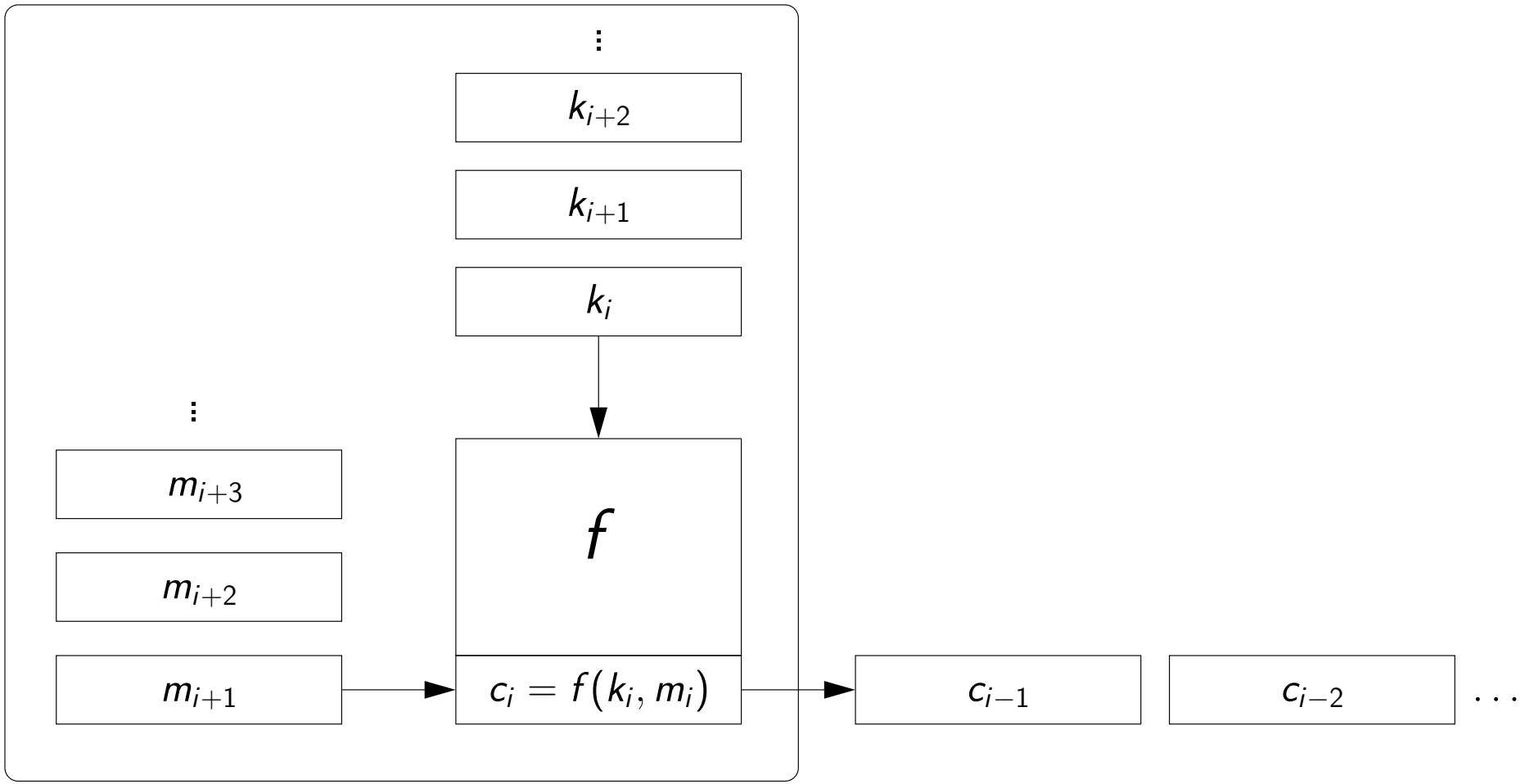
Stromchiffren

Stromchiffre

- Eine **Schlüsselstrom** $k_1, k_2, \ldots$ wird erzeugt, sodass für jeden Klartextblock ein eigener Schlüssel vorliegt.
- Ein Klartextblock wird jeweils mit der Verschlüsselungsfunktion und dem zugehörigen Schlüssel des Schlüsselstroms zu einem Geheimtextblock chiffriert.
- Ein Geheimtextblock kann nur mit den zugehörigen Schlüssel des Schlüsselstroms dechiffriert werden.

Stromchiffre

Stromchiffre mit Verschlüsselungsfunktion f und Schlüsselstrom $k_1, k_2, \ldots$.



Verschlüsseln

One-Time-Pad

One-Time-Pad (wörtlich *Einmal-Block*) ist eine Stromchiffre für (große) Nachrichten bekannter Länge mit symmetrischem Verschlüsselungsalgorithmus.

Es werden **zufällig** genauso viele Schlüssel erzeugt, wie Klartextblöcke zu verschlüsseln sind, dabei haben die Schlüssel mindestens die gleiche Länge wie die Klartextblöcke.

Jeder Schlüssel wird nur **einmal** verwendet, um einen Klartextblock mit der Verschlüsselungsfunktion in einen Geheimtextblock zu chiffrieren.

Bemerkung

Da es für jeden Klartextblock einen Schlüssel gibt, die Schlüssel zufällig gewählt und nur einmal verwendet werden, kann man zeigen, dass One-Time-Pad ein absolut sicheres Verfahren ist.

Bemerkung

One-Time-Password (*Einmalkennwort*) ist ein anderes Verfahren, das z.B. durch Transaktionsnummern-Listen (TAN-Listen) beim Online-Banking Anwendung findet.

Beispiel (1/2)

Für ein Umsetzung von One-Time-Pad wird die Verschlüsselungsfunktion $f_k : \Sigma \to \Sigma$ des Caesar-Verschlüsselung benutzt, mit dem Zusatz, dass das Alphabet $\Sigma = \{A, B, \ldots, Z, +, 0, 1, 2, \ldots 9\}$ an Position 27 das Pluszeichen und an den Positionen 28 bis 37 die Ziffern 0 bis 9 enthält.

Um den Text

`15+MAERZ+CAESAR+TREFFEN+DOLCHE+NICHT+VERGESSEN`

zu verschlüsseln wird eine ebenso lange zufällige Schlüsselfolge

`CTO+SHBF++OOEXKKS9OS+MPJM+Y+EXOPFRB+YNZSDX56TS`

erzeugt, z.B. durch Beobachtung der Bewegung von Blättern auf dem *Forum Romanum*.

Die Zeichen der zu verschlüsselnden Nachricht werden einzeln mit der Verschlüsselungsfunktion f_k und jeweils einem Zeichen der Schlüsselfolge zum Geheimtext

`4PECTMT4Q215XY1ABRTY5R29QE92M1E2OUJJO839K1OPY5`

chiffriert.

Beispiel (2/2)

Klartext:	`15+MAERZ+CAESAR+TREFFEN+DOLCHE+NICHT+VERGESSEN`
Schlüsselfolge:	`CTO+SHBF++OOEXKKS9OS+MPJM+Y+EXOPFRB+YNZSDX56TS`
Geheimtext:	`4PECTMT4Q215XY1ABRTY5R29QE92M1E2OUJJO839K1OPY5`

Zum Austausch von verschlüsselten Nachrichten ist das Verfahren aber nicht geeignet, denn zum Entschlüsseln braucht man die Schlüsselfolge.

Wird die Schlüsselfolge vorher oder parallel zum Geheimtext (z.B. durch zwei unabhängige Boten) ausgetauscht, degeneriert das Verfahren zu einer Blockchiffre mit einem großen Blocke und einem großen Schlüssel.

Zufall (1/2)

In der Kryptographie spielen Zufallszahlen und Zufallsfolgen eine wichtige Rolle.

Dabei gibt es verschiedenen Aspekte.

- In vielen Kommunikationsprotokollen muss an einer bestimmten Stelle einen zufälligen Wert wählen. Dabei hängt die Sicherheit des Protokolls direkt davon ab **wie zufällig** der Wert gewählt wurde.

  Im eigenen Interesse müssen die Kommunikationspartner darauf achten den Wert mit einem möglichst guten Zufallsgenerator zu wählen.
- Oft ist es nicht praktikabel oder sogar unmöglich echte Zufallszahlen und Zufallsfolgen zu verwenden. Das ist z.B. dann der Fall, wenn die Zahlen oder Folgen von mehreren erzeugt werden müssen.

  In diesem Fällen werden Pseudozufallswerte verwendet, die mit Hilfe eines deterministischen Algorithmus berechnet werden, aber für Außenstehende (z.B. Lauscher) zufällig aussehen.

Zufall (2/2)

Echte Zufallszahlen und Zufallsfolgen werden mit Hilfe physikalischer Phänomene erzeugt, z.B. mit Hilfe des Rauschens elektronischer Bauelemente, dem radioaktiven Zerfall oder dem Konvektionsströmungen in einer Lavalampe.

Ein klassisches Beispiel für das Erzeugen echte Zufallsfolgen ist das Werfen einer *fairen Münze*.

Bei der Beurteilung der Güte eines Zufallsfolgengenerators ist ein Problem, dass man die Zufälligkeit einer Folge **nicht** beweisen kann.

Die nicht Zufälligkeit einer Folge lässt sich beweisen, durch Angabe eines Algorithmus zur Erzeugung der Folge.

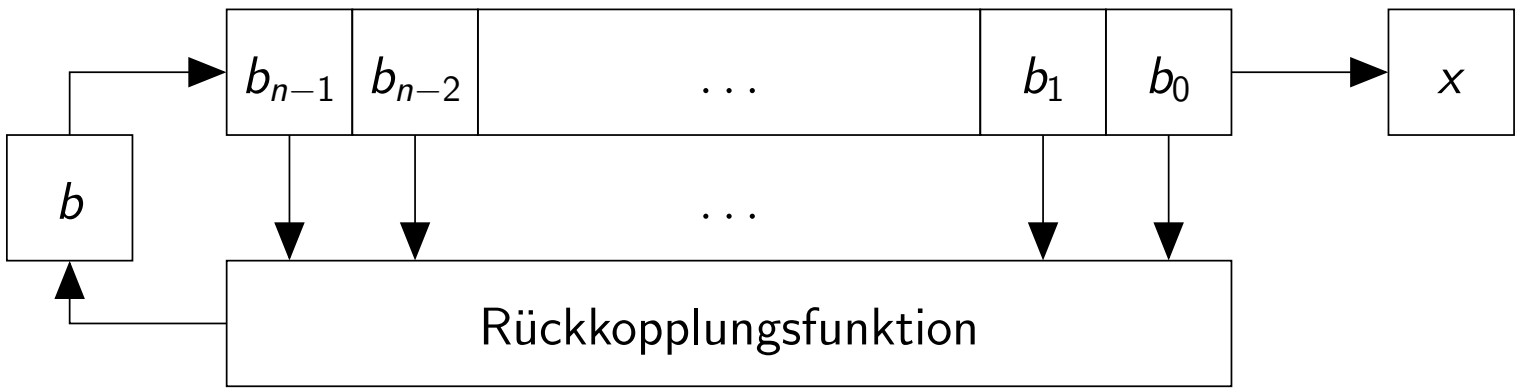
Bei Pseudozufallsfolgen steht man prinzipiell vor dem gleichen Problem. Es gibt keine Kriterien um echte Zufallsfolgen zu bewerten, deshalb kann man nicht entscheinden wie zufällig eine Pseudozufallsfolge aussieht.

Wiederum kann man nur das Gegenteil nachweisen, nämlich dass eine Pseudozufallsfolge nicht zufällig aussieht.

Lineare Schieberegister mit Rückkopplung (1/2)

Lineare Schieberegister werden häufig in der Kryptographie verwendet, da sie leicht in digitalen Schaltwerken zu realisieren sind.

Die Schieberegister mit Rückkopplung (*linear feedback shift register, LSR*) bestehen aus zwei Teilen, dem Schieberegister und der Rückkopplungsfunktion.



Lineare Schieberegister mit Rückkopplung (2/2)

Die Funktion eines n-stelligen Schieberegisters.

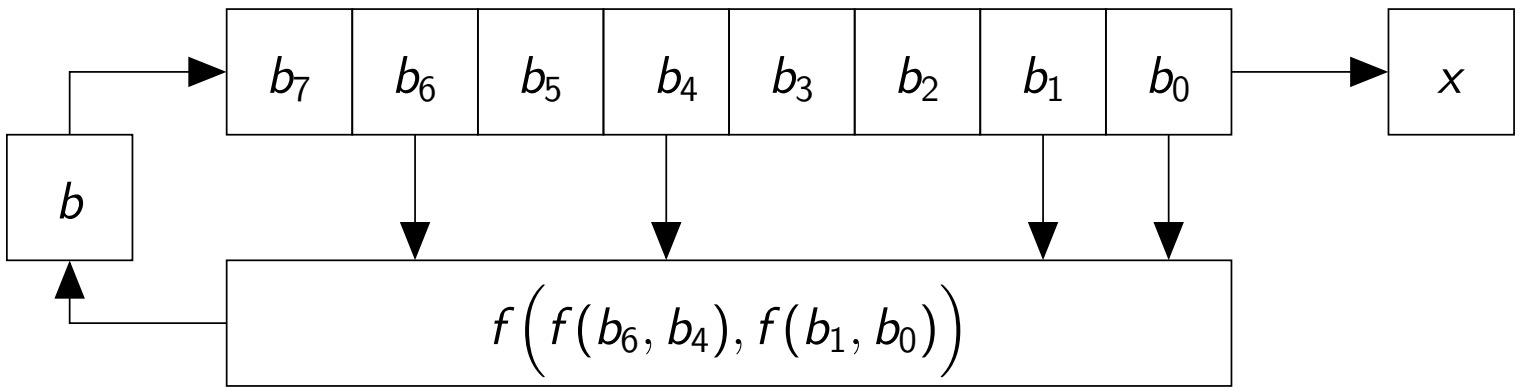
- Die Ausgabe des Schieberegister ist $x = b_0$.
- Das b wird in Abhängigkeit von $b_{n-1}, \ldots, b_0$ und der Rückkopplungsfunktion berechnet.
- Die Einträge des Schieberegister werden mit folgender Regel nach rechts verschoben $b_i = b_{i+1}$ für $i = 0, \ldots, n-2$.
- Von links wird b in das Schieberegister hinein geschoben $b_{n-1} = b$.

Eine Rückkopplungsfunktion kann mehr oder weniger kompliziert sein.

Beispiel (1/3)

Sei, wie im One-Time-Pad-Beispiel,

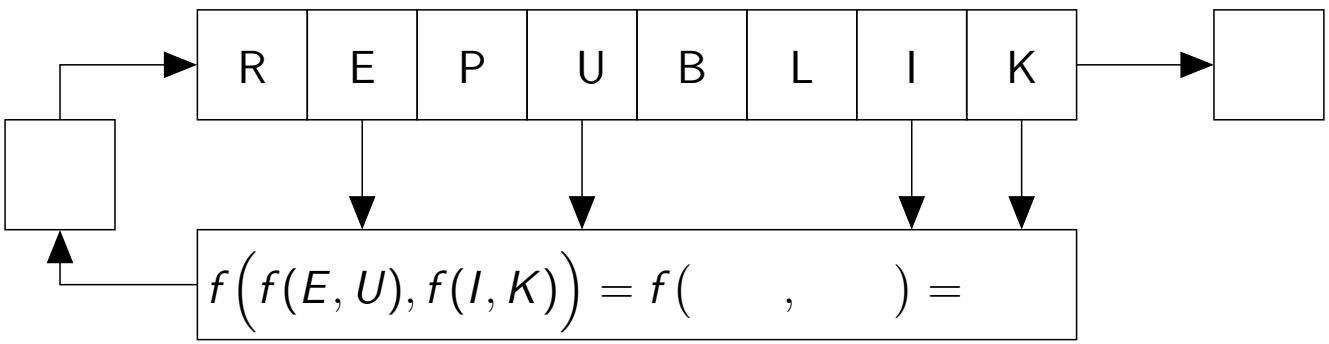
- $\Sigma = \{A, B, \ldots, Z, +, 0, 1, 2, \ldots 9\}$ das Alphabet,
- $f(k, m) : \Sigma \times \Sigma \to \Sigma$ die Verschlüsselungsfunktion der Caesar-Verschlüsselung.



Beispiel (2/3)

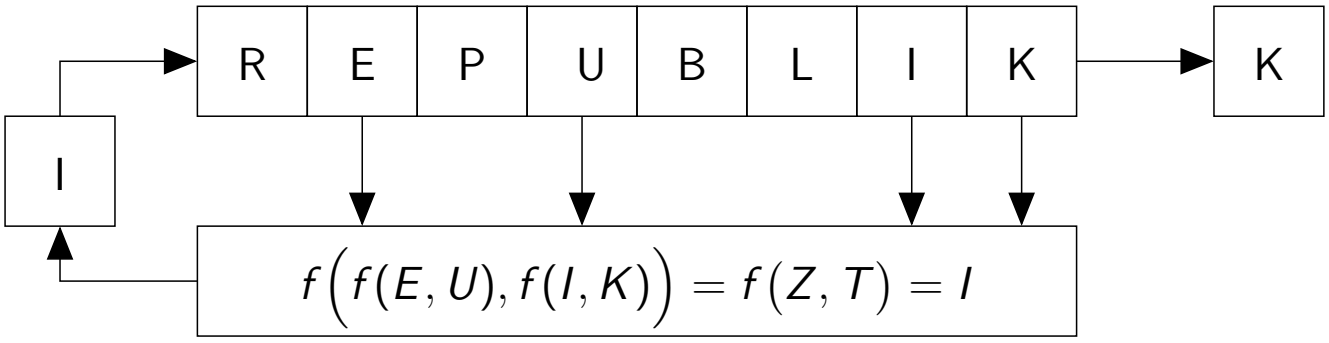
Initialisierung des Schieberegisters mit dem Schlüssel `REPUBLIK`.

```
                  11111111112222222223333333
1234567890123456789012345678901234567
-------------------------------------
ABCDEFGHIJKLMNOPQRSTUVWXYZ+0123456789
```



Beispiel (2/3)

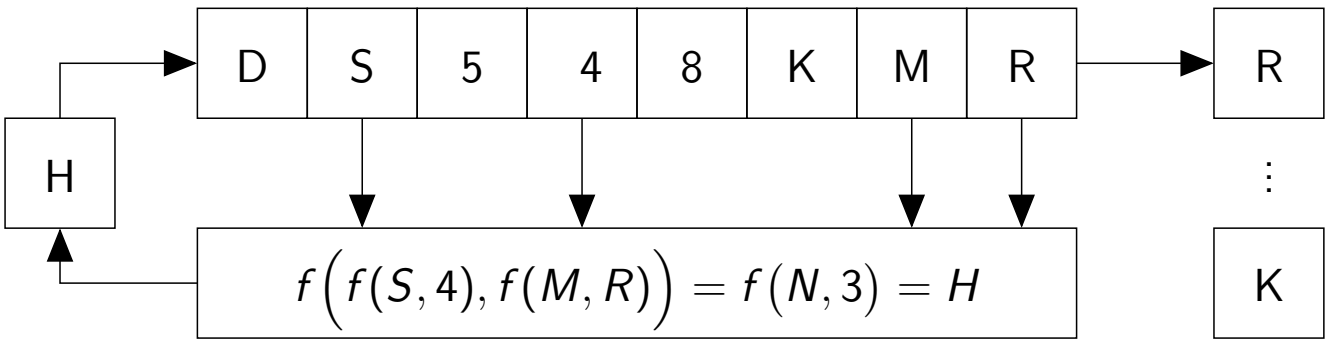
Initialisierung des Schieberegisters mit dem Schlüssel `REPUBLIK`.



Dann ergibt sich der Schlüsselstrom

`KILBUPERIR0V9XN8D43PVL13M4O+YZZWVVPWOWFQOD8V0R`

und folgender Zustand des Schieberegisters.



Beispiel (3/3)

Mit dem Schlüssel `REPUBLIK` können mit Hilfe eines lineare Schieberegisters mit (öffentlich) bekannter Rückkopplungsfunktion der Sender und jeder berechtigte Empfänger einen identischen Schlüsselstrom

`KILBUPERIROV9XN8D43PVL13M4O+YZZWVVPWOWFQOD8VOR`

erzeugen mit dem der Sender den Klartext

`15+MAERZ+CAESAR+TREFFEN+DOLCHE+NICHT+VERGESSEN`

zum Geheimtext

`CEBOVUWG8U1+SY4ZXM8VOQFUQJ+253P93YXFEHK77IRD54`

verschlüsseln kann und die berechtigten Empfänger den Klartext wiederherstellen können.

Angreifer haben es sehr schwer den Geheimtext zu entschlüsseln (siehe One-Time-Pad).

Linear/nicht linear

Folgen, die mit linearen Schieberegistern erzeugt werden haben sehr gute statistische Eigenschaften, d.h. die einzelnen Buchstaben, Bits, etc. sind in einer ausreichend großen Stichprobe gleichmäßig verteilt.

Trotzdem kann bei dieser einfachen Erzeugung des Schlüsselstroms schon aus relativ kurzen bekannten Schlüsselfolgen (nicht Geheimtexten) der Schlüssel mit vertretbarem Aufwand berechnet werden, weil ein **lineares** Verfahren benutzt wurde.

In der Praxis werden deshalb **nicht lineare** Verfahren verwendet, d.h. entweder werden nicht lineare Schieberegister (mit komplexerer Rückkopplungsfunktion) benutzt oder es werden mehrere lineare Schieberegister nicht linear gekoppelt.

Inhalt

Kryptosystem

Ein **Kryptosystem** besteht aus

- Klartextmenge M über einem Alphabt Σ_1, $M \subseteq \Sigma_1^*$
- Kryptotextmenge C über einem Alphabet Σ_2, $C \subseteq \Sigma_2^*$
- Schlüsselmenge $K \neq \emptyset$
- Verschlüsselungsabbildungen $e : M \times K \to C$
- Entschlüsselungsabbildungen $d : C \times K \to M$
- Es gilt, dass für jedes $k' \in K$ (Schlüssel) ein $k'' \in K$ (Gegenschlüssel) existiert, sodass gilt $d(e(m, k'), k'') = m$ für alle $m \in M$.

Ein Kryptosystem heißt **symmetrisch**, wenn Schlüssel und Gegenschlüssel gleich sind, d.h. für alle $k \in K$ gilt $d(e(m, k), k) = m$ für alle $m \in M$.

Asymmetrische Kryptosystem, bei denen sich Schlüssel und Gegenschlüssel unterscheiden, sind erst seit Ende der siebziger Jahre bekannt.

Inhalt

Substitution (1/3)

Kryptosysteme werden nach der Wirkung von Verschlüsselungs- und Entschlüsselungsabbildungen unterschieden.

Substitutions-Chiffre
Die Symbole bleiben *wo* sie sind, aber nicht *was* sie sind.

Die Verschlüsselungsabbildung $e : \Sigma_1 \to \Sigma_2$ ist injektiv und induziert eine Abbildung $\Sigma_1^* \to \Sigma_2^*$.

$m_1 m_2 \dots m_t \mapsto e(m_1)e(m_2)\dots e(m_t) = c_1 c_2 \dots c_t$

Beispiel

- Caesar-Verschlüsselung
- Anwendung einer beliebige Permutation der Alphabetssysmbole

Substitution (2/3)

Monographische Substitution

- Es werden nur einzelne Symbole ersetzt.

Beispiel. Strom/Blockchiffren bei denen die Blöcke jeweils einzelne Symbole sind.

Monoalphabetische Substitution

- Gleiche Substitution auf jedes Symbol anwenden (unabhängig von der Position)

Substitution (3/3)

Polygraphische Substitution

- Ersetze Symbolpaare (-tripel, ...; allgemein: **Polygramme**)
- $e : \Sigma_1^\ell \to \Sigma_2^r$ (injektiv) induziert Abbildung $\Sigma_1^* \to \Sigma_2^*$:
  $m_1 \dots m_\ell m_{\ell+1} \dots m_{2\ell} \dots \mapsto e(m_1 m_2 \dots m_\ell) e(m_{\ell+1} \dots m_{2\ell}) \dots =$
  $c_1 \dots c_r c_{r+1} \dots c_{2r} \dots$

Polyalphabetische Substitution

- Benutze mehrere Substitutionen in bestimmter Reihenfolge
- $e_1, e_2, \dots, e_r : \Sigma_1 \to \Sigma_2$ (injektiv) induzieren Abbildung $\Sigma_1^* \to \Sigma_2^*$:
  $m_1 m_2 \dots m_t \mapsto e_{f(1)}(m_1) e_{f(2)}(m_2) \dots e_{f(t)}(m_t) = c_1 c_2 \dots c_t$,
  wobei $f : \{1, \dots, t\} \to \{1, \dots, r\}$

Beispiel. Vigenère, One-Time-Pad.

Vigenère-Chiffre

Die Vigenère-Chiffre stammt aus dem 16. Jahrhundert und wurde von dem französischen Kryptographen Blaise de Vigenère (1523-1596)1 entwickelt.

Basiert auf der Verwendung der Caesar-Chiffre, allerdings mit periodisch wechselnden Schlüsseln.

Galt lange Zeit als nicht zu knacken, insbesondere war das Ermitteln der Schlüssellänge problematisch, und konnte erst um 1850 entziffert werden.

Beispiel

```
Klartext:       15+MAERZ +CAESAR+ TREFFEN+ DOLCHE+N ICHT+VER GESSEN
Schlüsselfolge: REPUBLIK REPUBLIK REPUBLIK REPUBLIK REPUBLIK REPUBL
Geheimtext:     JAF6CQ+9 HHQZUM+A AWU+HQWA VT0XJQ8Y +HXD16N1 YJ7CGZ
```

```
                  11111111112222222223333333
 12345678901234567890123456789012345678901234567
 -----------------------------------------------
 ABCDEFGHIJKLMNOPQRSTUVWXYZ+0123456789
```

Transposition

Transpositions-Chiffre
Symbole bleiben *was* sie sind, aber nicht *wo* sie sind.

Abhängig vom Schlüssel wird die Position der Symbole mit der Permutation π vertauscht.

$m_1 m_2 \dots m_t \mapsto m_{\pi(1)} m_{\pi(2)} \dots m_{\pi(t)} = c_1 c_2 \dots c_t$

Beispiel. Matrixtransposition

Matrixtransposition

Bei der **Matrixtransposition** wird der Klartext in Zeilen gleicher Länge angeordnet (unvollständige Zeile werden aufgefüllt).

Zur Bildung des Geheimtextes wird der so angeordnete Text spaltenweise zurückgeschrieben.

Beispiel

Klartext zeilenweise (Zeilenlänge 5) anordnen.

```
15+MAER
Z+CAESA
R+TREFF
EN+DOLC
HE+NICH
T+VERGE
SSENXXX
```

Geheimtext spaltenweise zurückschreiben.

```
1ZREHTS5++NE+S+CT++VEMARDNENAEEOIRXESFLCGXRAFCHEX
```

Inhalt

Kryptoanalyse

Die Dechiffrierung ohne Kenntnis der Geheiminformation, das *Brechen der Chiffre*, wird **Kryptoanalyse** genannt.

Grundannahme (**Kerckhoffs' Maxime**, 1883)

Die Sicherheit einer Chiffre darf nicht darauf beruhen, dass der Angreifer (Kryptoanalyst) das benutzte Verfahren nicht kennt.

Angriffsarten und Analysemethoden (1/3)

ciphertext-only

- Gegeben: $c_1 = e(m_1, k), \ldots, c_i = e(m_i, k)$
- Gesucht: $m_1, \ldots, m_i$ oder k oder Algorithmus, um $m_{i+1}, \ldots$ aus $c_{i+1} = e(m_{i+1}, k), \ldots$ herzuleiten

Beispiel. Ein Lauscher (englisch *eavesdropper*), z.B. in einem Netzwerk.

Angriffsarten und Analysemethoden (2/3)

known-plaintext

- Gegeben: $(m_1, c_1 = e(m_1, k)), \ldots, (m_i, c_i = e(m_i, k))$
- Gesucht: k oder Algorithmus, um $m_{i+1}, \ldots$ aus $c_{i+1} = e(m_{i+1}, k), \ldots$ herzuleiten

Beispiel. Wiederkehrende Anfangs- und Schlußformeln.

Angriffsarten und Analysemethoden (3/3)

chosen-plaintext

- Wähle $m_1, \ldots, m_i$, beobachte $c_1 = e(m_1, k), \ldots, c_i = e(m_i, k)$
- Gesucht: k oder Algorithmus, um $m_{i+1}, \ldots$ aus $c_{i+1} = e(m_{i+1}, k), \ldots$ herzuleiten

adaptive chosen-plaintext

- Wähle m_1, beobachte c_1, wähle m_2, usw.

Beispiel.

- Freund-Feind-Erkennung (challenge-response-Protokoll)
- Public-Key-Systeme

Sichere/unsichere Kryptosystem (1/2)

System heißt **sicher** gegen bestimmten Angriffstyp, wenn ein potentieller Angreifer die erforderlichen Berechnungen nicht mit **vertretbarem Aufwand** durchführen kann.

Nachweis der Sicherheit ist schwierig
$\Rightarrow$ Rückführung auf anerkannt schwierige Probleme

- Faktorisierung großer Zahlen
- NP-vollständige Probleme
- ...

Vorsicht. Schwierige Probleme können auch einfache Instanzen haben!

Sichere/unsichere Kryptosystem (2/2)

In der Regel ist es einfacher, die Unsicherheit eines Systems gegen bestimmte Angriffe nachzuweisen.

Offenbar unsicher gegen chosen-plaintext-Angriffe sind

- Caesar,
- Vigenère,
- Matrixtransposition.

Diese Verfahren sind ebenfalls unsicher gegen ciphertext-only-Angriffe, wenn **genügend viel** Kryptotext vorhanden ist.

Klartextraum

Kenntnisse des Angreifers über den Klartextraum.

- Sprache (natürliche Sprache, Programmiersprache, ...)
- häufige Wörter (Kontext!)
- Sprachstatistik (Symbol-, Bigramm-, Trigramm-, ..., -Häufigkeiten)
- Buchstabenmuster (z.B. 1221 im Englischen: cabbage, ballast, apparant, ...)
- Randinformationen
- ...

Sprachen

Häufigkeitsmerkmale prägen natürliche Sprachen sehr stark
$\Rightarrow$ wichtigster Einstiegspunkt für Kryptoanalyse

Nützlich sind **Häufigkeitsreihenfolgen**, i.d.R. aber allein nicht ausreichend.

deutsch (verschiedene Quellen):

`enrisdutaghlobmfzkcwvjpqxy` (1840)
`enirsahtudlcgmwfbozkpjvqxy` (1863)
...
`enisratduhglcmwobfzkvpjqxy` (1955)

englisch (verschiedene Quellen):

`etaoinshrdlucmfwypvbgkqjxz` (1884)
`etoanirshdlcufmpywgbvkxjqz` (1893)
...
`etaoinsrhldcumfpgwybvkxjqz` (1982)

Inhalt

Asymmetrische Verschlüsselung

Die **asymmetrische Verschlüsselung** verwendet zwei unterschiedliche Schlüssel.

Es wird eine **öffentlicher** und ein **privater** Schlüssel verwendet, die zueinander komplementär sind.

Daten, die mit dem öffentlichen Schlüssel verschlüsselt werden (verschlüsseln von Nachrichten), können mit dem private Schlüssel entschlüsselt werden.

Daten, die mit dem privaten Schlüssel verschlüsselt werden (signieren von Nachrichten), können mit dem öffentlichen Schlüssel entschlüsselt werden .

Entscheidend ist aber, dass der private Schlüssel nicht aus dem öffentlichen Schlüssel abgeleitet werden kann.

Bei der asymmetrischen Verschlüsselung erzeugt jeder Teilnehmer ein Schlüsselpaar, behält den privaten Schlüssel und machten den öffentlichen Schlüssel den anderen Teilnehmern zugänglich.

Für den Austausch von verschlüsselten Nachrichten chiffriert der Sender die Daten mit dem öffentlichen Schlüssel des Empfängers und schickt sie diesem. Der Empfänger kann die Daten anschließend mit Hilfe seines privaten Schlüssels dechiffrieren.

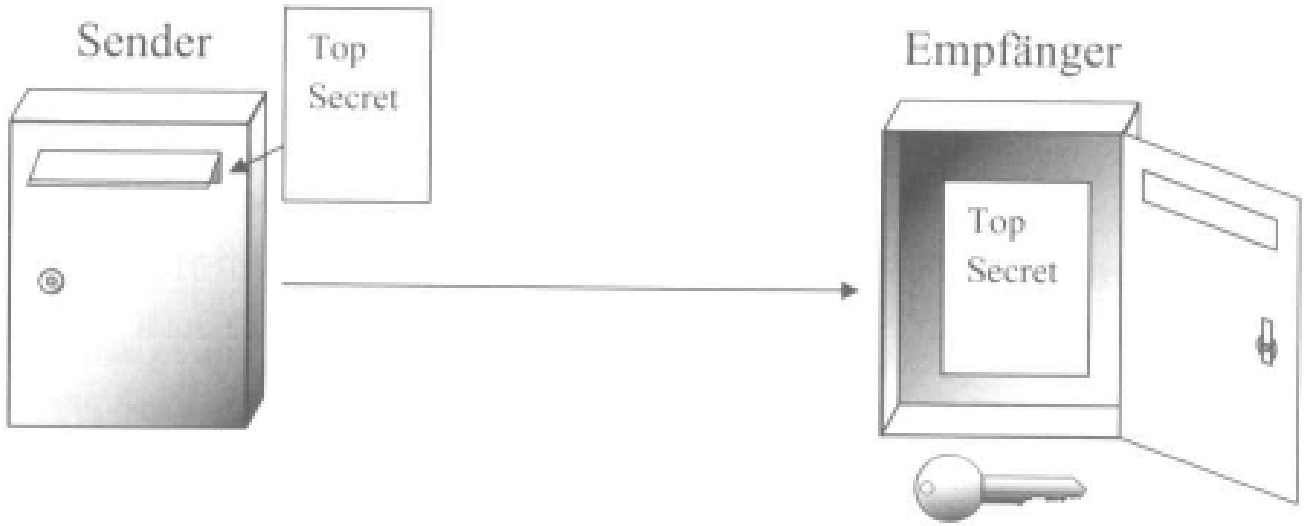
Anschauung

Man kann einem Empfänger eine verschlüsselte Nachricht schicken, ohne eine Geheiminformation zu besitzen.

Das kann man sich wie das Einwerfen einer Nachricht in einen Briefkasten vorstellen.

Jeder, der Zugang zum Briefkasten (dem öffentlichen Schlüssel) hat, kann eine Nachricht einwerfen (Verschlüsseln).

Nur der Empfänger kann mit seinem privaten Schlüssel den Briefkasten öffnen und die Nachricht entnehmen (Entschlüsseln).



Verschlüsselungsalgorithmus

Definition

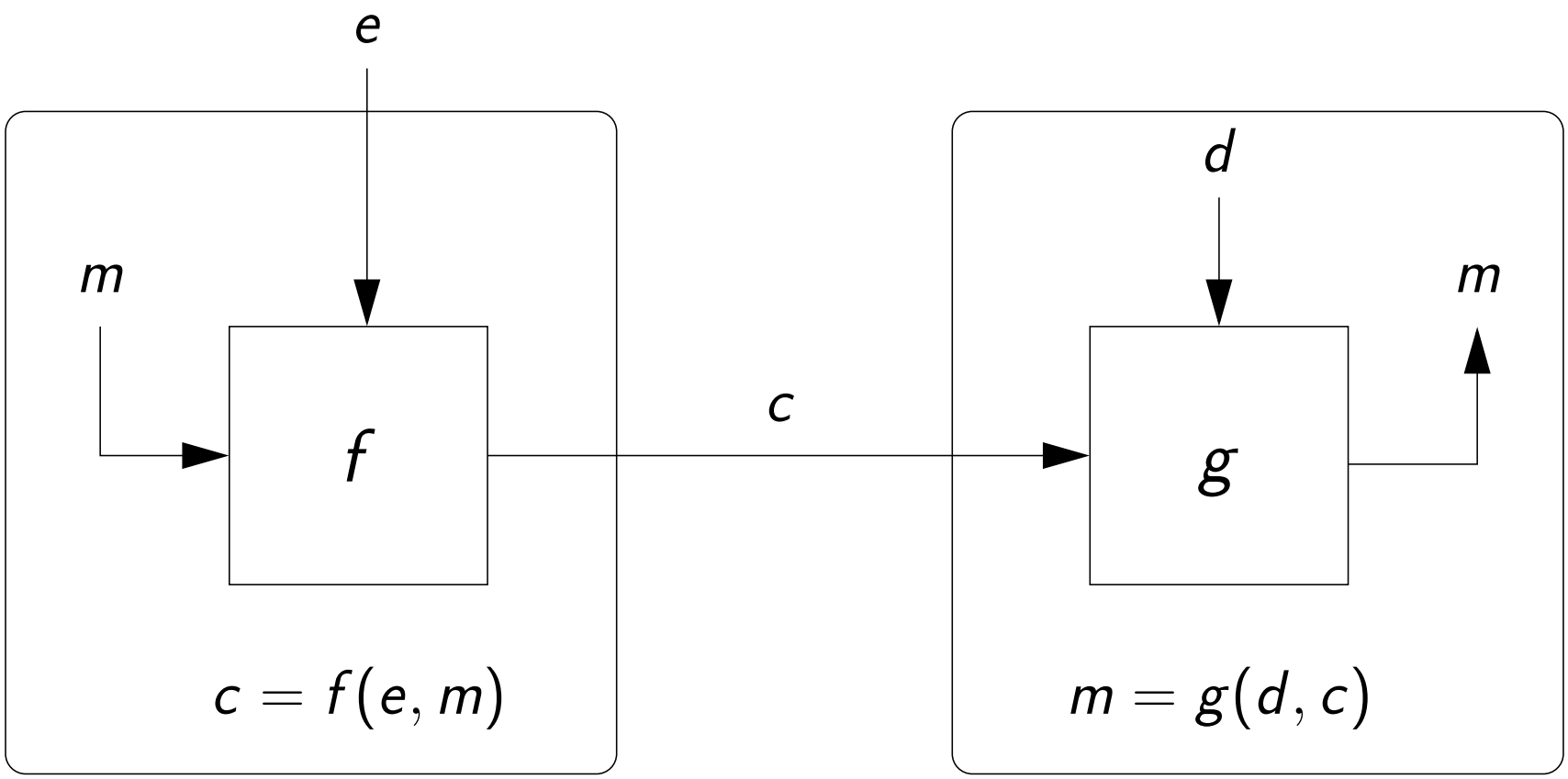
Ein **asymmetrischer Verschlüsselungsalgorithmus** besteht aus

- einer Funktion f mit zwei Eingabewerten, dem **öffentlichen Schlüssel** e und einem Text m, die Ausgabe ist der Text $f(e, m)$,
- einer Funktion g mit zwei Eingabewerten, dem **privaten Schlüssel** d und einem Text m, die Ausgabe ist der Text $g(d, m)$.

Für alle Texte m gelten die folgenden Beziehungen zwischen f und g.

$$g\big(d, f(e, m)\big) = m$$
$$f\big(e, g(d, m)\big) = m$$

Funktionsschema



Verschlüsseln
öffentlicher Schlüssel

Entschlüsseln
privater Schlüssel

Einwegfunktion

Eine **Einwegfunktion** ist eine Funktion, die einfach auszuführen, aber schwer (nur mit sehr großem Aufwand) zu invertieren ist.

Etwas formaler ist eine Einwegfunktion $f : X \to Y$ eine Abbildung für zwei Mengen X, Y, sodass Folgendes gilt.

- Für alle $x \in X$ ist $f(x)$ leicht zu berechnen.
- Für (fast) jedes $y \in Y$ ist es schwer ein Urbild $x \in X$, d.h. ein $x \in X$ mit $y = f(x)$, zu finden.

Einwegfunktionen spielen in der theoretischen und der praktischen Kryptographie eine entscheidende Rolle.

Beispiel

Die Funktion, die über Nachschlagen in einem gedruckten Telefonbuch einem Namen eine Telefonnummer zuordnet ist leicht auszuführen, da die Namen im Telefonbuch alphabetisch geordnet sind.

Allerdings ist die Umkehrung, die Zuordnung einer Telefonnummer zu einem Namen, mit Hilfe eine gedruckten Telefonbuchs ein sehr schwieriges Unterfangen.

Trapdoor-Einwegfunktion

Einwegfunktionen finden in der Kryptographie Verwendung, wenn alle Beteiligten die Funktion anwenden dürfen und kein Beteiligter die Umkehrfunktion kennt bzw. ermitteln kann, z.B. zur Integritätsprüfung von Daten.

Für die asymmetrische Verschlüsselung benötigt man ein erweitertes Konzept, da für Verschlüsseln und Entschlüsseln, sowohl Funktion als auch Umkehrfunktion benötigt werden.

Eine **Trapdoor-Einwegfunktion** ist eine Einwegfunktion, also eine eigentlich schwer zu invertierende Funktion, es sei denn man kennt die Zusatzinformation (die *trapdoor* zu Deutsch *Falltür*, im Deutschen oft *Hintertür*), mit deren Hilfe man die Funktion leicht invertieren kann.

Beispiel (1/3)

Ist $n \in \mathbb{N}$ (groß und) **keine** Primzahl, dann ist das Quadrieren modulo n

$$f : \mathbb{N} \to \{0, \ldots, n-1\}$$
$$x \mapsto x^2 \mod n$$

eine Einwegfunktion.

Seien p, q zwei (große) Primzahlen und $n = pq$, dann ist das Quadrieren modulo n eine Trapdoor-Einwegfunktion.

Die Trapdoor ist in diesem Fall die Faktorisierung von n, also die Faktoren p, q. Mit dieser Information ist das Invertieren einfach möglich.

Beispiel (2/3)

Allgemein kann man zeigen, dass für zwei (große) Primzahlen p, q und $n = pq$ die Potenzfunktion

$$f : \mathbb{N} \to \{0, \ldots, n-1\}$$
$$x \mapsto x^k \mod n$$

für beliebiges $k > 1$ eine Trapdoor-Einwegfunktion ist, mit Trapdoor p, q.

Beispiel (3/3)

Ein weitere Trapdoor-Einwegfunktion ist die Exponentialfunktion.

Für zwei (große) Primzahlen p, q und $n = pq$ ist

$$f : \mathbb{N} \times \mathbb{N} \to \{0, \ldots, n-1\}$$
$$f(m, x) = m^x \mod n$$

eine Trapdoor-Einwegfunktion, mit Trapdoor p, q.

RSA

Satz

Sei $n = pq$ das Produkt zweier Primzahlen $p, q \in \mathbb{N}$.
Dann gilt für alle natürlichen Zahl m, k mit $m < n$ folgende Gleichung.

$$m^{k(p-1)(q-1)+1} \mod n = m$$

Rechenregeln

Für natürliche Zahlen $x, i, j, n \in \mathbb{N}$ gilt Folgendes.

$$\left(x^i \mod n\right)^j \mod n = \left(x^i\right)^j \mod n = x^{i \cdot j} \mod n$$

RSA-Algorithmus

Der RSA-Algorithmus wurde 1978 von Rivest, Shamir and Adleman erfunden, der Algorithmus basiert auf folgendem Prinzip.

Sei $n = pq$ das Produkt zweier Primzahlen $p, q \in \mathbb{N}$.

Wähle bzw. berechne die Schlüssel $e, d \in \mathbb{N}$, sodass für ein $k \in \mathbb{N}$ gilt

$$e \cdot d = k(p-1)(q-1) + 1 \ .$$

Dann gilt das Folgende für jedes $m \in \mathbb{N}$ mit $m < n$.

$$\begin{aligned} \left(m^e \mod n\right)^d \mod n &= m^{e \cdot d} \mod n = m \\ \left(m^d \mod n\right)^e \mod n &= m^{e \cdot d} \mod n = m \end{aligned}$$

Somit sind die Funktionen f und g identisch,
$f(r, m) = g(r, m) = m^r \mod n$.