

# Matura

## NWES-SESD

Höhere Technische Bundeslehr- und Versuchsanstalt Anichstraße

---

Abteilung für Wirtschaftsingenieure/Betriebsinformatik

Ausgeführt im Schuljahr 2023/24 von:  
Gwercher

# Inhaltsverzeichnis

<b>I NWES</b>	<b>1</b>
<b>1 Bussysteme</b>	<b>2</b>
1.1 RS-232 (UART) . . . . .	3
1.2 CAN-Bus (Controll Area Network) . . . . .	6
1.3 I2C-Bus (IIC, I <sup>2</sup> C) . . . . .	8
<b>2 IPv4 (Internet Protocol Version 4)</b>	<b>13</b>
<b>3 Netzwerke im Alltag und Grundbegriffe</b>	<b>15</b>
3.1 Referenzmodell (OSI und TCP/IP) . . . . .	17
3.1.1 Layer 1 (Physical) . . . . .	18
3.1.2 Layer 2 (Data Link) . . . . .	20
3.1.3 Layer 3 (Network) . . . . .	24
3.1.4 Layer 4 (Transport) . . . . .	28
3.1.5 Layer 5, 6, 7 (Session, Presentation, Application) . . . . .	32
3.2 VLANs . . . . .	36
<b>4 Routing</b>	<b>39</b>
<b>5 IPv6</b>	<b>43</b>
<b>6 WLAN (Wireless Local Area Network)</b>	<b>47</b>
6.1 Wifi (802.11) . . . . .	48
<b>7 Network Security</b>	<b>53</b>
7.1 Firewall . . . . .	54
7.2 IDS & IPS . . . . .	55
7.3 Honeypot . . . . .	55
7.4 VPN . . . . .	56
7.5 ESA (Email Security Appliance) & WSA (Web Security Appliance) . . . . .	56
7.6 IPsec . . . . .	57
<b>8 Hashfunktionen</b>	<b>58</b>
8.1 Kryptographische Hashfunktionen . . . . .	58
8.2 Passwörter . . . . .	60
8.3 AAA . . . . .	61
8.4 PKI (Public Key Infrastruktur) . . . . .	61

8.5 Zusammenfassung . . . . .	62
8.6 HTTPS . . . . .	62
<b>II SESD</b>	<b>64</b>
<b>9 Aspekte der Datensicherheit</b>	<b>65</b>
<b>10 Kryptologie</b>	<b>67</b>
10.1 Symmetrische und Asymmetrische Kryptosysteme . . . . .	70
10.2 Angriffsarten . . . . .	73
10.2.1 Ciphertext-Only Angriff . . . . .	73
10.2.2 Kown-Plaintext Angriff . . . . .	73
10.2.3 Chosen-Plaintext Angriff . . . . .	74
10.2.4 Brute-Force Angriff . . . . .	74
10.2.5 Wörterbuchangriff . . . . .	74
10.2.6 Replay Angriff . . . . .	74
10.2.7 Man-in-the-Middle Angriff . . . . .	75
10.2.8 Verkehrsflussanalysen . . . . .	75
10.2.9 Harvest-now-decrypt-later Angriff . . . . .	75
10.2.10 Seitenkanalangriffe . . . . .	75
10.2.11 Angriffe auf die BenutzerInnen . . . . .	76
<b>11 Mathematische Grundlagen der Kryptographie</b>	<b>78</b>
11.1 Modulo . . . . .	78
11.2 Kongruenz modulo $m$ . . . . .	78
11.3 Uhrenarithmetik . . . . .	79
11.4 Restklassenring $\mathbb{Z}_m$ . . . . .	80
11.5 Berechnung des ggT . . . . .	81
11.5.1 Primfaktorzerlegung beider Zahlen . . . . .	81
11.5.2 Euklidischer Algorithmus . . . . .	81
11.5.3 Erweiterter euklidischer Algorithmus . . . . .	82
11.6 Eulersche Phi-Funktion . . . . .	82
11.7 Einweg- und Falltürfunktionen . . . . .	83
11.8 Modulares Potenzieren und diskreter Logarithmus . . . . .	85
11.8.1 Modulares Potenzieren . . . . .	85
11.8.2 Diskreter Logarithmus . . . . .	85
11.8.3 Aufwand des modularen Potenzieren . . . . .	86
11.8.4 Square-and-Multiply Algorithmus . . . . .	86
11.8.5 Aufwand der Berechnung des diskreten Logarithmus . . . . .	86
<b>12 Diffie-Hellman Schlüsselaustausch</b>	<b>87</b>
12.1 Ablauf . . . . .	87
12.2 Allgemein . . . . .	88
12.3 Man-in-the-Middle . . . . .	88

<b>13 RSA</b>	<b>89</b>
13.1 Ablauf . . . . .	89
13.2 Allgemein . . . . .	90
<b>14 One-Time-Pad</b>	<b>91</b>
14.1 Caesar-Verfahren . . . . .	91
14.2 Monoalphabetische Substitution . . . . .	91
14.3 Vigenère Verfahren . . . . .	91
14.4 Vernam Verfahren . . . . .	92
14.5 One-Time-Pad . . . . .	92
<b>15 Design moderner Blockchiffren</b>	<b>93</b>
15.1 Data Encryption Standard (DES) . . . . .	94
15.1.1 Schlüsselraum . . . . .	94
15.1.2 Blockgröße . . . . .	94
15.2 Triple-DES, 3DES, DESede . . . . .	94
15.3 Advanced Encryption Standard (AES) . . . . .	95
<b>I Quellcodeverzeichnis</b>	<b>III</b>

# Teil I

## NWES

# 1 Bussysteme

Bei einem Bussystem teilen sich mehrere Teilnehmer (z.B. Arduinos) einen Übertragungsweg.

## Grundlegende Eigenschaften

- Übertragungsart
  - Seriell: Zeichen werden nacheinander übertragen
  - Parallel: Zeichen werden gleichzeitig übertragen
- Topologie
  - Punkt-zu-Punkt
  - Linientopologie
  - Sterntopologie
  - Ringtopologie
  - Baumtopologie

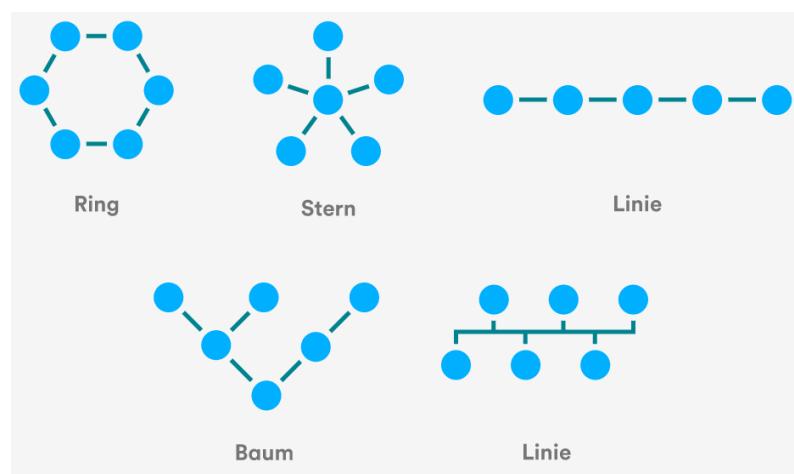


Abbildung 1.1: Bustopologiearten

- Priorisierung, Kollisionsvermeidung
- Synchronisierung
  - synchron: eigene Taktleitung
  - asynchron: keine Taktleitung, fix vorgegebene Übertragungsrate  
 Einheit:  $1 \text{ baud} = 1 \frac{\text{bit}}{\text{s}}$
- Fehlererkennung

**Konkrete Beispiele:** RS-232, CAN, I2C, SPI, USB, Bluetooth

## 1.1 RS-232 (UART)

RS-232 kann zwei Geräte miteinander verbinden.

- Topologie: Punkt-zu-Punkt
- Priorisierung, Kollisionsvermeidung: nicht nötig (2 Kabel: 1x Rx, 1x Tx)
- Synchronisierung: asynchron
- Übertragungsart: Seriell     3 bis 15V '1'  
                               -3 bis -15V '0'  
     Achtung: Alle Arduinos nutzen den gleichen GND-Pin!
- Fehlererkennung: eventuell mit einem Parity-Bit

### Konkrete Umsetzung

Beim RS-232 bestehen die Daten aus einem Startbit, 5-8 Datenbits, dann eventuell ein Parity-Bit und am Ende 1 bis 2 Stopbits

Bsp: 9600 8E1

9600	8	E	1
Übertragungswert	Anzahl der Datenbits	ein Even-Parity-Bit	Anzahl der Stopppbits

'a' ASCII, 97 → 01100001

0	01100001	1	1
Start	Datenbits	Parity-bit	Stopp-Bit

Das Parity-Bit kann

- 0 ... Odd

- E ... Even

- N ... None

sein

Bsp 2: 96000 801

'S', 83 → 1010011 0 1010011 1 1  
 'E', 69 → 01000101 0 1000101 0 1

## UART am Arduino

Der Arduino besitzt ein UART-Bauteil, welches für die aktuelle Kommunikation genutzt werden kann. UART-Bauteil sendet immer mit 8 Datenbits und immer mit einem Stoppenbit.

### Serial Funktionen

Serial.available() ... wandelt eingegebene Zeichen in ASCII-Zeichen/Zahlen um  
 Beispiel mit Ausgabe: HELLO → 72 69 76 76 79 10

Serial.print() ... gibt Daten im Serial Monitor aus

Serial.read() ... gleich wie Serial.available()

Serial.write() ... ähnlich wie Serial.print() und Serial.available() jedoch wird hier ASCII-Zahlen in Buchstaben umgewandelt.

z.B. Serial.write(72) → H  
 Serial.write(69) → E

```

1 // LED-Streifen oder 5 LEDs
2 // Zahleingaben -> so viele LEDs leuchten
3
4 // S E N D E R
5 #define BUTTON 4
6
7 int counter = 0;
8
9 boolean aktuell = false;
10 boolean letzte = false;
11 boolean steigendeF = false;
12
13 unsigned long zeit = 0;
14 unsigned long t = 100;
15 unsigned long wait2 = 0;
16
17 void setup() {
18     pinMode(BUTTON, INPUT);

```

```

19     Serial.begin(9600);
20 }
21
22 void loop() {
23     aktuell = digitalRead(BUTTON);
24
25     steigendeF = aktuell && !letzte;
26
27     if (steigendeF && millis() >= (zeit + t)) {
28         zeit = millis();
29         counter = (counter + 1) % 6;
30     }
31
32     if (millis() >= (5000 + wait2) ) {
33         Serial.print(counter); // zahl an anderen arduino senden
34         counter = 0;
35         wait2 = millis();
36     }
37
38     letzte = aktuell;
39 }
40 }
```

Quellcode 1.1: UART Sender

```

1 // LED-Streifen oder 5 LEDs
2 // Zahleingaben -> so viele LEDs leuchten
3
4 // R E C E I V E R
5 #define BUTTON 2
6
7 int counter = 0;
8
9 boolean aktuell = false;
10 boolean letzte = false;
11 boolean steigendeF = false;
12
13 unsigned long zeit = 0;
14 unsigned long t = 100;
15 unsigned long wait2 = 0;
16 unsigned long printWait = 0;
17
18 void setup() {
19     pinMode(5, OUTPUT);
20     pinMode(6, OUTPUT);
21     pinMode(7, OUTPUT);
22     pinMode(8, OUTPUT);
23     pinMode(9, OUTPUT);
24     pinMode(BUTTON, INPUT);
25     Serial.begin(9600);
26 }
```

```

27
28 void loop() {
29   if (Serial.available() > 0) {
30     counter = Serial.read() - 48;    // zahl von anderen arduino
31     auslesen
32     Serial.println(counter);
33     LEDS(counter);
34   }
35 }
36
37 void LEDS(int x) {
38   x = x + 4;
39   for (int i = 5; i <= 9; i++) {
40     digitalWrite(i, LOW);
41   }
42
43   for (int i = 5; i <= x; i++) {
44     digitalWrite(i, HIGH);
45   }
46 }
```

Quellcode 1.2: UART Receiver

## 1.2 CAN-Bus (Controll Area Network)

Der CAN-Bus ist ein serieller Bus der speziell für die Automobilindustrie entwickelt wurde.

### Aktuelle Anwendungen

- Autos (Vernetzung von Steuergeräten und Sensoren)
- Flugzeuge
- Raumfahrt
- Medizintechnik
- ...

Der CAN-Bus arbeitet nach dem Multi-Master-Prinzip und bildet eine Linientopologie.

## Aufbau

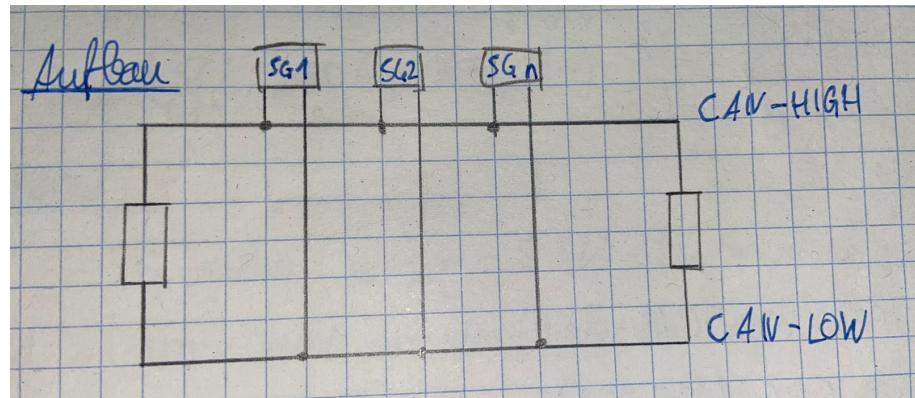


Abbildung 1.2: CAN-Bus Aufbau

Der CAN-Bus ist ein asynchroner Bus

Datenwert	Startbit	1	✓	Startbit	1
Message ID	11	←		Daten	8
Steuerbits	7	✓		(Parity	1)
Datenbits	0-64	✓		Stopp	1
Fehlererkennung	15	←			
Steuerbits	3	✓			
Stoppbits	7	✓			

## Message ID

Dort steht um welche Art von Nachricht es sich handelt und wie wichtig die Nachricht ist (z.B. Wasser, Öl, Airbag). Desto kleiner die Message ID ist, desto wichtiger ist die Nachricht. Jedes Sg kann nur eine Message ID aussenden (Priorisierung).

## Kollisionsvermeidung CSMA/CR

SG1: M-ID = 000100

SG2: M-ID = 011011

SG3: M-ID = 001101

0 ... dominant

1 ... rezessiv

## Sicherheit

- zwei Leitungen (CAN-LOW, CAN-HIGH)
- Fehlererkennung mit CRC (15 Bit)
- Stuffbit (Stopfbit) Bei fünf gleichen Bits wird ein anderes eingefügt  
0000000  
00000100

## Zusammenfassung

- seriell
- Message ID mit CSMA/CR
- Linientopologie
- asynchron
- Stuffbit, CRC, 2 Leitungen

## 1.3 I2C-Bus (IIC, I<sup>2</sup>C)

Der I2C-Bus wurde Anfang 80er Jahre von der Firma Philips entwickelt.

Der I2C-Bus ...

- ... ist seriell
- ... hat 2 Leiter: SCL & SDA (Serial Clock, Serial Data)
- ... ist synchron
- ... funktioniert nach Master-Slave-Prinzip.  
Es gibt einen Master (es wären mehrere möglich) → keine Kollisionen, Priorisierung unnötig durch Master

## Aufbau

- Linientopologie
- 2 Leitungen mit Pullup-Wiederstände → Ruhe zustand HIGH (SCL Takt, SDA Daten; A5, A4)

## Ablauf

- es werden immer 8 Bit Datenwerte gesendet
- um Daten zu senden muss der Pegel der Datenleitung stabil sein (0 oder 1), falls die Takteleitung auf HIGH ist. Sobald die Takteleitung auf LOW ist, kann die Datenleitung das Bit setzen.

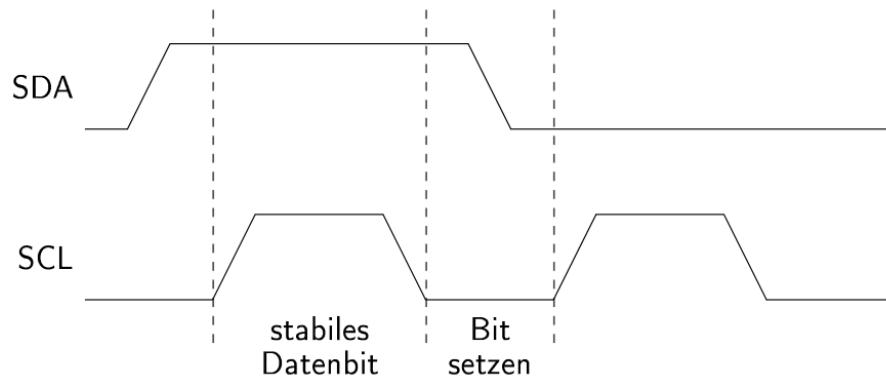


Abbildung 1.3: I2C-Bus Bitsetzung

- Steuersignale
  - Startsignal (fallende Flanke, während der Takt auf HIGH ist)
  - Stoppsignal (steigende Flanke, während der Takt auf HIGH ist)

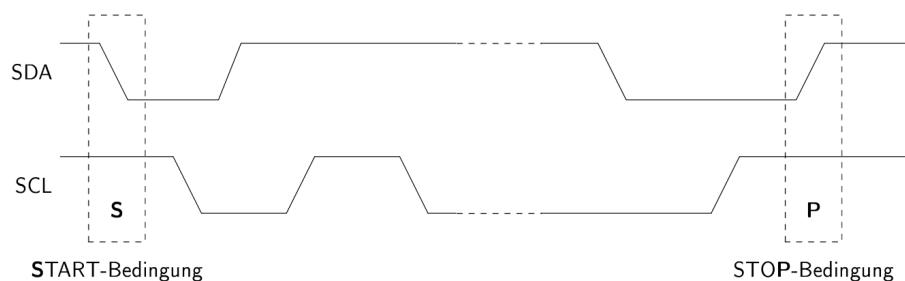


Abbildung 1.4: I2C-Bus Steuersignale

## Adressierung

- 7 Bit-Adressen für die Slaves  
→ 128 mögliche Teilnehmer (eig. 112)
- 0000000 → general call address (Broadcastadresse)



- Das 8te Bit sagt ob der Master lesen oder schreiben soll

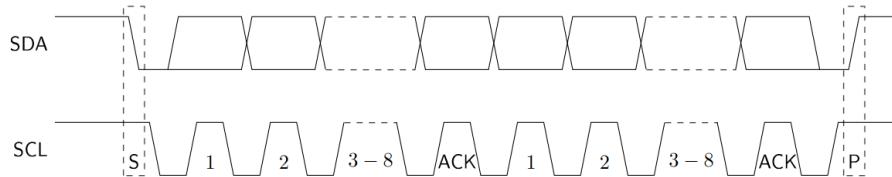


Abbildung 1.5: I2C Ablauf

## Zusammenfassung

- seriell
- Linientopologie
- Master-Slave → unnötig (bei einem Master)
- synchron
- ACK, Takt

Master	Slave
Wire.beginTransmission(address); Wire.endTransmission(); Wire.read(); Wire.write(); Wire.available();	Wire.begin(address); Wire.onRecieve(); Wire.onRequest();

```
1 #include <Wire.h>
2
3 void setup() {
4     Wire.begin(); // join I2C bus (address optional for master)
5 }
6
7 byte x = 0;
8
9 void loop() {
10    Wire.beginTransmission(8); // transmit to device #8
11    Wire.write("x is "); // sends five bytes
12    Wire.write(x); // sends one byte
13    Wire.endTransmission(); // stop transmitting
14
15    x++;
16    delay(500);
17 }
```

Quellcode 1.3: I2C #1 Master (Writer)

```

1 #include <Wire.h>
2
3 void setup() {
4     Wire.begin(8);           // join I2C bus with address #8
5     Wire.onReceive(receiveEvent); // register event
6     Serial.begin(9600);       // start serial for output
7 }
8
9 void loop() {
10    delay(100);
11 }
12
13 // function that executes whenever data is received from master
14 // this function is registered as an event, see setup()
15 void receiveEvent(int howMany) {
16     while (1 < Wire.available()) { // loop through all but the last
17         char c = Wire.read(); // receive byte as a character
18         Serial.print(c);      // print the character
19     }
20     int x = Wire.read();        // receive byte as an integer
21     Serial.println(x);        // print the integer
22 }
```

Quellcode 1.4: I2C #1 Slave (Receiver)

```

1 #include <Wire.h>
2
3 void setup() {
4     Wire.begin();           // join I2C bus (address optional for master
5     Serial.begin(9600);    // start serial for output
6 }
7
8 void loop() {
9     Wire.requestFrom(8, 6); // request 6 bytes from slave device
#8
10
11    while (Wire.available()) { // slave may send less than requested
12        char c = Wire.read(); // receive a byte as character
13        Serial.print(c);      // print the character
14    }
15
16    delay(500);
17 }
```

Quellcode 1.5: I2C #2 Master (Reader)

```

1 #include <Wire.h>
2
3 void setup() {
4     Wire.begin(8);          // join I2C bus with address #8
```

```
5     Wire.onRequest(requestEvent); // register event
6 }
7
8 void loop() {
9     delay(100);
10}
11
12 // function that executes whenever data is requested by master
13 // this function is registered as an event, see setup()
14 void requestEvent() {
15     Wire.write(" hello "); // respond with message of 6 bytes
16     // as expected by master
17 }
```

Quellcode 1.6: I2C #2 Slave (Sender)

## 2 IPv4 (Internet Protocol Version 4)

Eine IPv4-Adresse ist eine 32-Bit Zahl. Es gibt also  $2^{32} \approx 4,3$  Milliarden IPv4-Adressen.

**Bsp:**

11000000	10101000	00001010	00001010
192	168	10	10
→ 192.168.10.10			

### Schreibweise:

Die IPv4-Adresse wird in Dotted Decimal Notation geschrieben. Die IP-Adresse wird in 8-Bit Blöcke (Oktetten) geteilt, dezimal übersetzt und durch Punkte getrennt.

### Verwendung

Jedes Gerät soll durch eine Adresse (IP-Adresse) eindeutig identifiziert werden. Zusätzlich sollten auch Gruppen (Netze) von Computern erstellt werden (mit Subnetzmasken). Ein Gerät mit IP-Adresse nennt man Host.

### Subnetmask

Ist eine 32-Bit Zahl, die in Dotted Decimal Notation beschrieben wird. Es kommen zuerst alles Einsen und nach der ersten Null nur noch Nullen.

### Typische Subnetmasken

	Präfix	Hosts
255.0.0.0	8	$2^{24} - 2 = 16.777.214$
255.255.0.0	16	$2^{16} - 2 = 65.534$
255.255.255.0	24	$2^8 - 2 = 254$

**Bsp:**

Telefonnummer	IP-Adresse
+43 664 123456	172.16. 20.25
Netz einzigartig	255.255. 0.0

Netzteil      Hostteil

Die Subnetzmaske trennt die IP-Adresse in Netzteil und Hostteil. IP-Adresse und Subnetzmaske gehören immer zusammen.

1. 2 IP-Adressen im gleichen Netz

10.10.226.120 / 24  
 10.10.226.80 / 24

2. 2 IP-Adressen nicht im gleichen Netz

11.40.30.124 / 24  
 14.8.50.100 / 24

3. Anzahl der Hosts

$2^8 - 2$   
 10.10.226.0 (Netzadresse), 10.10.226.255 (Broadcastadresse)

192.168.20.100 / 8

Netz: 192.0.0.0

Broad: 192.255.255.255

### 3 Netzwerke im Alltag und Grundbegriffe

#### Netzwerk Komponenten

- Endgeräte (PC, Handy, Uhr, TV, Server,...)
- Intermediary Devices (Router, Repeater, Switch, Hub, Access Point)
- Übertragungsmedien (Drahtlos, Kupfer, Glasfaser)

#### Host-Aufgaben

- Client-Server-Modell
- Peer-to-Peer Modell
  - + Komplexität
  - + Leichter zum Aufsetzen
  - Security
  - Erweiterbarkeit

#### Netzwerk Dokumentation

- Physische Topologie (Räume, ...)
- Logische Topologie (Netze, ...)

#### Netzwerke nach Größe

- SOHO ... small office home office
- LAN ... local area network
- MAN ... metropolitan area network
- WAN ... wide area network
- Internet

## Netzwerke nach Funktion

- SAN ... storage area network
- Intranet, Extranet

## Internetzugang

- Kabel (Glasfaser)
- DSL / Dial Up
- Mobilfunknetz
- Satellit

## Trends

- Video / Streaming
- Cloud
- Drahtlos (5G)
- BYOD (bring your own device)
- Online Collaboration
- Powerline Method

## Netzwerkarchitektur

- Quality of Service QoS
- Erweiterbarkeit
- Security
- Fehlertoleranz

## Security

- Ransomware
- DoS / DDoS
- Virus, Wurm, Trojaner

- Social Engineering
- Zero-Day-Attack

### 3.1 Referenzmodell (OSI und TCP/IP)

OSI		Protokolle	TCP/IP
7	Application Layer	HTTPS, FTP, Telnet, SSH	Application Layer
6	Presentation Layer	POP, SMTP, IMAP	
5	Session Layer	DHCP, NTP, DNS	
4	Transport Layer	TCP, UDP	
3	Netzwerk Layer	IP, ICMP; OSPF, BGP, RIP	
2	Data Link Layer	Wifi, Ethernet, ARP	
1	Physical Layer		Network Access Layer

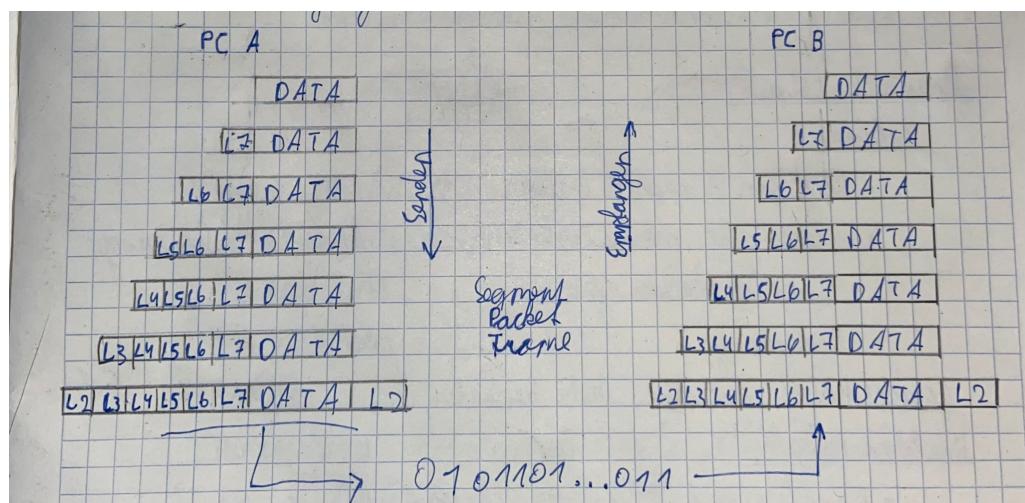


Abbildung 3.1: OSI-Modell Datenübertragung

**Layer 1 (Physical):** Bits übertragen

**Layer 2 (Data Link):** Lokale Adressierung, Fehlererkennung

**Layer 3 (Network):** Globale Adressierung, Routing

**Layer 4 (Transport):** Datenpaketaufteilung, Segmentierung, Datenfluss steuern

**Layer 5 (Session):** Session Verwalten, Verschlüsselung

**Layer 6 (Presentation):** Darstellung der Daten

**Layer 7 (Application):** Funktionen für die Application

### 3.1.1 Layer 1 (Physical)

#### Aufgaben

- Bits von A nach B bringen
- elektrische, mechanische oder andere physische Verbindung zwischen zwei Geräten
- Kodierung

**Geräte:** Kabel, Antenne, Hub, Repeater,...

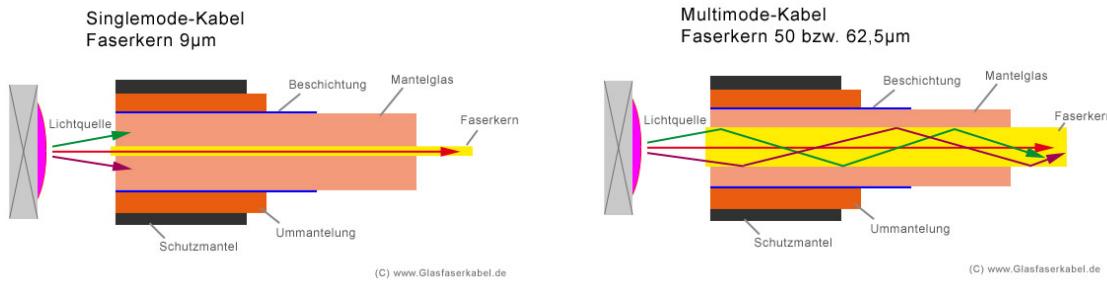
#### Wichtige Begriffe

- Bandbreite (bits/s → theoretisch)
- Durchsatz (bits/s → praktisch)
- Latenz (Dauer der Daten von A bis B in ms)

#### Typische Medien

- Kupferkabel (Twisted-Pair-Kabel)
  - + Günstig ≈ Distanz (ca 100m)
  - + einfache Handhabung ≈ Geschwindigkeit
    - Interferenzen (Störungen)
  - Straight Through (beide Enden gleich)
  - Crossover (verschiedene Enden)

(durch Auto MDIX werden Enden automatisch konfiguriert)
- Koaxialkabel
- Glasfaserkabel
  - Arten: Single-Mode (Senden Laser, Reichweite 1-10km)
  - Multi-Mode (Senden LED, Reichweite ca 600m)



(a) Single-Mode

(b) Multi-Mode

Abbildung 3.2: Glasfaserkabelarten

- |                                        |                         |
|----------------------------------------|-------------------------|
| + Speed<br>+ Reichweite<br>+ Störungen | - Teuer<br>- Handhabung |
|----------------------------------------|-------------------------|

- Drahtlos

Übertragung: elektromagnetische Wellen über Luft

- |                                                                                                       |
|-------------------------------------------------------------------------------------------------------|
| + Flexibel<br><br>- Störungen<br>- Shared Medium<br>- Reichweite (ca 100m), Hindernisse<br>- Security |
|-------------------------------------------------------------------------------------------------------|

### 3.1.2 Layer 2 (Data Link)

#### Aufgaben

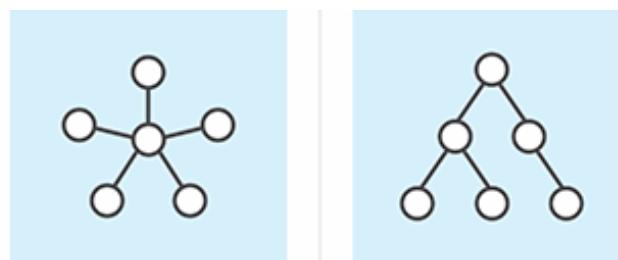
- lokale Adressierung
- Fehlererkennung
- Zugang zum Medium herstellen
- Kommunikation mit Layer 3

**Geräte:** Netzwerkkarte, Switch, Bridge,...

**Standards:** Wifi (802.11), Ethernet (802.2, 802.3)

#### Topologie

- Sterntopologie
- Baumtopologie
- Punkt-zu-Punkt



**Stern**

Verfügt über ein zentrales Gerät, das Daten an andere Knoten im System überträgt.

**Baum**

Verbindet Geräte in einer Struktur, die einem Baum ähnelt, bei dem übergeordnete Knoten mit untergeordneten Knoten verbunden sind.

Abbildung 3.3: Baum- und Stern-topologie

## Ethernet

L2 (Header)										L3-L7	DATA	L2 (Trailer)							
Preamble 7 Byte	Start Frame Delimiter 1 Byte	Dest MAC 6 Byte	Source MAC 6 Byte	VLANs 4 Byte	802.1q 2 Byte	Type/Length 1514-1522 Byte													

Abbildung 3.4: Ethernet Frame

## MAC-Adresse

Die MAC-Adresse ist eine 48-Bit Zahl und wird in hexadecimal dargestellt.

Bsp:

Hersteller für den Hersteller einzigartig  
DC F5 05 | 17 9A 69

Jede Netzwerkkarte besitzt eine weltweit einzigartige (theoretisch) MAC-Adresse.

## Type

Kodierung für Layer 3

0x800 → IP

0x806 → ARP

## Fehlerkennung

Frame Checksum (CRC)

Polynomdivision mit einem Polynom von Grad 32

## Funktion eines Switches

Der Switch baut mit der Source-MAC seine MAC-Tabelle auf. Dort steht zu jeder MAC-Adresse der passende Port. Falls die MAC-Adresse schon eingetragen ist, wird ein Timer aktualisiert. Sollte es noch keinen Eintrag geben wird er hinzugefügt und bleibt dort eine gewisse Zeit (5 Minuten) bevor er gelöscht wird. Der Switch vergleicht die Destination-MAC mit seiner MAC-Tabelle. Falls der Switch keinen Eintrag findet sendet er an alle Ports (Flooding, Unknown Unicast). Sonst sendet er an den Port, wo er den Frame bekommen hat.

Layer 2 Broadcast Adresse: FF:FF:FF:FF:FF:FF

## L2, L3 Adressierung

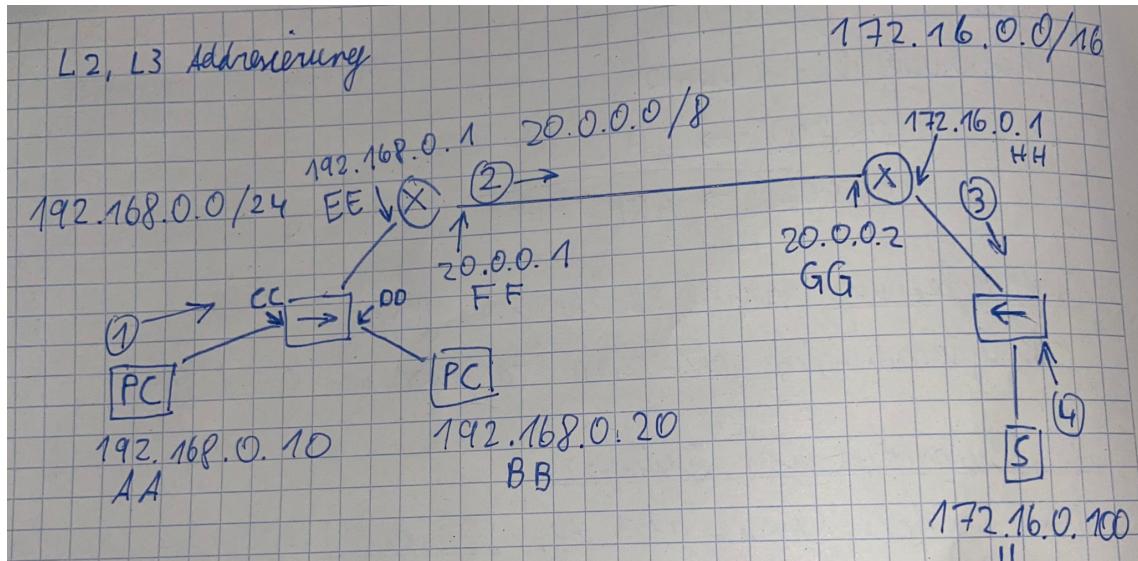


Abbildung 3.5: Layer 2 & 3 Adressierung

	Source MAC	Destination MAC	Source IP	Destination IP
1	AA	EE	192.168.0.10	172.16.0.100
2	FF	GG	192.168.0.10	172.16.0.100
3	HH	II	192.168.0.10	172.16.0.100
4	II	HH	172.16.0.100	192.168.0.10

## ARP (Address Resolution Protocol)

Nutzt ein Host um zu einer gegebenen IP-Adresse die passende MAC-Adresse zu finden

### ARP-Request (Broadcast)

Source MAC: eigene MAC-Adresse

Destination MAC: FF-FF-FF-FF-FF-FF

Type: 0x806 Danach ARP-Header (IP, MAC, Protokoll)

### ARP-Reply

Unicast (auch als Broadcast möglich)

Source MAC: eigene MAC-Adresse (gesucht)

Destination MAC: MAC-Adresse (Anfrage)

Type: 0x806  
 Danach ARP-Header

### ARP-Cache

Die Einträge werden im ARP-Cache gespeichert (ca 5 min)  
 IP MAC Time

### ARP-Spoofing

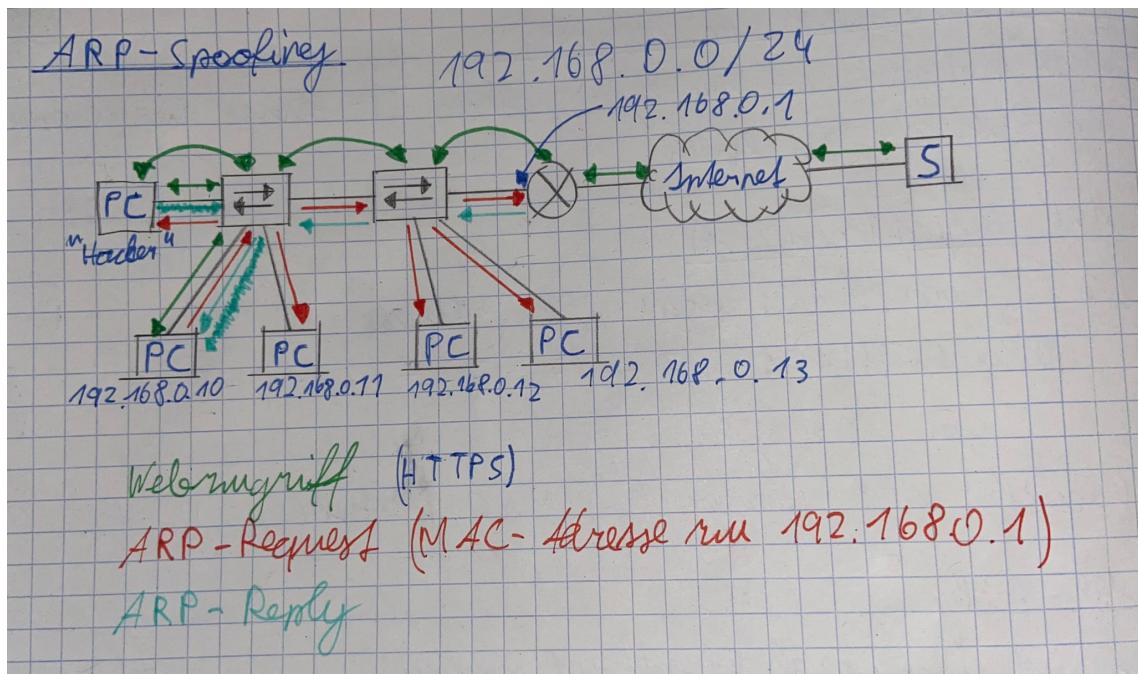


Abbildung 3.6: ARP-Spoofing

### 3.1.3 Layer 3 (Network)

#### Aufgaben

- Routing
- Globale Adressierung
- Kommunikation mit L2 & L4

**Protokolle:** IPv4, IPv6, ICMP, RIP, OSPF, EIGRP, IS-IS, BGP

#### IPv4

Eigenschaften von IP

- Verbindungslos
- Best Effort
- Medium unabhängig

**IP-Header (8.2.2)** Wichtige Felder: Source & Destination IP, Time-to-Live

#### Kommunikationsart

- Unicast (IP des Host)
- Multicast (224.0.0.0 - 239.255.255.255)
- Broadcast (letzte IP im Netz, 255.255.255.255)

#### Spezielle IP-Adressen

- 127.0.0.0 / 8 ... localhost
- 10.0.0.0 / 8
- 172.16.0.0 / 12
- 192.168.0.0 / 16 ... private IP-Adressen (NAT)
- 169.254.0.0 / 16 ... APIPA
- 192.0.2.0 / 24 ... Testnetz

**Fazit:** Zu wenig IPv4-Adressen!

## Deshalb

- VLSM (variable length subnet mask)
- NAT
- IPv6

## Classful Addressing (uralt)

Das erste Oktett bestimmt die Subnetzmaske (/8, /16, /24)

Klasse A	0-127	(0...)	/8
Klasse B	128-191	(10...)	/16
Klasse C	192-223	(110...)	/24
Klasse D	224-239	(1110...)	Multicast
Klasse E	240-255	(11110...)	für spätere Verwendung

## Classless Addressing (veraltet!)

Die Subnetzmasken /8, /16, /24 können beliebig verwendet werden

## CIDR (Classless Inter-Domain Routing)

Es können beliebige Subnetzmasken (z.B. /25, /26, ...) verwendet werden. Alle Subnetze werden gleich groß.

## VLSM (variable length subnet mask)

Alle Subnetzmasken können beliebig verwendet werden. Die Netzte dürfen sich nicht überschneiden.

## Subnetzmasken

Präfix Notation	Dotted Decimal Notation	Hosts	Subnetz von /24
/25	255.255.255.128	$2^7 - 2 = 126$	2
/26	255.255.255.192	$2^6 - 2 = 62$	4
/27	255.255.255.224	$2^5 - 2 = 30$	8
/28	255.255.255.240	$2^4 - 2 = 14$	16
/29	255.255.255.248	$2^3 - 2 = 6$	32
/30	255.255.255.252	$2^2 - 2 = 2$	64
/31	255.255.255.254	$2^1 - 2 = 0$	für spezielle Anwendung
/20	255.255.240.0	$2^{12} - 2 = 4.094$	/

### Bsp 1 (CIDR):

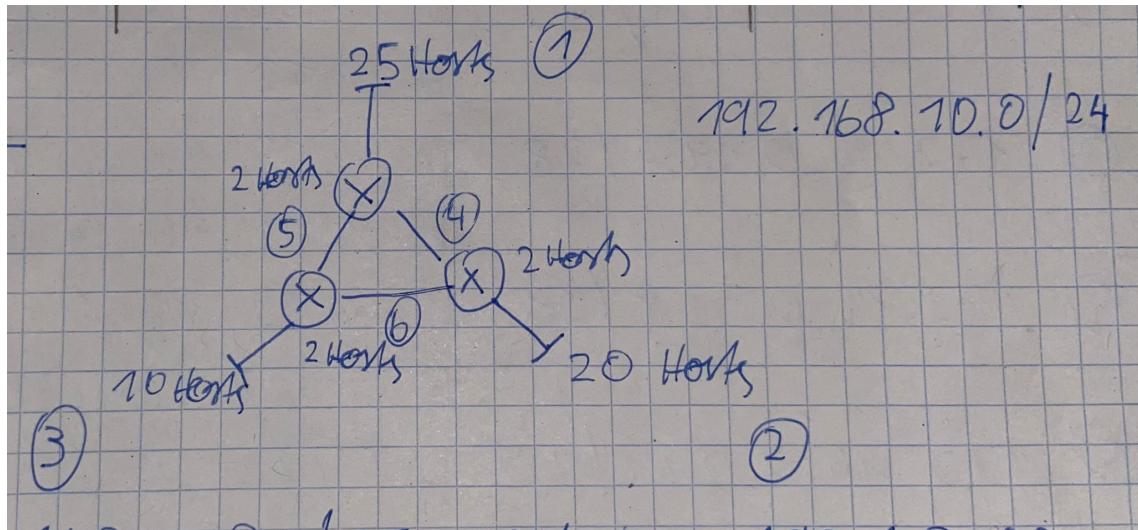


Abbildung 3.7: CIDR Beispiel

1	192.168.10.0 / 27	Netzadresse	192.168.100.0
		Broadcast	192.168.100.31
2	192.168.10.32 / 27	Netzadresse	192.168.100.32
		Broadcast	192.168.100.63
3	192.168.10.64 / 27	Netzadresse	192.168.100.64
		Broadcast	192.168.100.95
4	192.168.10.96 / 27	Netzadresse	192.168.100.96
		Broadcast	192.168.100.127
5	192.168.10.128 / 27	Netzadresse	192.168.100.128
		Broadcast	192.168.100.159
6	192.168.10.160 / 27	Netzadresse	192.168.100.160
		Broadcast	192.168.100.191

### Bsp 2 (VLSM):

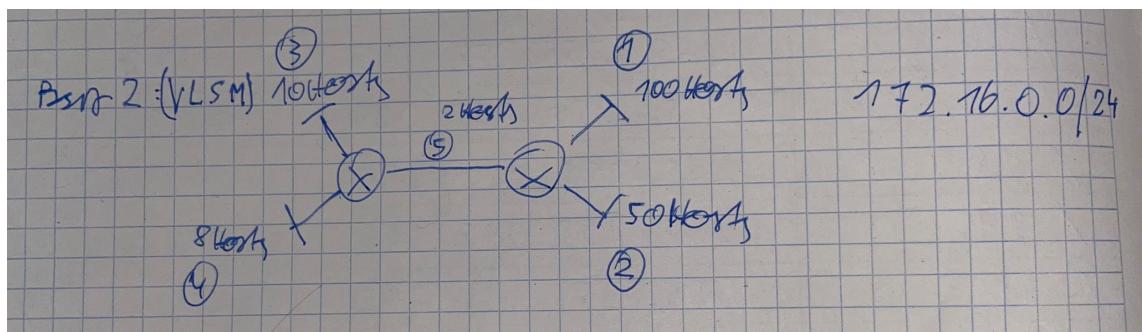


Abbildung 3.8: VLSM Beispiel

1	172.16.0.0 / 25	Netzadresse	172.16.0.0
		Broadcast	172.16.0.127
2	172.16.0.128 / 26	Netzadresse	172.16.0.128
		Broadcast	172.16.0.191
3	172.16.0.192 / 28	Netzadresse	172.16.0.192
		Broadcast	172.16.0.207
4	172.16.0.208 / 28	Netzadresse	172.16.0.208
		Broadcast	172.16.0.223
5	172.16.0.224 / 30	Netzadresse	172.16.0.224
		Broadcast	172.16.0.227

(PT: 10.4.3, 11.5.5, 11.9.3, 11.10.1)

### 3.1.4 Layer 4 (Transport)

#### Aufgaben

- Anwendungen identifizieren
- Segmentierung
- ev. Flusskontrolle, Verbindungsau- & abbau
- Kommunikation mit L3 & L5

#### Protokolle

- TCP (Transmission Control Protocol)
- UDP (User Datagram Protocol)

TCP	UDP
Anwendungen identifizieren (Ports) Segmentierung Verbindungen auf- bzw abbauen Segmente ordnen wiederholtes Senden Flusskontrolle	Anwendungen identifizieren (Ports) Segmentierung

**TCP:** HTTP (80)/HTTPS (443), SMTP (25), POP (110), IMAP (143), Telnet (23), SSH (22), FTP (20/21),...

**UDP:** DNS (53), DHCP (67/68), VoIP, Streaming,...

#### Ports

Der Port ist eine 16-Bit Zahl →  $2^{16} = 65.536$

Der Port identifiziert die Anwendung, sowohl beim Server als auch beim Client.

#### Gruppe von Ports

Well-Known-Ports	0 - 1.023
Registered-Ports	1.024 - 49.151
Private Ports	49.152 - 65.535

## L4-Adressierung

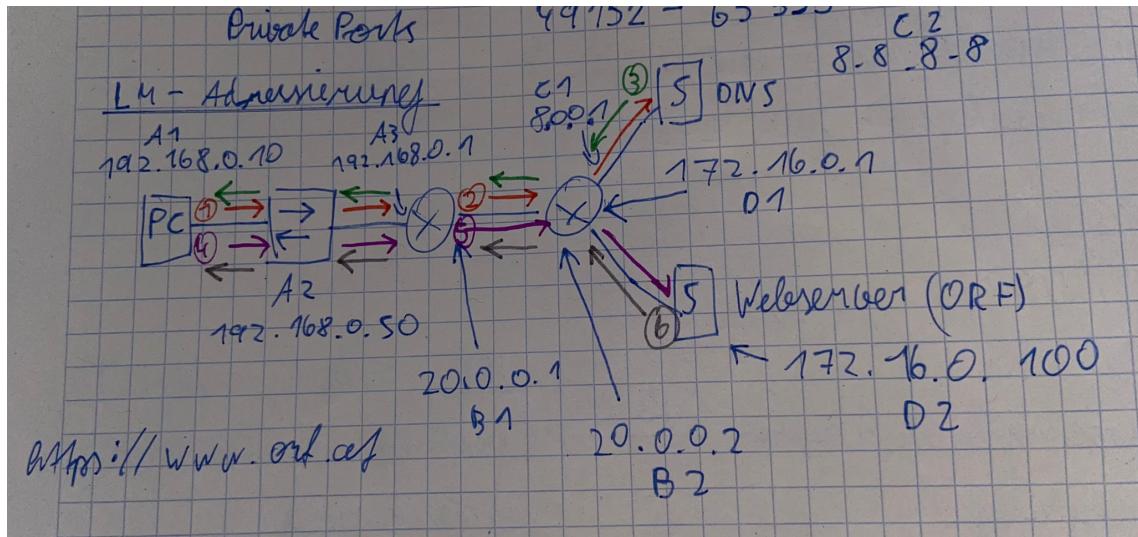


Abbildung 3.9: L4-Adressierung

	L2 (MAC)		L3 (IP)		L4 (Ports)	
	Source	Destination	Source	Destination	Source	Destination
1	A1	A3	192.168.0.10	8.8.8.8	53.722	53
2	B1	B2	192.168.0.10	8.8.8.8	53.722	53
3	C2	C1	8.8.8.8	192.168.0.10	53	53.722
4	A1	A3	192.168.0.10	172.16.0.100	60.112	443
5	B1	B2	192.168.0.10	172.16.0.100	60.112	443
6	D2	D1	172.16.0.100	192.168.0.10	443	60.112

## TCP

**Verbindungsauftbau: Drei-Wege-Handshake**

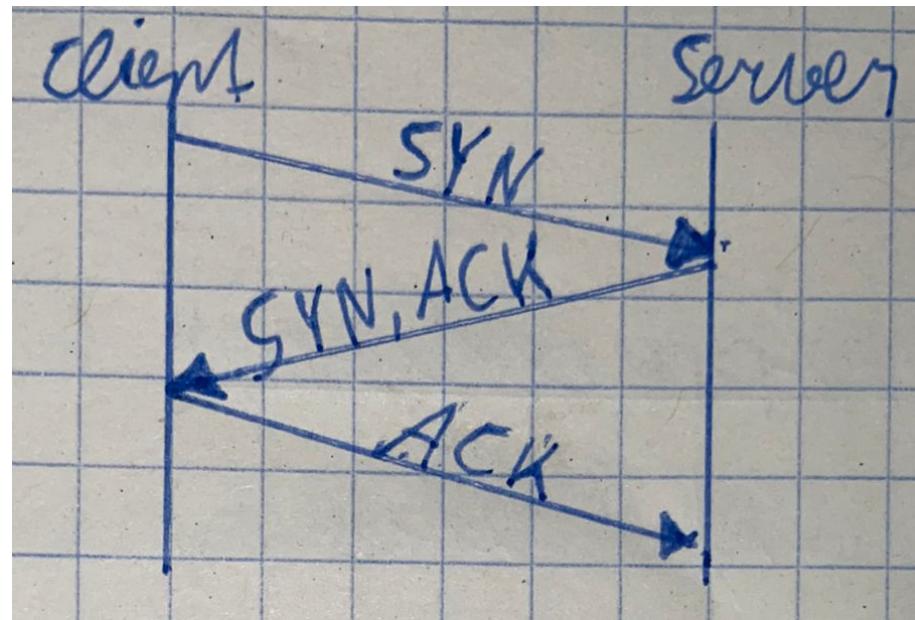


Abbildung 3.10: TCP 3-Way-Handshake

#### Verbindungsabbau: Zwei-Wege-Handshake

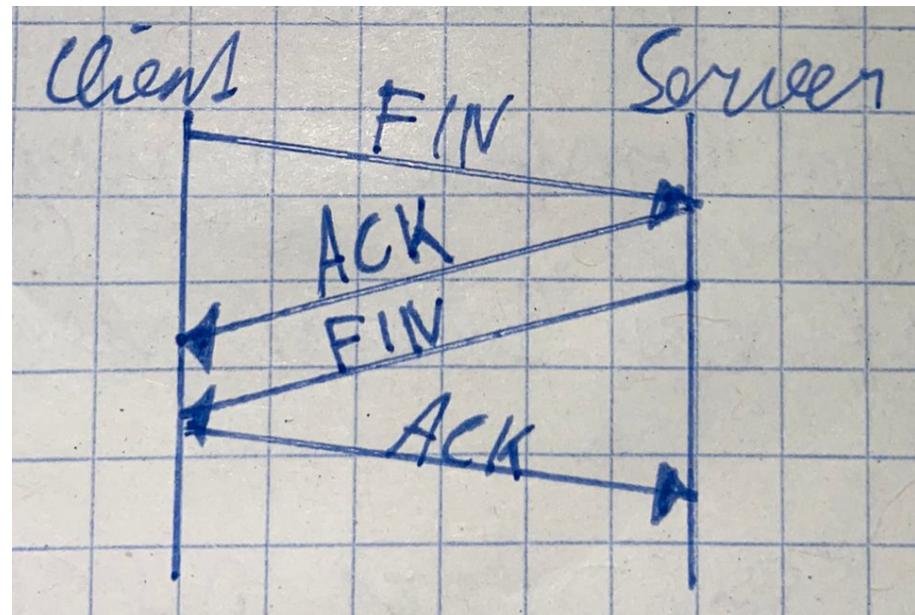


Abbildung 3.11: TCP 2-Way-Handshake

## Segmentierung

Es wird eine SEQUENCENUMBER mitgeschickt. Diese gibt die Reihenfolge an. Der Client bestätigt die Segmente mit ACK-Segmenten. Die ACK-NUMBER gibt an, welches Segment als nächstes kommen soll.

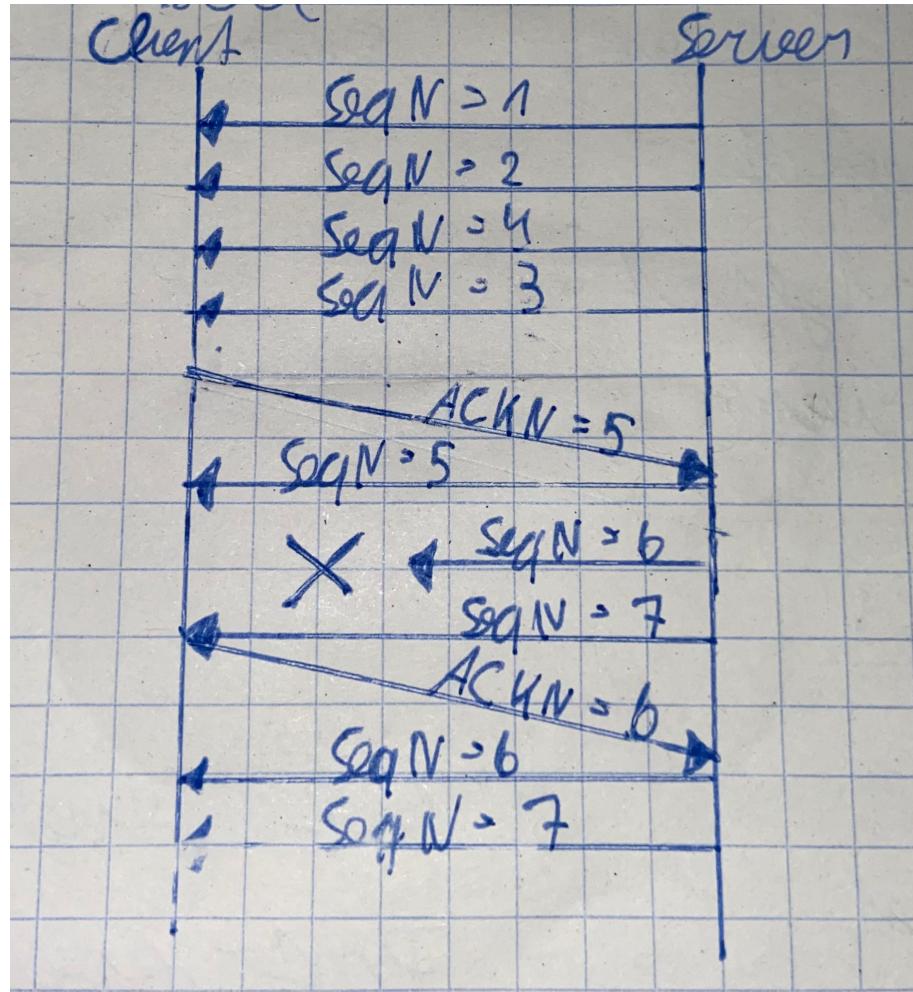


Abbildung 3.12: Layer 4 Segmentierung

## Flow-Control

Die Window Size gibt an wann das nächste ACK-Segment erwartet wird.

### 3.1.5 Layer 5, 6, 7 (Session, Presentation, Application)

#### Aufgaben

- Session erstellen und halten
- Regelung der Session, Restart, Exchange, Idle
- Format und Präsentation der Daten
- Verschlüsselung und Komprimierung der Daten
- Anwendungsspezifische Informationen

**Protokolle:** HTTP/HTTPS, FTP, Telnet/SSH, DHCP, DNS, SMTP, POP, IMAP

#### DNS (Port 53, UDP)

Um zu einer Domain die passende IP-Adresse zu finden. Typische DNS-Server: 8.8.8.8

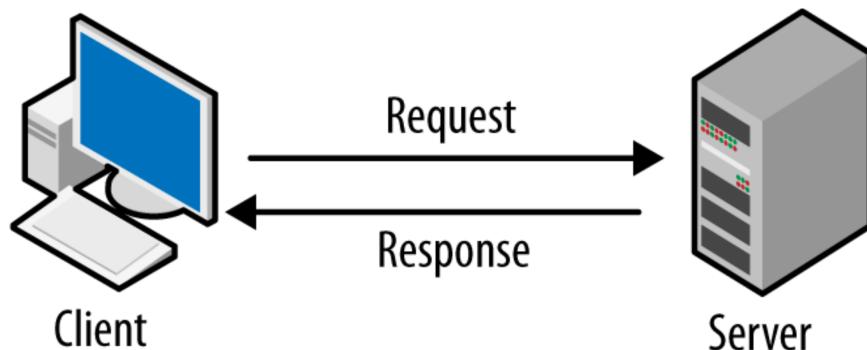


Abbildung 3.13: Request/Response Modell

#### Einträge

A ... IPv4-Endgerät  
AAAA ... IPv6-Endgerät  
MX ... Mail-Server

## Hierarchisches DNS-Modell

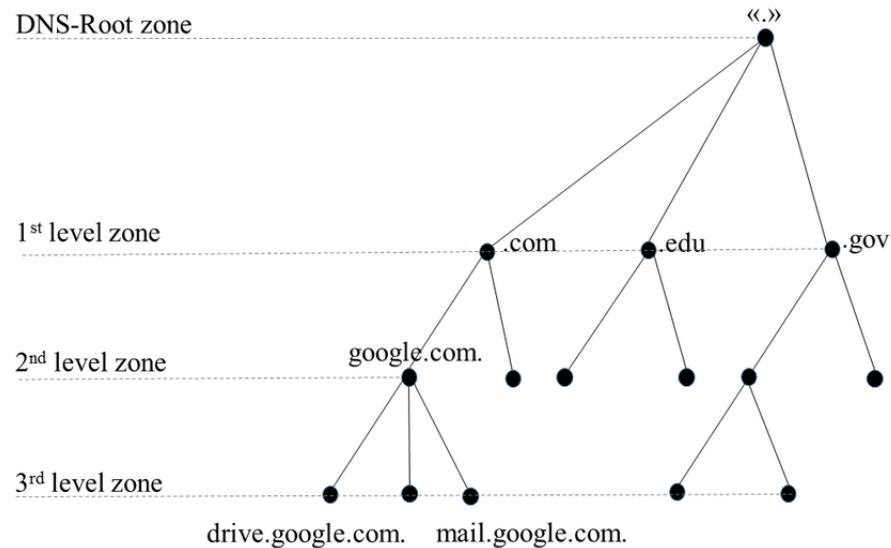


Abbildung 3.14: DNS-Hierarchie

Falls der DNS-Server keinen Eintrag findet, wird das Paket weitergeleitet. Der Client speichert die erhaltenen DNS-Einträge.

## DHCP (Port 67/68, UDP)

Die Hosts erhalten dynamisch eine IP-Konfiguration (IP-Adresse, Subnetzmaske, Default Gateway, DNS-Server, Lease Time,...).

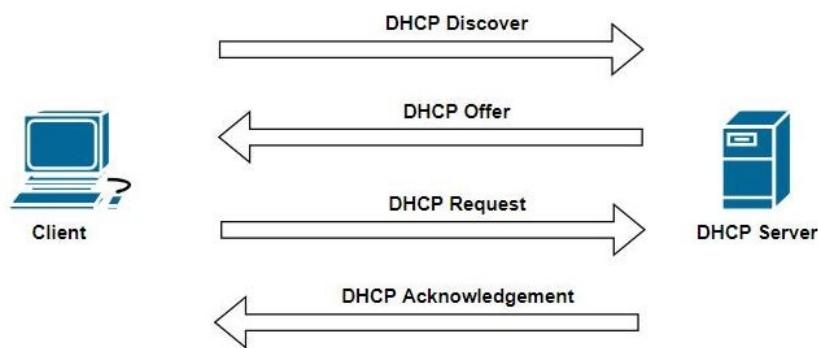


Abbildung 3.15: DHCP-Handshake

DHCP-Discover ... Broadcast

DHCP-Offer ... Unicast

DHCP-Request ... Broadcast

DHCP-ACK ... Unicast

**Achtung:** DHCP-Spoofing

## HTTP/HTTPS (Port 80/443, TCP)

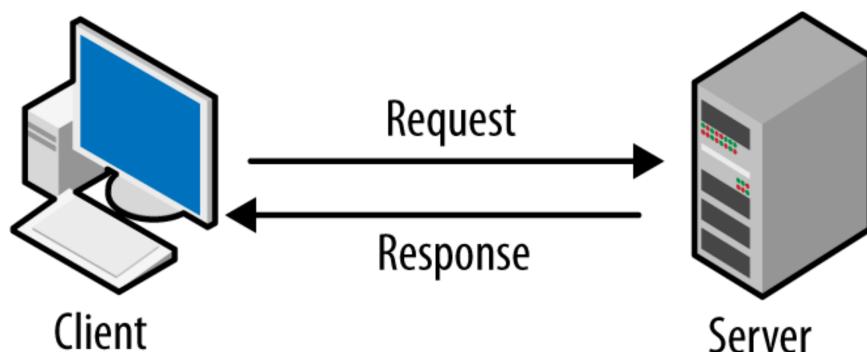


Abbildung 3.16: Request/Response Modell

URL:	https://	www.google.com/	index.html
	Protokoll	Domain IP-Adresse (DNS)	Ordnerstruktur, Datei

## Befehle

Get, Post, Put, Delete,...

Bei HTTP ist alles im Klartext.

Bei HTTPS wird zusätzlich mit SSL/TLS verschlüsselt.

## E-Mail

E-Mail-Adresse:	name	@	gmail.com
	Benutzername		Domain

## SMTP (Port 25, TCP)

Senden von Emails. Wird zum Senden von Mails und dem Weiterleiten zum Zielserver benutzt. SMTP kann zusätzlich Feedback geben (z.B. Ziel nicht erreichbar,...).

## **POP (Port 110, TCP)**

Empfangen von E-Mails. Man erhält vom Server das Original. Die Mail wird am Server gelöscht (Vorteil: Speicherplatz, Security).

## **IMAP (Port 143, TCP)**

Empfangen von E-Mails. Man erhält vom Server eine Kopie. Das Original bleibt am Server gespeichert (Vorteil: Verbindung mit mehreren Geräten ist praktisch, Backup).

## 3.2 VLANs

Ein physisches Netz wird in mehrere logische Teilnetze (Layer 2) unterteilt.

Vorteile	Nachteile
Kosten	(Konfiguration)
Security	
Flusskontrolle	
Übersicht	
kleinere Broadcast-Domain	
Effizienz & Performance	

### Arten von VLANs

- Daten VLANs
- Default VLAN (bei cisco 1)
- Voice VLAN
- Management VLAN
- Native VLAN (Frames ohne VLAN-Tag kommen in das Native VLAN, kann nur am Trunk passieren)

**Access Ports** transportieren nur ein VLAN.

**Trunk Ports** können viele VLANs transportieren.

Die VLAN-Namen werden zusätzlich im Header eingetragen (802.1q → Ethernet)

### ACL

Je nach IP-Adresse (Standard) bzw. Port (Extended) wird ein Packet blockiert oder zugelassen.

### Wildcardmask

Subnetzmaske: Teilt IP-Adresse in Netz- und Hostteil

1 ... Netzteil (relevant für das Netz)

0 ... Hostteil (irrelevant für das Netz)

→ unflexibel



Wildcardmaske '1' & '0' können beliebig gezählt werden

0 ... relevantes Bit der IP-Adresse

1 ... nicht relevantes Bit der IP-Adresse

255.255.255.255      any  
0.0.0.0                  host

## Position der Wildcardmask

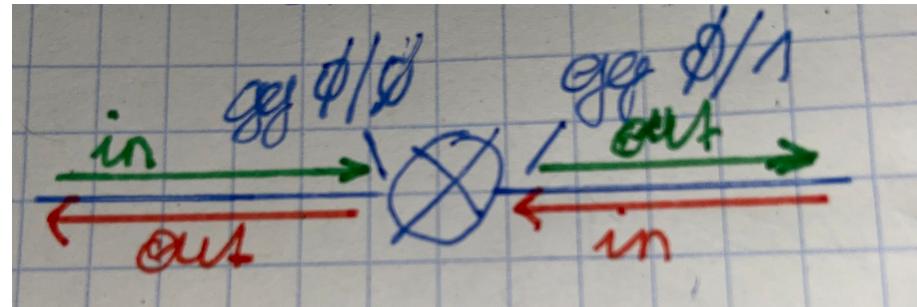


Abbildung 3.17: Position der Wildcardmask

Regeln bei Interfaces: eingehend & ausgehend

**Achtung** Die letzte Zeile in jeder ACL ist 'deny any'.

## Static NAT (1:1 Mapping)

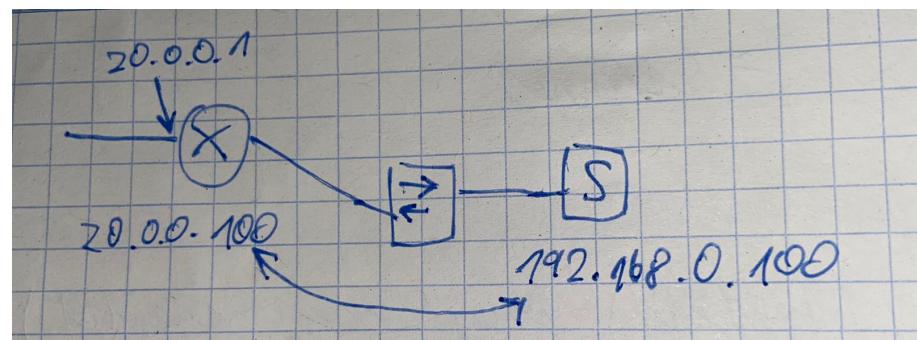


Abbildung 3.18: Static NAT, 1:1 Mapping



## NAT mit PAT (n:1 Mapping)

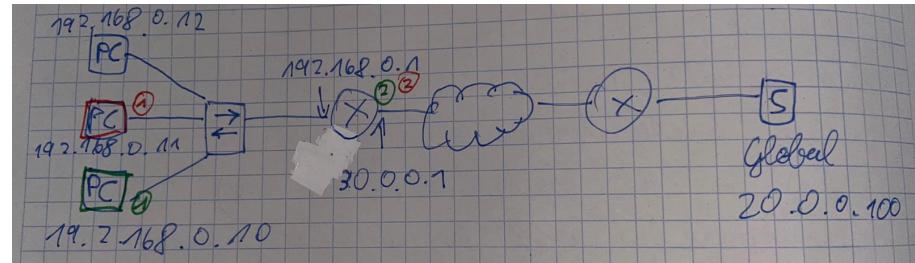


Abbildung 3.19: NAT mit PAT, n:1 Mapping

	Source IP	Destination IP	Source Port	Destination Port
1	192.168.0.10	20.0.0.100	51000	443
1	192.168.0.11	20.0.0.100	51000	443
2	30.0.0.1	20.0.0.100	51000	443
2	30.0.0.1	20.0.0.100	51001	443

Vorteile	Nachteile
<ul style="list-style-type: none"><li>+ IP-Adressen sparen</li><li>+ Security</li><li>+ IP-Adressen Schema kann frei gewählt werden</li></ul>	<ul style="list-style-type: none"><li>- Ende zw. Ende Verbindung geht verloren</li><li>- Paketverfolgung und Troubleshooting</li><li>- Performance</li></ul>

## 4 Routing

Router muss entscheiden welcher Weg der 'beste' Weg ist.

→ bei welchen Interface (Netz) rauschicken = Routing

Routing Tabelle wird durch ...

- dynamisch (Routingprotokolle)
- statische Einträge

... aufgebaut (in der Praxis meist aus Mischung).

Router wählt Route mit am meisten Bits bei Ziel übereinstimmung (Vergleich von Route & Destination IP).

### 1) Einträge in Routing Tabelle

- Direkt verbundene Netze: Aktive & angeschlossene Netze am Router mit IP-Konfiguration → automatisch (Status Code: C, L)
- Remote Netze: statisch oder dynamisch (vom Routingprotokoll abhängig) eingetragen (Status Code: S, R, O, E,...)
- Default Route (gateway of last resort): Next Hop falls der Router keine passende Route findet, statisch oder dynamisch.  
Route: 0.0.0.0 / 0 ... 0 Bits müssen übereinstimmen

### 2) Eintrag in Cisco CLI

R	30.0.4.0/24	[120/7]	via 10.0.3.2	00:13:29	Serial 10/1/1
Status	Ziel	AD/Metrik	IP (ausgehendes Interface)	Zeitstempel	Interface

#### Status Code

C ... connected      Direkt verbundene Netzte

- L ... local IP vom Interface, lokale Route
- S ... static statisch eingegebene Route
- R ... RIP entsprechendes Routingprotokoll
- O ... OSPF entsprechendes Routingprotokoll
- E ... EIGRP entsprechendes Routingprotokoll

## Ziel

IP-Adresse des Zielnetzes mit Präfix (nicht unbedingt Subnetzmaske). Es müssen die angegebene Anzahl von Bits (Präfix) mit Destination IP-Adresse übereinstimmen (damit Route in Frage kommt). Route mit am meisten übereinstimmenden Bits (von links). Problem: es wird keine Subnetzmaske der Destination IP mitgeschickt → normalerweise auch nicht bekannt.

Dest IP: 172.16.0.10 (letztes Oktett: 00001010)

- 
- 1) 172.16.0.0 /16 Bis Bit 16 übereinstimmend
  - 2) 172.16.0.0 /24 Bis Bit 24 übereinstimmend
  - 3) 172.16.0.0 /26 Bis Bit 26 übereinstimmend
  - 4) 172.16.0.0 /30 Bis Bit 29 nicht übereinstimmend
  - 5) 172.17.0.0 /24 am 2. Oktett stimmt es nicht überein

Router wählt 3. Variante (???). Dort stimmt die angegebene Anzahl an Bits (Präfix) überein

## AD: Administrative Distanz

Router kann Route über mehrere Arten lernen (z.B. statisch, RIP, OSPF,...). AD gibt an wie 'vertrauenswürdig' eine Route ist. Router verwendet Route mit niedrigster AD, andere Routen sind Backups und werden vorerst nicht im Routing Table angezeigt.  
→ wenn 'beste' Route ausfällt wird nächst beste verwendet

Standard Werte bei Cisco Routern

AD	
Direkte Routen	0
Statische Routen	1
EIGRP	90
OSPF	110
RIP	120

## Metrik

Von einem Routingprotokoll kann der Router mehrere Routen zum gleichen Ziel lernen. Die Metrik gibt an, 'wie weit' das Ziel entfernt ist. Der Router verwendet die Route mit der geringsten Metrik. Falls eine Route ausfällt, wird auf Backup-Routen zurückgegriffen.

## 3) Statische Routen

Werden in kleineren Netzen mit geringen Veränderungen, bei speziellen Zielnetzen oder Router mit nur einen Nachbar (Stub-Network) verwendet.

Problem: Statische Routen werden nicht automatisch aktualisiert und müssen händisch aktualisiert werden.

## 4) Dynamische Routingprotokolle

Je nach Ablauf des Routingprotokolls unterscheidet man unterschiedliche Kategorien.

- **Pfadvektorprotokolle**

Diese Protokolle speichern den Pfad/Weg zum Ziel. Sie sind besonders effizient gegen Routing-Schleifen und eignen sich dadurch zum Routen von autonomen Systemen.

Beispiel: BGP (Border Gateway Protocol), Metrik: Anzahl der autonomen Systemen bis zum Ziel (Zusatzinformation IGP-Metrik: wie lange dauert es durch ein autonomes System)

- **Distanzvektorprotokolle**

Diese Protokolle speichern nur die Distanz zum Ziel

Beispiel: RIP (Routing Information Protocol), Metrik: Anzahl der Hops  
EIGRP (Enhanced Interior Gateway Routing Protocol), Metrik: Bandbreite, Auslastung, Delay, Zuverlässigkeit

EIGRP kennt die ganze Topologie im System, speichert diese aber nicht direkt ab.

- **Link-State-Protokolle**

Diese Protokolle kennen die ganze Topologie im System. Daraus berechnet sich jeder Router die besten Routen zu allen Zielen.

Beispiel: OSPF (Open Shortest Path First), Metrik: Bandbreite

## 5) Algorithmen zur Bestimmung des kürzesten Weges

- Bellman Ford (RIP)

- Dijkstra (OSPF)
- DUAL (EIGRP)

## Dijkstra Algorithmus

Ablauf:

1. Startknoten mit 0 markieren, alle anderen mi  $\infty$ : 'Distanz' und 'besucht' merken
2. Solange es unbesuchte Knoten gibt:

- Jenen Knoten mit der kürzesten Distanz wählen
- Als besucht markieren
- Für alle unbesuchten Knoten die Distanz berechnen
- Falls der Wert kleiner ist, als der aktuelle, diese speichern

## 5 IPv6

Eine IPv6-Adresse ist eine 128 Bit Zahl. Es gibt  $2^{128}$  IPv6-Adressen ( $340 \cdot 10^{36}$ ).

### Schreibweise einer IPv6-Adresse

- Hexadezimale Schreibweise (32 Zeichen)
- Gruppen von 16 Bit mit : getrennt
- Führende Nullen werden in jeder Gruppe weggelassen
- Einmalig kann der längste Block an Nullen mit :: ersetzt werden

Bsp:

2001:ABAD:0000:0430:0000:0000:00C9:0001

2001:ABAD:0:430:0:0:C9:1

2001:ABAD:0:430::C9:1

### Subnetzmaske

- trennt in Netz- und Hostteil
- nur noch Präfix-Notation
- Es wird fast nur /64 verwendet

### Idee von IPv6

- mehr IP-Adressen
- Problem: alle Protokolle die IPv4 verwenden müssen erneuert werden
- Alte Fehler/Security-Probleme beheben
- leichterer Header

## Übergang von IPv4 zu IPv6

- Dualer Stack
- Translation

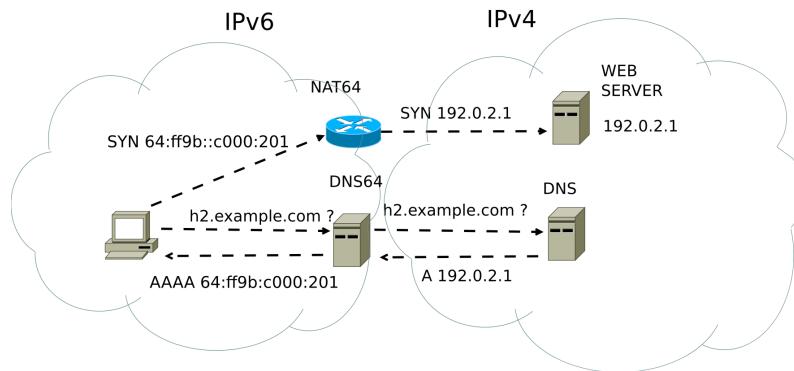


Abbildung 5.1: IPv4-IPv6 Translation mit NAT64

- Tunneling

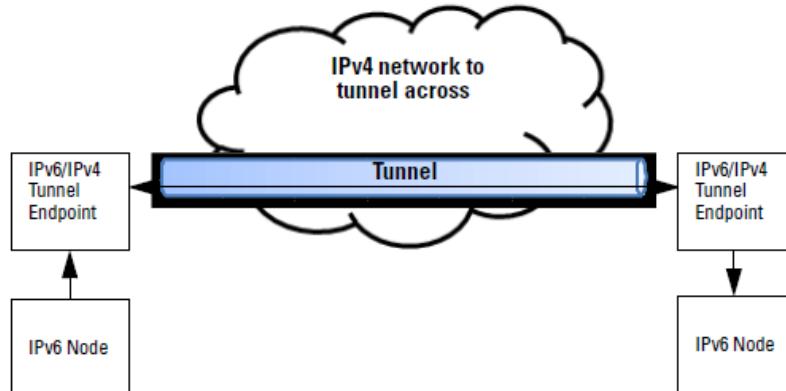


Abbildung 5.2: IPv4-IPv6 Tunneling

## Kommunikationsarten

- Unicast
- Multicast
  - `ff02::1` ... all-nodes-multicast (Broadcast)
  - `ff02::2` ... all-router-multicast
- Anycast (der 'nächste' einer Gruppe bekommt den Anycast)

## IPv6-Unicast Adressen

- Global Unicast Adressen 2000-3fff (vgl. öffentliche IP)
- Link Local Adressen fe80-febf (für das lokale Netz, nicht routbar)
- loopback ::1 (vgl. IPv4 127.0.0.1)
- Unspecified Adress :: (vgl. IPv4 0.0.0.0)
- Unique Local fc00-fdff (vgl. IPv4 NAT)
- Embedded IPv4

## IP-Konfiguration

- statisch (GUA, LLA)
- dynamisch
  - SLAAC
  - SLAAC mit stateless DHCPv6 Server
  - DHCPv6

## SLAAC

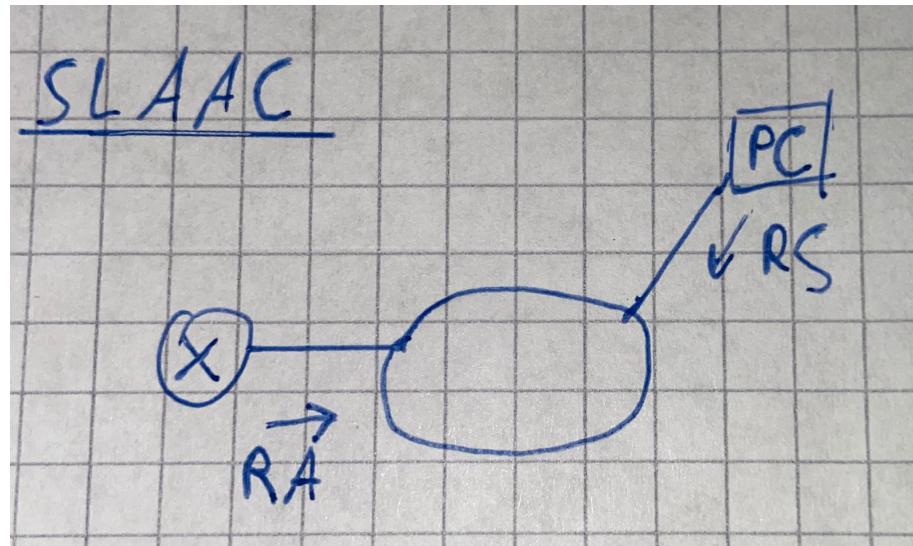


Abbildung 5.3: SLAAC

Router senden (ca. alle 200s) ein RA-Paket (Router Advertisement) aus. Dies enthält die wichtigsten Informationen für die Hosts (Präfix, Präfix-Länge, Default-Gateway). Die Hosts geben sich dann selbst die IPv6-Adresse.

Hostteil

- Zufallszahl (ND-Protokoll)
- EUI-64 (MAC-Adresse)

Die Hosts können RS (Router Solicitation) Pakete aussenden um das RA-Paket anzufordern.

## 6 WLAN (Wireless Local Area Network)

Bei einem drahtlosen Netzwerk findet die Übertragung ohne Kabel statt. Es werden elektromagnetische Wellen über die Luft übertragen. Für die drahtlose Übertragung im Netzwerk bedeutet dies, dass sich Layer 1 und Layer 2 ändern. Die darüber liegenden Layer bleiben unverändert. Durch diese Änderung der Übertragungsart ergeben sich einige Vorteile aber auch Nachteile.

Vorteile	Nachteile
<ul style="list-style-type: none"> <li>+ BYOD: bring your own device</li> <li>+ Kosten: Besonders in bestehenden Gebäuden</li> <li>+ Anpassungsfähigkeit</li> </ul>	<ul style="list-style-type: none"> <li>- Geteiltes Medium für viele Teilnehmen</li> <li>- Störungen</li> <li>- Geschwindigkeit und Reichweite</li> <li>- Security</li> </ul>

### Antennen

Antennen sind die Grundlage für eine Übertragung über die Luft. Sie geben ein Signal in die Luft ab (Senderantenne) und können es auch wieder aus der Luft aufgreifen (Empfängerantenne). Je nach Anwendung eignen sich verschiedene Arten von Antennen.

- Omnidirektionale Antennen: senden in alle Richtungen (Kugel)
- Direkte Antennen: Können gezielt senden
- MIMO (multiple input multiple output) Antennen: aktuell meist 8 Antennen

### Arten von Wireless Netzwerken

Wie auch schon bei den verkabelten Netzen unterscheidet man Netzte nach ihrer Größe. Je nach Größe ergeben sich unterschiedliche Anforderungen und Schwierigkeiten.

- **WPAN:** kurze Distanz (ca 10m), Frequenz meist 2.4 GHz z.B. Bluetooth, Zigbee
- **WLAN:** mittlere Distanz (ca 100m), Frequenz ist 2.4 GHz oder 5 GHz z.B. Wifi

- **WMAN:** große Distanz (kann sehr unterschiedlich sein), Frequenz zwischen 2 und 66 GHz z.B. Wifi, WiMax
- **WWAN:** riesige Distanzen (bis zu 50km), Frequenz zwischen 2 und 66 GHz z.B. WiMax

## Technologien von Wireless Netzwerken

Es gibt unterschiedliche Technologien dir drahtlos übertragen. Je nach Reichweite und Anwendungsgebiete sind unterschiedliche Technologien sinnvoll. Nicht alle Technologien sind dazu geeignet oder dafür entworfen um Netzwerksdaten zu übertragen. Manche Technologien können dies trotzdem umsetzen.

- **Wifi** (IEEE 802.11)
- **Bluetooth** (IEEE 802.15)
- **WiMax** (IEEE 802.16)
- **Satelliten Breitband:** kann als Internetzugang genutzt werden, z.B. Starlink (Oktober 2023 ca. 5000 Geräte, in einer Entfernung von 500 bis 600km, beantragt sind 22.000 Satelliten)
- **Mobilfunk Breitband:** viele verschiedene Standards die meist nach gravierenden Änderungen (Generationen) unterteilt werden. Bei der Änderung in eine neue Generation ist die Geschwindigkeit immer ein entscheidender Faktor (3G → 10x → 4G → 100x → 5G).

## 6.1 Wifi (802.11)

### Elektromagnetische Welle

Gesendet wird mit elektromagnetischen Wellen. Diese kennt man vom sichtbaren Licht. Dort nimmt der Mensch unterschiedliche Wellenlängen als verschiedene Farben wahr. Jene Wellenlänge die zum übertragen von Wifi genutzt werden liegen außerhalb des sichtbaren Lichts. Desto länger die Welle ist, desto kürzer ist seine Frequenz (indirekt Proportional). **Kurze Wellen** besitzen **mehr Energie** als lange Wellen.

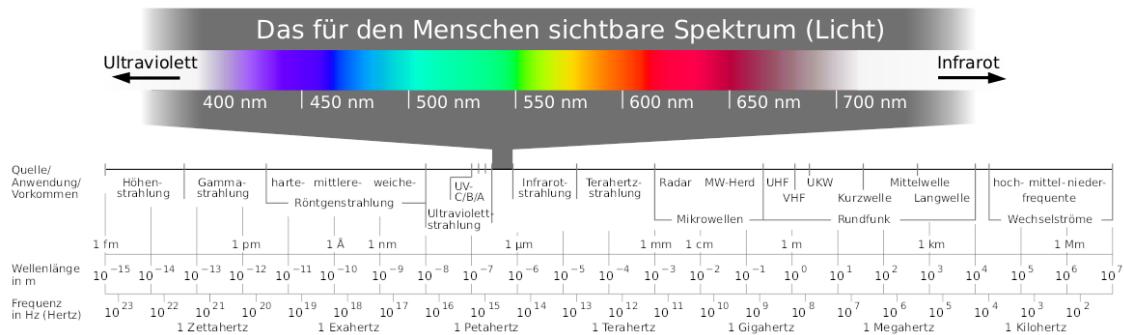


Abbildung 6.1: Elektromagnetisches Spektrum

- **2.4 GHz** (1-10dm) UHF: Ultra High Frequency
- **5 GHz** (1-10cm) SHF: Super High Frequency

Für das 5G Netz wurden neue Frequenzbereiche festgelegt und versteigert.

- Mobilfunk: 600 MHz bis 6 GHz
- WLAN: 24 GHz bis 40 GHz

## Komponenten im WLAN

- Endgeräte: mit Netzwerkkarten und Antennen
- Wireless Router: Multifunktionsgeräte mit eingebautem Switch, Router, Modem Access Point,...
- Access Points: Schnittstelle zwischen dem drahtlosen Netz und dem verkabelten Netz. Man unterscheidet zwischen Autonomen-Access-Points (schwer erweiterbar) und Controller-Based-Access-Points.

## Wifi Frame

Frame Control	Metainformationen z.B. Protokoll, Art des Frames,...	2 Bytes
Duration	Übertragungsdauer, aufgrund unterschiedlicher Framelänge	2 Bytes
Address 1	Empfänger MAC-Adresse	6 Bytes
Address 2	Sender MAC-Adresse	6 Bytes
Address 3	MAC BSSID (WLAN Segment)	6 Bytes
Sequence Control	Hängt vom AP ab	
Address 4	MAC-Adresse vom Access Point	6 Bytes
Frame Body:	Header der restlichen Layer und Daten	
FC	Fehlerüberprüfung mit CRC	4 Bytes

## Operations-Modi

- Ad Hoc: Peer-to-Peer Netzwerk ohne Router
- Infrastruktur: Dahinter ein verkabeltes Netz
- Tethering: Hotspot zur Weiterleitung zwischen zwei Netzen

## Kollisionen (CSMA/CA)

Wireless Netzwerke nutzen eine Half Duplex Medium zum Senden. Man kann zeitgleich senden und empfangen. Zusätzlich ist es ein Shared Medium, das heißt viele Teilnehmer sind mit dem gleichen Medium verbunden. Somit kann es zu Kollisionen kommen (CSMA - Carrier Sense Multiple Access). Wifi löst das Problem mit Collision Avoidance (CA), es versucht also Kollisionen zu vermeiden. Falls gerade keiner sendet wird um Zeit beim Access Point angefragt. Dann erhält man einen Zeitslot indem man seine Daten senden und empfangen kann.

## Verbinden mit einem Accesspoint

- AP finden (aktiv, passiv)
- Authentifizieren: SSID, Passwort, Network Mode (a, b, g,...), Security (WPA, WPA2,...), Channel
- Verbindung herstellen

## Channels

Die Frequenzen werden in kleinere Bereiche aufgesplittet. Gleiche Channels können sich gegenseitig stören. Überlappende Access Points sollten verschiedene Channels nutzen.

- 2.4 GHz: Europa 13 Channels (1, 6 & 11 nicht überlappend)
- 5 GHz: 24 Channels (alle ohne Überlappung)

## WLAN-Angriffe

- Datendiebstahl: Shared medium → Verschlüsselung
- DoS: falsch konfiguriert, Störsender,...
- Rogue Access Point: zusätzlichen falschen AP ins Netz eingefügt
- Evil Twin: einen AP einfügen, der gleich aussieht aber in ein anderes Netz führt

## Sicherheit und Verschlüsselung

- SSID Beacon verbergen (passic)
- MAC-Adressen filtern (L2 Security)
- Authentifizierung
  - **Open:** ohne Password (nicht empfohlen)
  - **Shared Key:** WEP, WPA (TKIP+AES), WPA2, WPA3

Bei WPA2 unterscheidet zwei Varianten zum Authentifizieren:

- **Personal:** ein Passwort für alles (PSK, Pre Shared Key), eher im privaten Bereich
- **Enterprise:** Anmeldung mit Username und Passwort, man meldet sich bei einem Server (z.B. RADIUS), eher im Firmenbereich

## WPA2-Personal Handshake: Pre-Shared-Key (4-Way)

Zum Austausch der Schlüssel zwischen dem Access Point und dem Client findet ein 4-Way-Handshake statt. Dabei werden die benötigten Schlüssel generiert. Zum Generieren der Schlüssel muss das Passwort nie übertragen werden, deshalb nennt man die Variante auch PSK (Pre-Shared-Key). Der Schlüssel wurde also davor schon ausgemacht.

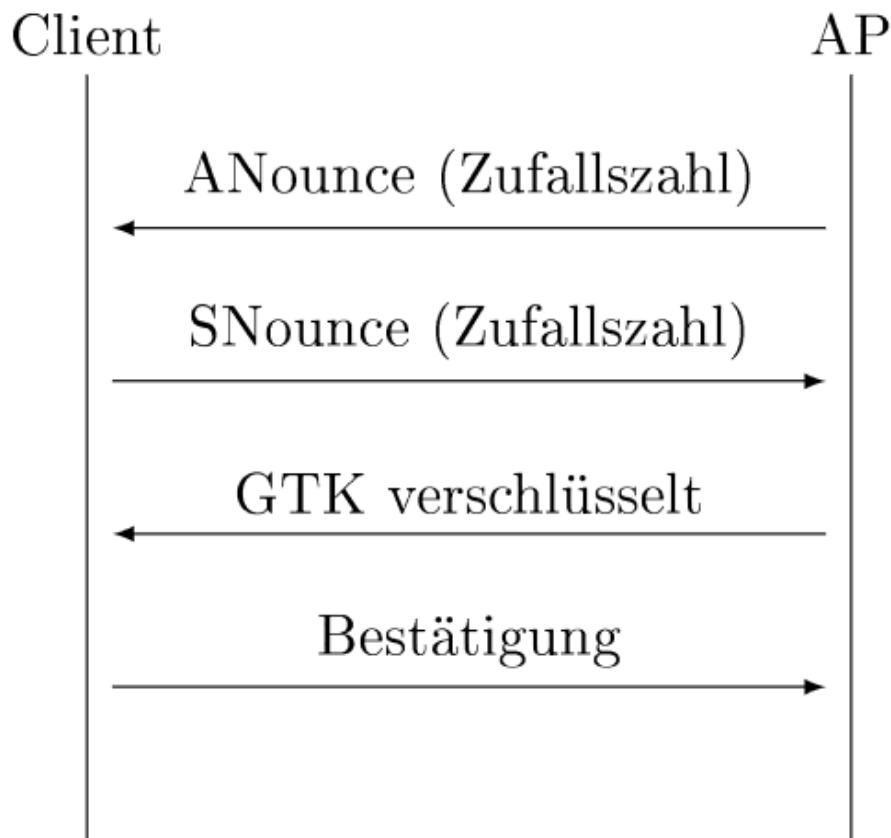


Abbildung 6.2: WPA2 Personal Handshake

**PTK (Pairwise Transient Key):** für Unicasts, jeder hat seinen eigenen Schlüssel mit dem AP

**PTK = PRF(Pwd+ANounce+SNounce+APMAC+ClientMAC)**

**PRF (Pseudo Random Function):** ist eine Pseudo-Zufallsfunktion die dann den Schlüssel erzeugt und den Geräten bekannt ist.

**GTK (Group Temporal Key):** für Broadcast und Multicasts im Netz, für alle Teilnehmer gleich.

Das Passwort wird nie über das Medium ausgetauscht, deshalb nennt man das Verfahren Pre Shared Key. Die Nachricht wird nach Layer 2 verschlüsselt. Dieser kann nicht verschlüsselt werden, da der Access Point die Frames identifizieren muss. Danach im verkabelten Netz, wie sonst auch immer, wird wieder nach Layer 4 verschlüsselt (z.B. mit TLS).

## 7 Network Security

Netzwerkangriffe können auf unterschiedliche Arten, unterschiedlichen Ebenen und verschiedene Protokolle stattfinden. Deshalb muss bei einem Security-Konzept möglichst alles berücksichtigt werden.

Angriffe	OSI-Modell	Abwehr
Social Engineering Passwörter Pretexting	L8-Mensch	Schulungen Vernünftig und vorsichtiges handeln
SQL-Injection Wurm, Virus, Trojaner Ransomware, Spyware Protokolle (HTTP, FTP, Telnet, DHCP, DNS,...)	L7-Application L6-Presentation L5-Session	Firewall, IPS, IDS, ESA, WSA Eingabeüberprüfung gute Software & Protokolle End-Point-Detection Anti-Virus, Updates
DDOS → TCP (SYN-Flood, → UDP	L4-Transport	IPS, IDS Firewall, ACL
Routing, DDoS MITM, IP-Spoofing Protokolle (ICMP,	L3-Network	Firewall, ACL, IPS, IDS sichere Protokolle (IPsec)
MITM: ARP-Spoofing, MAC-Spoofing DDOS: MAC-Flooding Protokolle (STP, CDP)	L2-Data Link	MAC-Filter, AAA sichere Protokolle Verschlüsselung
DoS (Störsender, Zerstörung der Infrastruktur) Physischer Zugang MITM, Hardware	L1-Physical	Zutrittskontrolle Backups

Dem Angreifer reicht eventuell ein einziger Angriffspunkt im Netz. Meist sind die User (Personen) das größte Problem. → **"Der Angreifer muss nur einmal gewinnen"**

## Sicherheitsrichtlinien

User	Unternehmen
<ul style="list-style-type: none"> <li>• Passwörter (Mindestlänge, eins pro Account, keine persönlichen Daten) → Passwortmanager, 2FA</li> <li>• Datenverwaltung (Wann?, Wo?, Welche?, Wann?,...)</li> <li>• Firewall</li> <li>• Updates (OS, Software)</li> <li>• Antivirus/Antispysoftware</li> <li>• Vernünftig handeln</li> </ul>	<ul style="list-style-type: none"> <li>• Passwortrichtlinien, User Verwaltung, Recht vergeben</li> <li>• Firmengeräte, spezielle Rechte, wie beim User</li> <li>• DMZ, VPN, Firewall, IPS, IDS, WSA, ESA</li> <li>• Zugangskontrollen</li> <li>• Schulung der Mitarbeiter</li> <li>• Backups</li> <li>• Pen-Testing</li> <li>• Risikoanalyse → Schwachstellen kennen</li> <li>• Verhaltensanalyse</li> <li>• Datatransfer sichtbar machen</li> </ul>

## 7.1 Firewall

Mit einer Firewall kann der eingehende/ausgehende Datenverkehr kontrolliert, protokolliert und gefiltert werden ( sperren, freigeben).

### Unterscheidung nach Position

- **Personal Firewall** (am eigenen Gerät) z.B. Windows Defender, UFW,...
- **External Firewall** (zwischen lokalen & globalen Netz) z.B. ASA, Fortinet, Barracuda, PFSense,...

### Unterscheidung nach Funktion

- (L4) **Paketfilter**: IP-Adressen, Ports z.B. ACL
- (L4) **Stateful Inspection**: Untersucht die ganze Sitzung (mehrere Aufrufe zu z.B. gleiche IPs)
- (L7) **Application Firewall**: Proxy Server
- (Daten) **Deep Paket Inspection Firewall**

Eine falsch konfigurierte Firewall bietet keinen Schutz. Eine Firewall muss ständig gewartet und aktualisiert werden.

## 7.2 IDS & IPS

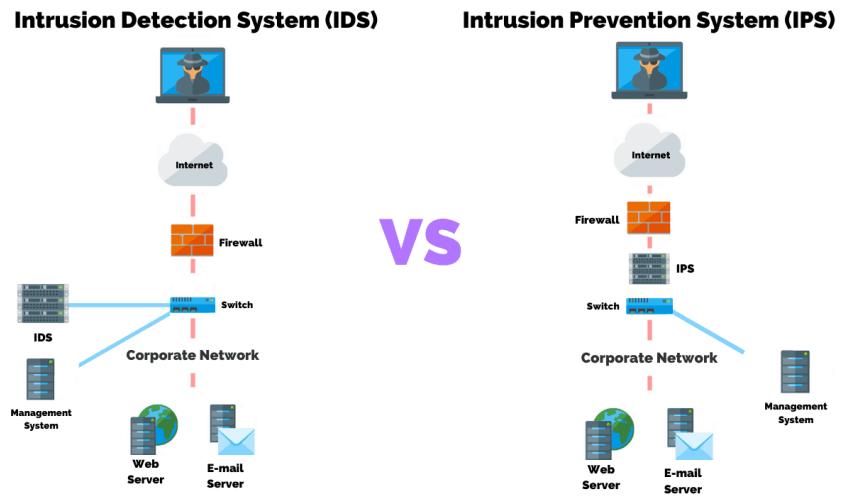


Abbildung 7.1: IPS & IDS

IDS (Intrusion Detection System)	IPS (Intrusion Prevention System)
Wird nur parallel informiert + schneller (da es parallel ist) - nur Warnungen	Alles muss über IPS - langsamer (da seriell) + sicherer

## 7.3 Honeypot

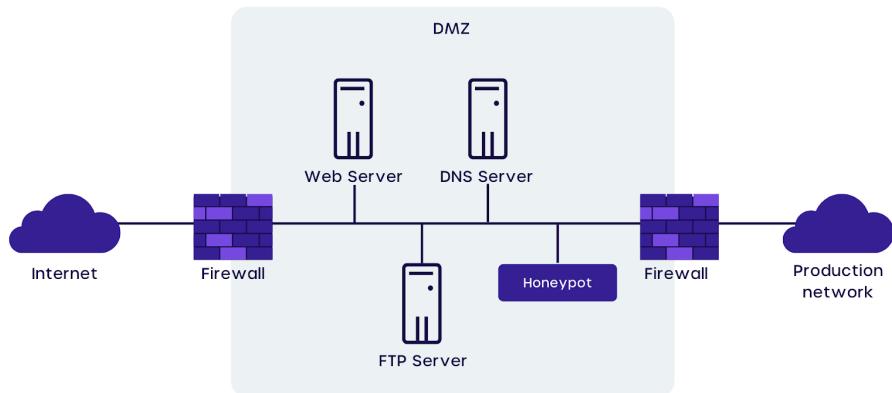


Abbildung 7.2: Honeypot

Bei einem Honeypot werden bewusst veraltete Software & Hardware für Angreifer als Köder aufgestellt.

## 7.4 VPN

Erstellt eine verschlüsselte Verbindung zu einem entfernten VPN-Server.

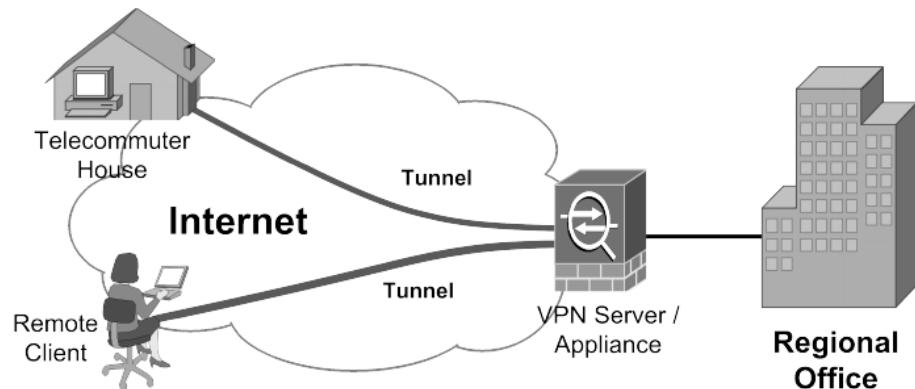


Abbildung 7.3: VPN

### Arten von VPN

- Ende-zu-Ende VPN
- Ende-zu-Netz VPN
- Netz-zu-Netz VPN

## 7.5 ESA (Email Security Appliance) & WSA (Web Security Appliance)

Funktioniert wie ein Proxy-Server, spezifisch für Email/Web.

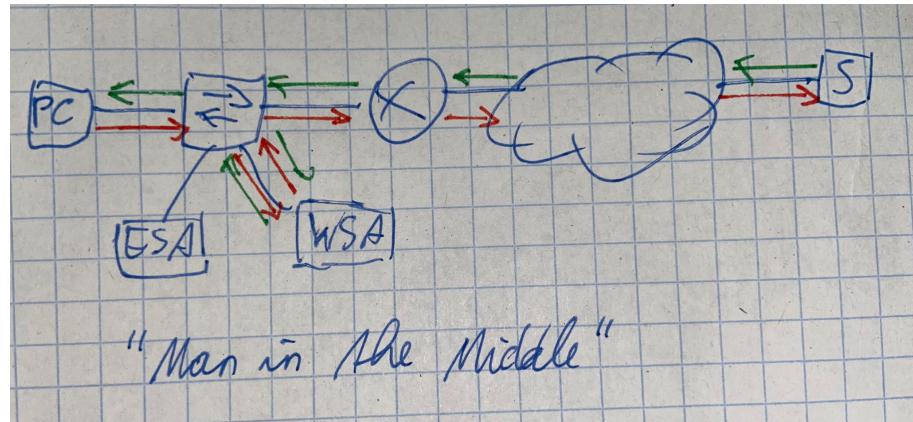


Abbildung 7.4: ESA & WSA

## 7.6 IPsec

Ziel: Sicheres Protokoll zur Datenübertragung

- **Transport-Modus:** Verschlüsselung ab L4 und fügt eine Authentifizierung in den Header ein.
- **Tunnel-Modus:** Alles wird verschlüsselt und es wird ein neuer verschlüsselter Header angehängt. Darin stehen die wichtigsten Felder (MAC, IP,...) + Authentifizierung

# 8 Hashfunktionen

Bei einer Hashfunktion ist die Wertemenge meist wesentlich kleiner als die Lösungsmenge.  
 Die Elemente der Wertmenge können normalerweise eine beliebige Länge haben.  
 Die Elemente der Lösungsmenge haben meist eine fixe Länge.

- Anfangsbuchstabe:     Hallo → H  
                              Tim → T
- Postleitzahl:        Grins → 6591  
                           Neustift → 6167
- CRC ...

## 8.1 Kryptographische Hashfunktionen

Kryptographische Hashfunktionen müssen spezielle Eigenschaften erfüllen. Die wichtigste Eigenschaft ist, dass es sich um eine Einwegfunktion handelt.

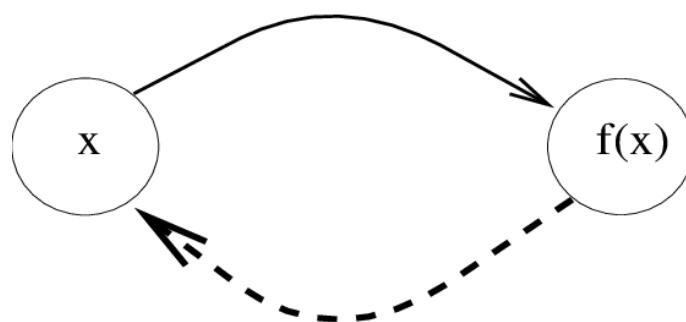


Abbildung 8.1: Einwegfunktion

### Eigenschaften

- **Einwegfunktion:** Unumkehrbarkeit muss so gut wie möglich gegeben sein
- Diffusion ('Lawineneffekt'): kleine Änderung in der Eingabe bewirkt die ganze Ausgaben.

- Konfusion: Von Hashwert kann man keine Rückschlüsse auf den Eingabewert machen
- Eindeutigkeit
- Kollisionsresistenz: Die Wahrscheinlichkeit, dass Kollisionen vorkommen soll so klein wie möglich sein (im besten Fall gleich Null sein)

## Hash-Algorithmen

- MD5: 128 Bit Hashwert, unsicher!
- SHA (secure hash algorithm)
  - SHA1: 160 Bit Hashwerte unsicher!
  - SHA2
    - SHA224, SHA256, SHA 384, SHA512
  - SHA3: grundlegend anders, 224, 256, 384 Bitwerte → auch frei wählbar
- GOST, Whirlpool

## Ablauf SHA-256

- Block erstellen (Auffüllen, Startblock erstellen, Wurzel von Primzahlen)
- Bitrotation, Zeilen integrieren (Diffusion)
- XOR, Bitshift (viele Runden)
- Auswahlfunktion, Mehrheitsfunktion (Einwegfunktion)

**Zusatz:** Message Authentication Code z.B. HMAC: Schlüssel-Hash-Nachrichtauthentifizierung, Pre-Shared-Key

## Angriffe

- Brute-Force
- Phising
- Wörterbuchangriff
- Algorithmus nutzen
- Rainbow-Table (viel Speicher)
  - viele Hashwerte als Kette gespeichert

## 8.2 Passwörter

### 1. lokale Speicherung

- Länge des Passworters (mind 12 Zeichen)
- keine Wörter, persönliche Informationen
- Buchstaben/Zahlen/Symbole
- Jedes Passwort nur einmal verwenden
- Passwortmanager verwenden oder MFA als Alternative

### 2. Speicherung am Server

- **Klartext** ↳ Zugriff auf Datenbank, Admin, MITM
- **Gehashed** ↳ MITM, gleiche Passwörter erkennbar
- **Gehashed + Salt**: Mitm, Brute-Force
- **Gehashed + Salt + Pepper**  
**Salt**: Zufällige Zeichenkette die im Klartext in der Datenbank gespeichert wird → Jeder bekommt eigenen Hashwert  
**Pepper**: Zufällige Zeichenkette die NICHT in der Datenbank steht

### 3. Austausch Client-Server

- PAP unsicher!

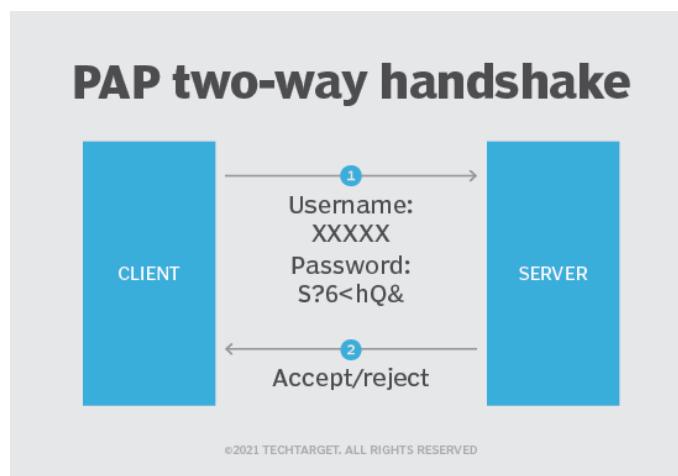


Abbildung 8.2: PAP

- CHAP: Bei CHAP wird bei jedem Anmeldeversuch eine Challenge (Zufallszahl) gesendet. Der Client hashed sein Passwort mit der Zufallszahl und sen-

det es dem Server. Der Server kann dann das Passwort in der Datenbank mit der gesendeten Zufallszahl hashen und es mit dem Client-Hash vergleichen. Somit wird zum einem, das Passwort nie im Klartext gesendet und zum anderen kann der gesendete Hash nicht noch einmal gesendet werden, da die Zufallszahl 'einzigartig' ist und bei jedem Anmeldeversuch anders ist.  
→ MITM Anmeldung mit dem Hashwert ist durch die Challenge nicht mehr möglich

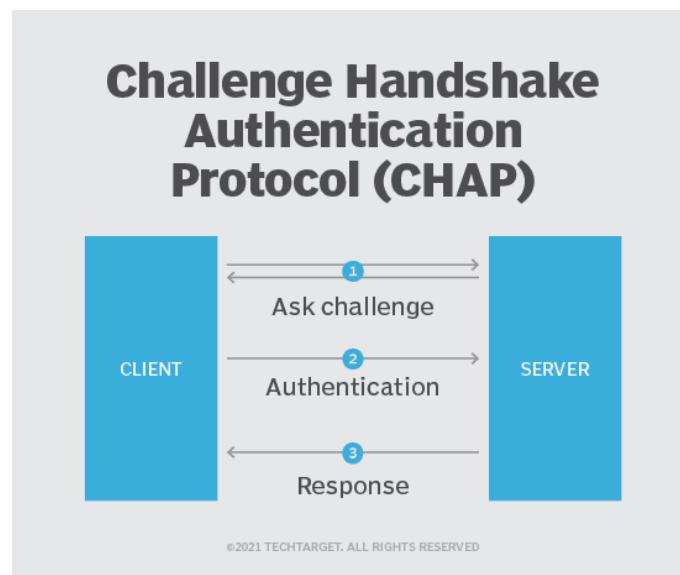


Abbildung 8.3: PAP

**Alternativen:** MS-CHAPv1, MSCHAPv2, EAP, PEAP,...

## 8.3 AAA

Autorisierung (Was?), Authentifizierung (Wer?) & Accounting (Wann?)  
Protokolle: RADIUS, TACACS+

## 8.4 PKI (Public Key Infrastruktur)

Digitale Zertifikate sind für die Authentifizierung eines öffentlichen Schlüssels und seiner zulässigen Anwendung bzw. Geltungsbereich

Vergleich:  
Zertifikat → Pass  
digitale Signatur → Foto

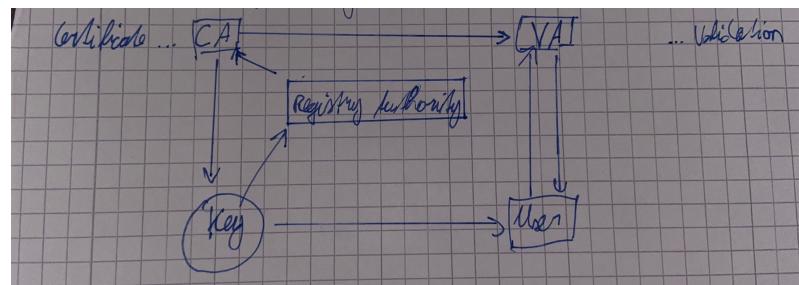


Abbildung 8.4: Public Key Infrastruktur

**Alternative:** Web of Trust

## 8.5 Zusammenfassung

verwendete Algorithmen	typische Protokolle
AES/DES RSA Diffie-Hellman Hashfunktionen (SHA256,...) MAC (bzw. HMAC) Authentifizierung (AAA, CHAP) PKI	HTTPS IPsec VPN SSH PSK (WLAN) RADIUS ...

## 8.6 HTTPS

HTTP + TLS (SSL)

TLS ... transport layer security, Verschlüsselung nach Layer 4

- Symmetrische Verschlüsselung
- Asymmetrischer Schlüsselaustausch (ab TLS 1.3 Diffie-Hellman)
- Authentifizierung (Server), PKI, RSA
- Hashfunktionen (SHA256,...)
- Sicherung der Nachrichtenintegrität (HMAC)

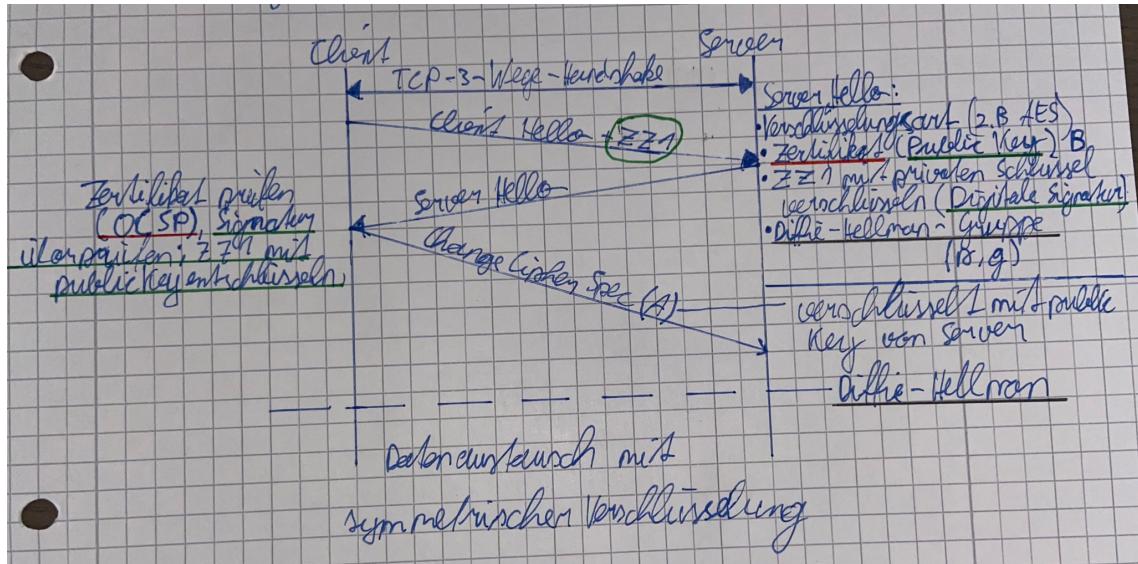


Abbildung 8.5: HTTPS Verbindungsauflauf

## Teil II

## SESD

## 9 Aspekte der Datensicherheit

John McCumber erstellte ein Würfelformat, welches drei Dimensionen der Datensicherheit darstellt (Cybersecurity Cube, McCumber Cube).

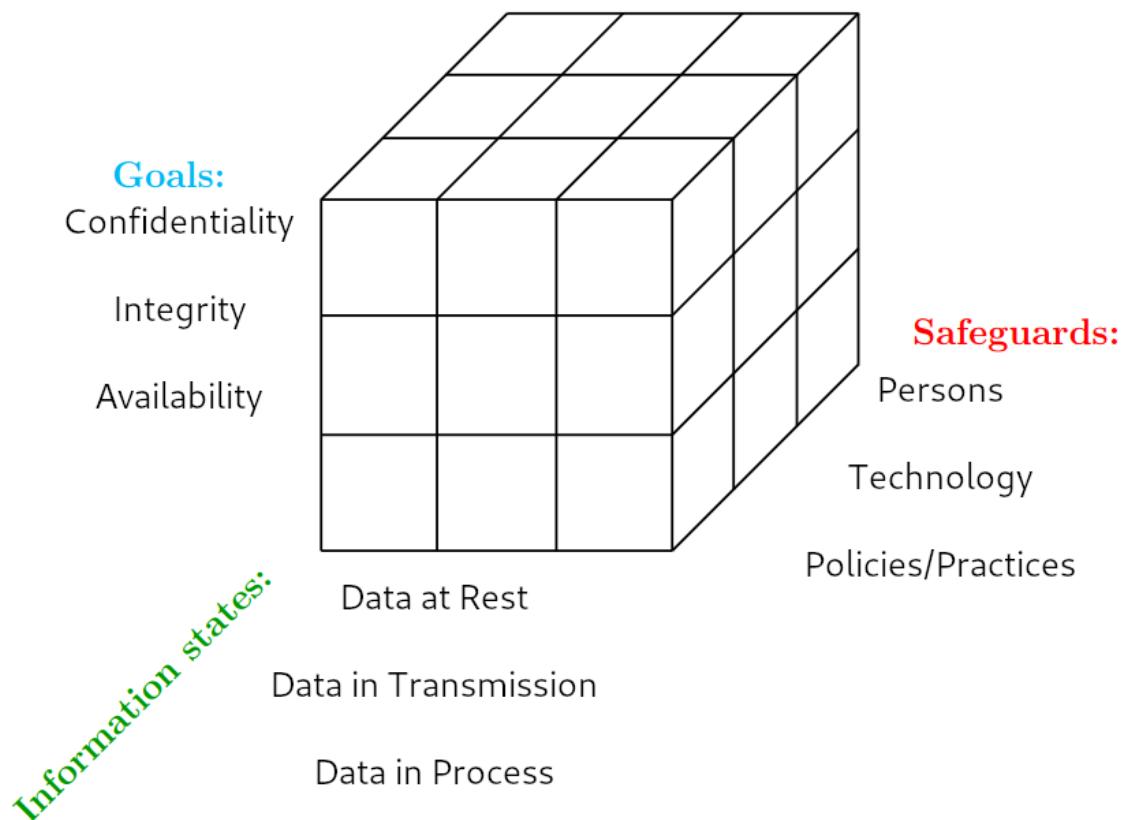


Abbildung 9.1: John McCumber Cybersecurity Cube

Die dreidimensionale Darstellung des Würfels zeigt, dass die Faktoren zusammenhängen. Dementsprechend müssen alle Faktoren und ihre Verbindungen untereinander berücksichtigt werden. Der Würfel soll zu einer methodischen Vorgehensweise verhelfen.

- Es ist nicht ausreichend, wenn Daten auf der Festplatte verschlüsselt abgespeichert sind, diese aber unverschlüsselt über das Internet versendet werden

- Ohne Policy mit Richtlinien für regelmäßige Sicherungen wird die Verfügbarkeit kaum gewährleistet, auch wenn Personen gut geschult sind und Technologie (Firewall) von Angriffen schützt (z.B. Feuer)

- **Safeguards (Schutzmaßnahmen)**

- **Persons:** Bewusstseinsschulung zum Umgang mit Daten und Geräten, Erkennung von Social Engineering,...
- **Policies:** Definition von (in-)akzeptablen Verhalten und Ablauf bei einem Vorfall. Passwortrichtlinien, wer darf wohin, Konsequenzen definieren,...
- **Technology:** Software & Hardware: Firewall, IDS, physische Zugangskontrollen,...

- **Information States (Zustände)**

- **Data at Rest:** gespeicherte Daten auf HDD, SSD, USB-Stick, RAID, NAS, Cloud,...
- **Data In Process:** Dateneingabe, -ausgabe und -veränderung; Datenformat, Eingabefehler → Schutz gegen z.B. SQL-Injections
- **Data in Transmission:** Datenversendung über LAN, WLAN, Sneakernet (physischer Transport von Datenträgern mittels PKW)

- **Goals (Ziele)**

- **Confidentiality (Vertraulichkeit):** Zugriffskontrolle, Verschlüsselung, Datensparsamkeit, Anonymisierung, Tokenization, Steganographie,...
- **Integrity (Integrität):** Konsistenzchecks, Prüfsummen
- **Availability (Verfügbarkeit):** häufige Probleme: DoS-Angriffe, Stromausfall, Naturkatastrophen; Sicherstellung durch Back-Ups, Redundanz, Abwehrmaßnahmen, Wartung/Reinigung von Hardware, Wiederherstellungspläne, Tests,...

Weitere Ziele

- **Authenticity (Authentizität):** Nachricht kommt vom tatsächlich behaupteten Sender
- **Non-repudiation (Nichtabstreitbarkeit):** Handlungen lassen sich nicht Abstreiten z.B. durch digitale Unterschrift
- **Accountability (Zurechenbarkeit):** Handlung einer bestimmten Person zuordnen/zurechnen
- **Anonymity (Anonymität):** Aktivisten, Journalisten, Whistleblower,...

# 10 Kryptologie

Die Kryptologie ist die Wissenschaft der Verschlüsseln. Sie wird unterteilt in:

- **Kryptographie:** Verschlüsseln und Entschlüsseln
- **Kryptoanalyse:** kryptographische Verfahren berechnen

## Wozu dient Verschlüsselung?

Es wird verschlüsselt um das Ziel der **Confidentiality (Vertraulichkeit)** zu erreichen. Die asymmetrische Kryptographie kann aber auch zum Erlangen von Authentizität und Nichtabstrebbarkeit verwendet werden.

## Wie funktioniert Verschlüsselung?

Alice möchte eine Nachricht (Klartext) an Bob schicken, ohne dass unberechtigte Zuhörer wie Eve den Chiffretext nicht verstehen.

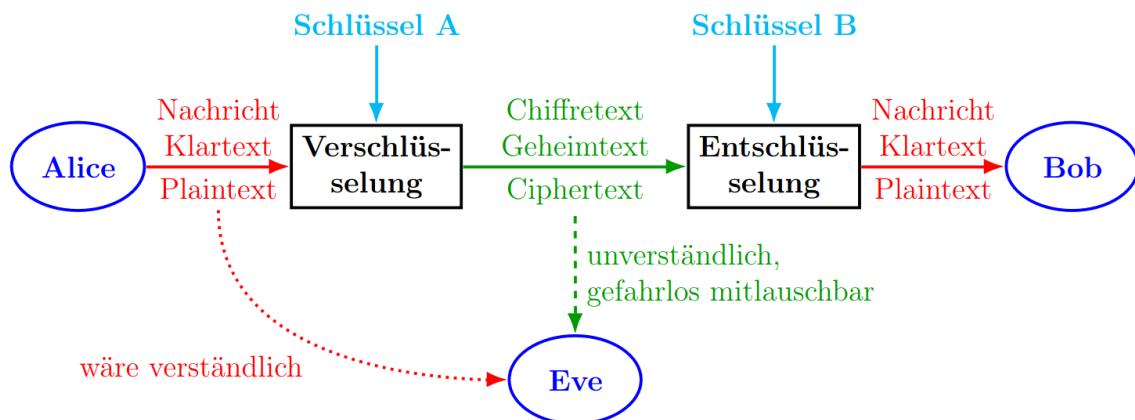


Abbildung 10.1: Grundlegende Situation der verschlüsselten Kommunikation

Die Daten werden so verändert, dass sie für Eve nicht rekonstruierbar sind, für Bob aber schon. Daher benötigen Alice und Bob für das Ver- und Entschlüsseln der Nachricht einen Schlüssel.

## Geschichtliche Einordnung

Der Wunsch, militärische, politische oder persönliche Geheimnisse vor Unbefugten zu schützen, ist mit Sicherheit alt. Einfache Verfahren dazu sind z.B. von Caesar bekannt. Es gab neuzeitlich Verfahren, die sich lange Analyse widersetzen (z.B. Vigenère).

Während des 1. und 2. Weltkrieges wurde die Kommunikation zwischen den Truppen verschlüsselt, mitgehört, geknackt, anders verschlüsselt, usw. Im 2. Weltkrieg wurde erstmals MathematikerInnen miteinbezogen (England: Alan Turin, Analyse von Enigma). Im Gegensatz zu früher, als die Kryptoanalyse eher von SprachwissenschaftlerInnen betrieben wurde, wird sie heute von InformatikerInnen und MathematikerInnen durchgeführt.

Früher war der Bedarf von Verschlüsselung für Herrschende und Regierungen vermutlich höher als für Privatpersonen. Heutzutage ist sie aber auch für Privatpersonen wichtig.

Die Zeit der asymmetrischen Kryptosystemen, die das Schlüsselaustauschproblem lösten, begann 1976. Mit ihnen wurde es möglich, dass 2 Personen über das Internet abhörsicher kommunizieren können.

## Wie wird die Unverständlichkeit für Eve erreicht?

Früher wurde vor allem darauf gesetzt, dass Unbefugten das Verfahren selbst unbekannt war. Dies hat viele Nachteile:

- Funktioniert zwischen zwei Verfeindeten Fraktionen (Freund-Feind-Schema), aber nicht wenn 2 Personen kommunizieren wollen, da man für jedes ein eigenes Verfahren benötigen würde
- Entwicklung und Beibringen von neuen Verfahren ist aufwendig
- Die Geheimhaltung des Verfahrens ist, besonders bei großen Maßstäben, schwierig (Verreden, Gefangennahme, Folter, Bestechung, Drohung).
- Selbst entwickelte Verfahren haben mit hoher Wahrscheinlichkeit Schwachstellen. Moderne Verfahren sind öffentlich bekannt und seit Jahrzehnten erforscht, analysiert und diskutiert.

Daher folgen heute anerkannte Verfahren (DES, AES, RSA,...) dem **Prinzip von Kerkhoff**: *"Das Verfahren darf nicht der Geheimhaltung bedürfen und soll ohne Schaden in Feindeshand fallen können."*

Moderne Verfahren sind so sicher, dass sie vor Angriffen sicher sind, auch wenn die Angreifer das Verfahren kennen. Die Sicherheit liegt in der Geheimhaltung des Schlüssels.

Schlüssel lassen sich leichter erzeugen, verbreiten und geheimhalten als Verfahren. Außerdem kann man für Kommunikation mit mehreren Personen einfach verschiedene Schlüssel verwenden.

## Key Management

Auch wenn es einfacher ist den Schlüssel als wie das Verfahren geheimzuhalten, ist das Key Management heikel. Hier geht es um das Erzeugen von Schlüsseln, ihren Austausch, Speicherung, Nutzung und Vernichtung. Moderne Chiffren sind so gut entworfen, dass Attacken meist auf Social Engineering, Seitenkanalangriffe oder am Key Management basieren.

## Wie hört Eve überhaupt ab?

Aus Glasfaserkabeln wird Licht gestreut

Bei WLAN und anderen drahtlosen Übertragungen: Broadcast → Frequenzen

Weiter Möglichkeiten sind Man-in-the-Middle Angriffe, DNS/DHCP/ARP-Spoofing

## 10.1 Symmetrische und Asymmetrische Kryptosysteme

Es gibt 2 verschiedene Möglichkeiten für die Schlüssel von *Alice* und *Bob*:

1. Beide besitzen den **gleichen** Schlüssel → **symmetrische Kryptosystem**

- Daher müssen beide den Schlüssel geheim halten und kennen, da sonst Eve Nachrichten entschlüsseln oder Mallory Nachrichten verschlüsseln und versenden könnte.
- Beide können Nachrichten **ver- und entschlüsseln**
- Typische Schlüssellängen sind **80-256 Bit**
- Die Algorithmen sind typischerweise schnell, da sie einfache Operationen (Addition, XOR, Bitshifts, Substitutionen,...) verwenden.
- Alle symmetrischen Kryptosysteme haben das Problem des **Schlüsselaustausches**

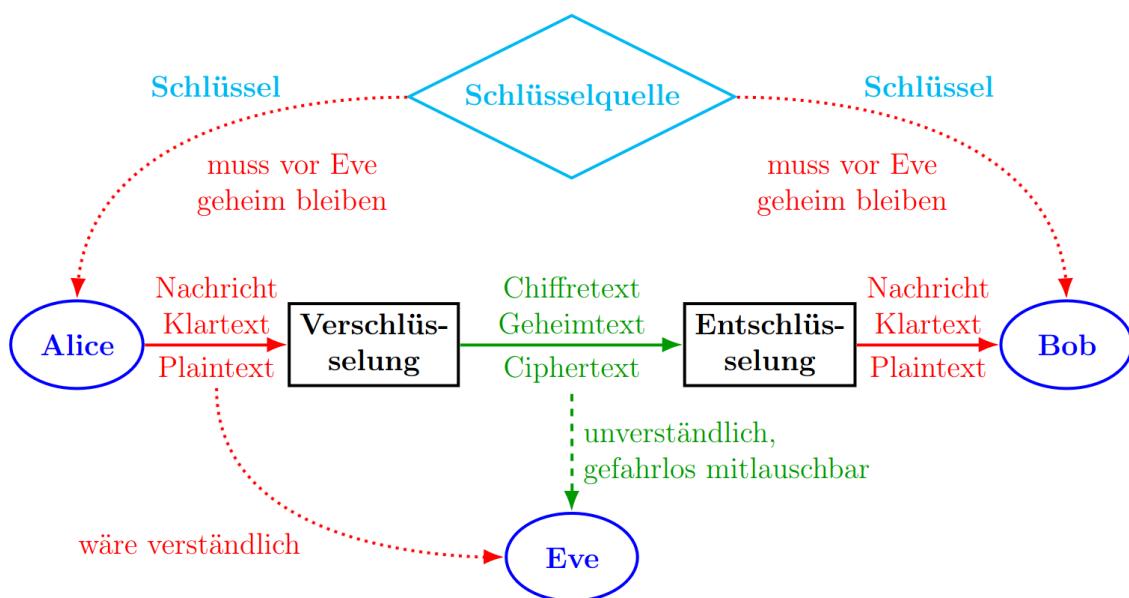


Abbildung 10.2: Symmetrische Kryptosysteme

2. Beide besitzen **verschiedene** Schlüssel → **asymmetrische Kryptosystem**

- *Bob* hat einen geheimen Schlüssel, mit dem die Nachricht entschlüsselt werden kann. Der Schlüssel zum verschlüsseln ist jedem (öffentlich) bekannt. So kann Eve Nachrichten verschlüsseln, Mallory aber nicht entschlüsseln.

- Alice kann ihre Nachrichten verschlüsseln, aber nicht entschlüsseln
- Typische Schlüssellängen sind **512-4096 Bit**
- Die Algorithmen sind typischerweise langsam, weil sie auf 'schwierigere' Operationen verwenden (z.B. Potenzieren).
- Da Alice nur den öffentlichen Schlüssel benötigt, lösen asymmetrische Kryptosysteme das Schlüsselaustauschproblem

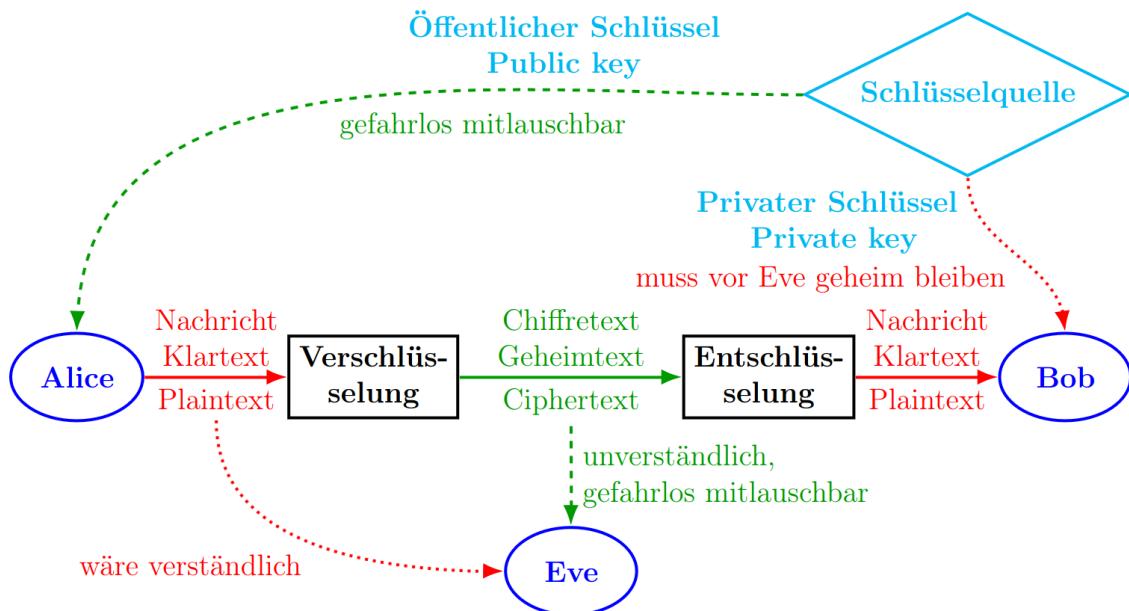
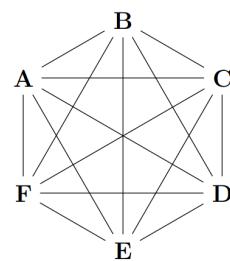


Abbildung 10.3: Asymmetrische Kryptosysteme

Neben den bisher erwähnten Unterschieden zwischen symmetrischen und asymmetrischen Kryptosystemen gibt es noch einen weiteren Aspekt: die Anzahl der benötigten Schlüssel, um mit vielen Personen zu kommunizieren.



Eine Gruppe aus 6 Personen möchte verschlüsselt kommunizieren. Dabei will jede Person in der Lage sein, mit jeder anderen Person privat zu kommunizieren.

1. Anzahl der verschiedenen Schlüssel: symmetrische Kryptosysteme  
 $\frac{6 \cdot 5}{2} = 15$
2. eine Person muss 5 Schlüssel speichern (symmetrisch)
3. Anzahl der verschiedenen Schlüssel: asymmetrische Kryptosysteme  
 $6 \cdot 2 = 12$
4. eine Person muss 1 Schlüssel speichern (asymmetrisch)
5. Bei n Personen

Symmetrisch:  $\frac{n^2}{2}$  Schlüssel insgesamt,    n-1 Schlüssel speichern

Asymmetrisch:  $2 \cdot n$  Schlüssel insgesamt,    1 Schlüssel speichern

## 10.2 Angriffsarten

In diesem Abschnitt geht es um die möglichen Angriffe auf Kryptosysteme, die teilweise danach eingeteilt werden, was der Angreifer weiß und kann. Es wird immer davon ausgegangen, dass der Angreifer das verwendete Verfahren kennt, weiß, wie Schlüssel aussehen, etc.

$k$  ... verwendeter Schlüssel

$m_i$  ... die  $i$ -te Klartext Nachricht

$c_i$  ... den zugehörigen Chiffertext

### 10.2.1 Ciphertext-Only Angriff

Eve hat nur die Ciphertexte  $c_i$ ,  $i = 1, \dots, n$ , zur Verfügung. Mögliche Ziele: Klartextnachricht herausfinden, Schlüssel herausfinden oder Ciphertext zu Klartext zuordnen.

Von dieser (schwachen) Angriffsmöglichkeit ist immer auszugehen, weile Eve nur die Nachrichten abfangen und analysieren muss. Analysemethoden: Brute-Force, Wörterbuchattacke, Statistik/Muster.

### 10.2.2 Known-Plaintext Angriff

Eve hat neben den Ciphertexten  $c_i$ ,  $i = 1, \dots, n$ , auch die Klartexte  $m_i$  zur Verfügung oder Ahnung davon. Mögliche Ziele: Schlüssel herausfinden oder Ciphertext zu Klartext zuordnen.

Wieso sollte Eve die Klartexte kennen?

- Sie fängt mehrere Nachrichten ab, beobachtet das Verhalten des Empfängers und schließt so auf den Inhalt der Nachricht. ("Finden Sie sich um 17:00 im Park ein", "Finden Sie sich um 12:00 im Park ein" → Nachrichtenteile die sich verändern und welche die sich nicht verändern)
- Eve vermutet, dass das immer gleiche Ende eines Briefe "Beste Grüße" heißen könnte.
- Regelmäßige Berichte zur gleichen Uhrzeit in gleicher Form z.B. Wetterbericht beim Militär
- Funksprüche, welche melden, dass sie nichts zu melden haben

### 10.2.3 Chosen-Plaintext Angriff

Der Angriff entspricht dem Known-Plaintext Angriff, bloß dass *Eve* die Möglichkeit hat, die Klartexte  $m_i$  vorzugeben oder zu beeinflussen. Wie kann *Eve* *Alice* dazu bringen Klartexte zu verschlüsseln und an *Bob* zu senden?

- Durch inszenieren berichtet ein feindlicher Agent etwas
- England 'sätze' Gebiete in der Nordsee mit Minen. Die darauffolgenden Nachrichten der Deutschen wird höchstwahrscheinlich Koordinaten der Gebiete beinhalten.
- Die Amerikaner wussten, dass Japan 'AF' angreifen will und vermuteten, weil andere hawaiianische Inseln mit 'A' begannen, dass es sich um Midway handeln könnte. Sie sendeten eine Nachricht von Midway, dass die ein angeblicher Bedarf an Vorräten anliegt. Daraufhin sendeten die Japaner eine Nachricht, dass 'AF' Vorräte benötige.
- Ein Chosen-Plaintext Angriff ist bei asymmetrischen Kryptosystemen immer möglich, weil der Schlüssel, der zum Verschlüsseln verwendet wird öffentlich bekannt ist. So könnte *Eve* typische Nachrichten ('ja', 'nein',...) verschlüsseln und mit dem Ciphertext vergleichen.

### 10.2.4 Brute-Force Angriff

Dieser Angriff bezeichnet das bloße Durchprobieren aller möglichen Schlüssel. Dieser Angriff ist immer möglich. Ihm abzuwehren gelingt durch lange Schlüsselräume.

### 10.2.5 Wörterbuchangriff

Wenn die Schlüssel nicht alle gleich wahrscheinlich sondern von Menschen ausgewählt sind (besonders bei Passwörtern), ist es vernünftig, wahrscheinliche Schlüssel zuerst durchzuprobieren (Wörter aus Wörterbüchern, Kombinationen, Falschschreibungen, Passwortlisten,...).

### 10.2.6 Replay Angriff

*Mallory* hört die verschlüsselte Nachricht an *Bob* ab und sendet sie später, ohne sie zu verstehen, wieder an *Bob*. (Wenn *Alice* einen Aktienkauf autorisiert hat, muss sie nun das Vielfache bezahlen).

Abwehren lassen sich Replay Angriffe durch Zeitstempel oder Schlüsselwechsel nach mehreren Nachrichten.

### 10.2.7 Man-in-the-Middle Angriff

*Mallory* schaltet sich zwischen *Alice* und *Bob*, so dass *Alice* denkt, sie würde mit *Bob* kommunizieren und umgekehrt. In Wahrheit interagieren beide mit *Mallory*, die die Nachrichten von *Alice* und *Bob* lesen kann und weiterleitet. *Alice* und *Bob* merken nicht, dass ihre Kommunikation abgehört wird, zumindest bis *Mallory* mit dem Angriff aufhört. (Bei Unterbrechung der Kommunikation wird *Mallory* auffliegen)

### 10.2.8 Verkehrsflussanalysen

Eve bekommt mit, dass *Alice* und *Bob* kommunizieren (wie oft, wann, welcher Rhythmus,...). Dies ist dann schon sehr aussagekräftig. (zwei CEOs bei Fusion, Preisabsprache,...)

Verkehrsflussanalysen können prinzipiell nicht abgewehrt werden. Jedoch können versendete Dummy-Nachrichten die Kommunikation verschleiern.

### 10.2.9 Harvest-now-decrypt-later Angriff

Eve hört jetzt die Nachrichten ab und knackt die Verschlüsselung später wenn sie bessere Werkzeuge hat. Deswegen muss man sich auch heute bereits mit 'Quantum-Ready' Verschlüsselungen beschäftigen.

### 10.2.10 Seitenkanalangriffe

Darunter werden Angriffe verstanden, die nicht das Verfahren an sich sonder seine Nutzung, Umsetzung oder Implementierung mit scheinbar nutzloser Zusatzinformationen angreifen.

- **Zeitangriff:** 1996 wurde RSA geknackt, indem die Zeit für die Verschlüsselung gemessen wurde. Die Nachricht wurde variiert und aus der Dauer auf die Anzahl der 1en im Exponenten (privater Schlüssel) geschlossen. Einige 1.000 Messungen reichen, um RSA mit 1024 Bit zu knacken.  
Abgewehrt kann der Angriff, indem zufällige Zahlen hinein- und später wieder herausgerechnet oder künstliche Zeitverzögerungen eingebaut werden.
- **Stromangriff:** Ist die Verschlüsselung in Hardware implementiert, sind die Multiplikationen von den Quadrierungen im Square-and-Multiply Algorithmus am Oszilloskop gut unterscheidbar. Ebenso Substitutionen und Permutationen.  
Durch parallel dazu laufende Dummy-Operationen lässt sich dies vermeiden.

- **Elektromagnetische Strahlung:** Elektronische Geräte emittieren elektromagnetische Strahlung, aus der sich auf die Operationen schließen lassen könnte. TEMPEST ist der Deckname für ein Abhörverfahren bzw. Spezifikation, die die verhindern soll.  
 Im Botschaftsgebäude gibt es speziell abgeschirmte Räume (Faraday'sche Käfige). Auch bei Mils Electronic gab es so etwas, zusätzlich wurden Signale im Kabeln gefiltert, damit Tastendrücke nicht im Stromnetz aufgeprägt werden.
- **Eingebaute Hintertür:** Es wird absichtlich eine Schwachstelle eingebaut. Verhinder lässt sich das teilweise durch offengelegte Verfahren und Quellcodes.  
 Das FBI bauten eine Hintertür in eine App welche Organisierten Verbrechern zur Kommunikation verwendeten (Operation Trojan Shield).
- **Implementierungsfehler:** Nicht alle Fehler fallen sofort auf, RSA funktioniert auch mit schlechten Zahlen. Besonders anfällig: Verfahren 'ohne Umkehrung' wie Hashfunktionen und Zufallszahlgeneratoren.
- **Das Problem des sicheren Löschens:** Daten zu verschlüsseln ist völlig nutzlos, wenn sich die Originale nicht sicher löschen kann? Dazu: temporäre Dateien im Betriebssystem.

### 10.2.11 Angriffe auf die BenutzerInnen

**Folter, Erpressung, Bestechung,...** Die meisten Geldautomatenbetrüger sind Insider (Bankangestellte, Techniker,...). Ebenso kommt der Großteil der IT-Angriffe von innen! Abgemildert kann dieses Problem durch Rollentrennung oder Logs/Protokolle.

**Schwachstelle Mensch:** Menschen nutzen aus Faulheit schwache Passwörter oder haben sie auf Post-its am Schreibtisch, chiffrieren nicht, übertreiben Sorgfalt (immer die gleiche Anrede mit Titeln, militärische Sprache), bedienen Hard- und Software falsch, **Social Engineering**,...

Ein häufiges Verständnisproblem ist, zu denken, dass Verfahren 'doch irgendwie unsicher' sind, die sich unter Einsatz aller Computer in Jahrzehnten brechen lassen.

- Angriffe kosten Geld
- Angriffe werden nicht durchgeführt, wenn sie teurer sind als was sie preisgeben
- es gibt meist einfachere und billigere Möglichkeiten
  - Wohnungstür mit Stahlkern bei Terrassentür aus Glas
  - Iris- und Fingerabdruckscanner bei Tür mit Katzenklappe

- 10-stellige PINS, welche auf die Karte geschrieben werden
- Rollenspiel-Zitat: *"If you find yourself in the company of a halfling and an ill-tempered dragon, remember that you do not have to outrun the dragon; you simply have to outrun the halfling"*
- die einfachsten, billigsten und effektivsten Angriffe sind **Social Engineering**

# 11 Mathematische Grundlagen der Kryptographie

## 11.1 Modulo

$a$  ganzzahlig durch  $m$  dividieren  $\rightarrow$  ganzzahligen Quotienten  $q$  und den Rest  $r$ :

$$a = q \cdot m + r$$

$a$  ... Ganzzahl

$m$  ... Modul

Für den Rest dieser Gangzahldivision:  $r = a \bmod m$  (" $a$  modulo  $m$ )

Beachte:

- Es gibt nur  $m$  verschiedene Reste bei ganzzahliger Division durch  $m$
- der Rest  $r$  ist nicht negativ! Das gilt auch bei  $a < 0$

$$13 \bmod 3 = 1 \quad 13 = 4 \cdot 3 + 1$$

$$42 \bmod 5 = 2 \quad 42 = 8 \cdot 5 + 2$$

$$42 \bmod 9 = 6 \quad 42 = 4 \cdot 9 + 6$$

$$-13 \bmod 3 = 2 \quad -13 = -5 \cdot 3 + 2$$

$$-42 \bmod 9 = 3 \quad -42 = -5 \cdot 9 + 3$$

$$-36 \bmod 9 = 0 \quad -36 = -4 \cdot 9 + 0$$

$\bmod$  ist eine **Rechenoperation**. Sie erhält zwei Argumente,  $a$  und  $m$ , und liefert eine Zahl  $r$  als Ergebnis.

## 11.2 Kongruenz modulo $m$

Wenn 2 ganze Zaheln ( $a, b$ ) nach Division durch  $m$  den gleichen Rest haben, nennt man sie **kongruent modulo  $m$**  und schreibt:

$$a \equiv b \pmod{m}$$

Beispiel:  $72 \equiv 12 \pmod{10}$

Andere Modulo  $m$  sodass  $72 \equiv 12 \pmod{m}$ :

2, 3, 4, 5, 6, 10, 12, 15, 30, 60

Wir sehen, dass man  $a$  und  $b$  in gewissem Sinne als gleich ansehen kann. Deshalb verwendet man ein Symbol, das an Gleichheit erinnert. Bei der Kongruenz modulo  $m$  handelt es sich nicht um eine Rechenoperation sondern um eine **Aussage**, dass sich nämlich  $a$  und  $b$  nur um ein Vielfaches des Moduls  $m$  unterscheiden:

$$a \equiv b \pmod{m} \Leftrightarrow a - b = k \cdot m$$

Welche der folgenden Ausdrücke sind wahr?  $25 \equiv 0 \pmod{5} = \checkmark$

$$25 \equiv 20 \pmod{5} = \checkmark$$

$$25 \equiv 20 \pmod{10} = \times$$

$25 \bmod 5 = \text{kein Sinn}$

Berechne, falls möglich  $25 \equiv 10 = \times$

$$25 \bmod 10 = 5$$

## 11.3 Uhrenarithmetik

Um die Rechenregeln im Zusammenhang mid modulo kennenzulernen, rechnen wir mit Uhrzeiten (modulo 24).

- Ein Fertigungsprozess startet um 17 Uhr und dauert 21h. Um welche Uhrzeit endet er?

$$17 + 21 = 38 \equiv 14 \pmod{24} \rightarrow \text{er endet um 14 Uhr}$$

- Ein Fertigungsprozess startet um Mitternacht und dauert 38h für den ersten Arbeitsschritt und dann noch einmal 40h für den zweiten.

$$38 + 40 = 78 \equiv 6 \pmod{24}$$

Alternativ:

$$38 + 40 \equiv 14 + 40 \pmod{24}$$

$$38 + 40 \equiv 14 + 16 \pmod{24}$$

$$38 + 40 \equiv 30 \pmod{24}$$

$$38 + 40 \equiv 6 \pmod{24}$$

Wir erkennen: **Bei der Addition modulo  $m$  ist es egal, ob vor dem Addieren oder in Zwischenschritten um  $m$  reduziert wird**

- Ein Arbeitsschritt dauert 22h und muss mit Start um Mitternacht 7-mal durchgeführt werden. Um wie viel Uhr ist er zu Ende?

$$22 \cdot 7 = 154 \equiv 10 \pmod{24}$$

Alternativ:

$$22 \cdot 7 \equiv (-2) \cdot (mod 24) \equiv -14 \pmod{24} \equiv 10 \pmod{24}$$

Wir erkennen: **Bei der Multiplikation modulo  $m$  ist es egal, ob vor dem Multiplizieren oder in Zwischenschritten um  $m$  reduziert wird.**

- Daraus folgt, dass **auch beim Potenzieren die Basis im Voraus um  $m$  reduziert werden darf.**

$$3^5 \bmod 2 = 1^5 \bmod 2 = 1 \bmod 2 = 1$$

$$(12 + 20) \bmod 3 = (0 + 2) \bmod 3 = 2$$

$$(81 - 40) \bmod 7 = (4 - 5) \bmod 7 = -1 \bmod 7 = 6$$

$$(23 \cdot 15) \bmod 4 = (3 \cdot 3) \bmod 4 = 9 \bmod 4 = 1$$

$$(15 \cdot 16 \cdot 17) \bmod 5 = (0 \cdot 1 \cdot 2) \bmod 5 = 0$$

$$(18 + 34 \cdot 23) \bmod 8 = (2 + 2 \cdot 7) \bmod 8 = (2 + 14) \bmod 8 = 16 \bmod 8 = 0$$

$$17^8 \bmod 7 = 3^8 \bmod 7 = 6561 \bmod 7 = 2$$

$$j^2 = -1$$

$$j^3 = -j$$

$$j^4 = 1$$

$$j^{10} = -1$$

$$j^{2024} = 1$$

$$j^{9876543210} = -1$$

$$(100 + 823) \bmod 5 = (0 + 3) \bmod 5 = 3$$

$$(12 + 5 \cdot 18) \bmod 7 = (5 + 5 \cdot 4) \bmod 7 = (5 + 20) \bmod 7 = (5 + 6) \bmod 7 = 11 \bmod 7 = 4$$

$$(23 \cdot 18) \bmod 5 = (3 \cdot 3) \bmod 5 = 9 \bmod 5 = 4$$

$$(19 \cdot 37 \cdot 22) \bmod 7 = (5 \cdot 2 \cdot 1) \bmod 7 = 10 \bmod 7 = 3$$

$$(41 \cdot 23 \cdot 25) \bmod 9 = (5 \cdot 5 \cdot 7) \bmod 9 = (25 \cdot 7) \bmod 9 = (7 \cdot 7) \bmod 9 = 49 \bmod 9 = 4$$

...

$$14^3 \bmod 13 = 1^3 \bmod 13 = 1$$

$$(212 \cdot 31^2) \bmod 21 = (2 \cdot 10^2) \bmod 21 = 200 \bmod 21 = 4 \cdot 50 \bmod 21 = 4 \cdot 8 \bmod 21 = 32 \bmod 21 = 11$$

$$(19^4 + 14 \cdot 25^5) \bmod 8 = (3^4 + 6 \cdot 1^5) \bmod 8 = 87 \bmod 8 = 7$$

## 11.4 Restklassenring $\mathbb{Z}_m$

Wenn man modulo  $n$  rechnet, gibt es immer nur  $n$  Zahlen. Daraus folgt (Beispiel modulo 24):  $\{\dots, -22, 2, -26, 50, \dots\}$ .

Dementsprechend kann man Zahlen die das selbe nach modulo rechnen ergeben wie folgt hinschreiben:  $\bar{2} = \{\dots, -22, 2, -26, 50, \dots\} = \overline{26}$

Man wählt einen Repräsentanten, Zahl mit Strich drüber.

Beispiel  $\mathbb{Z}_{24}$ :

Addition  $\{\dots, -19, 5, 29, \dots\}$  und  $\{\dots, -1, 23, 47, \dots\}$

$$\overline{5} + \overline{23} = \overline{28} = \overline{4} \text{ oder}$$

$$\overline{5} + 23 = \overline{5} + \overline{-1} = \overline{4}$$

## 11.5 Berechnung des ggT

Die berechnung des **größten gemeinsamen Teiler** kann durch unterschiedliche Möglichkeiten berechnet werden. Durchgeführt mit  $a = 2079$  und  $b = 735$ .

### 11.5.1 Primfaktorzerlegung beider Zahlen

Sehr aufwändig! Man zerlegt beide Zahlen in ihre Primfaktoren. Der ggT wird aus allen gemeinsamen Primfaktoren gebildet (Taucht ein Primfaktor mehrfach auf, wird er so oft verwendet, wie er in beiden Zahlen vorkommt):

$$2079 = 3 \cdot 693 = 3 \cdot 3 \cdot 231 = 3 \cdot 3 \cdot 3 \cdot 7 \cdot 11$$

$$735 = 3 \cdot 245 = 5 \cdot 3 \cdot 49 = 7 \cdot 7 \cdot 5 \cdot 3$$

$$\text{ggT}(2079, 735) = 3 \cdot 7 = 21$$

### 11.5.2 Euklidischer Algorithmus

Man subtrahiert immer die kleinere von der größeren Zahl und fährt mit kleinerer Zahl und Differenz fort. Die letzte Zahl, die nicht durch subtrahieren auf 0 führt, ist der ggT.

2079 - 735 = 1344	2079 mod 735 = 609
1344 - 735 = 609	735 mod 609 = 126
735 - 609 = 126	609 mod 126 = 105
609 - 126 = 483	126 mod 105 = 21
483 - 126 = 357	(105 mod 21 = 0)
357 - 126 = 231	
231 - 126 = 105	
126 - 105 = 21	
105 - 21 = 84	
84 - 21 = 63	
63 - 21 = 42	
42 - 21 = 21	
(21 - 21 = 0)	

$$\rightarrow \text{ggT}(1079, 735) = 21$$

Entweder durch Subtrahieren oder Modulo rechnen.

### 11.5.3 Erweiterter euklidischer Algorithmus

Mit dem erweiterten euklidischen Algorithmus kann das Invers einer Zahl berechnet werden. Hier wird das ggT als Vielfachsumme von  $a$  und  $b$  angeschrieben:

$$ggT(a, b) = s \cdot a + t \cdot b$$

Links wird der normale euklidische Algorithmus durchgeführt. Jedoch notieren wir, wie oft die kleinere Zahl in die größere passt. Diese Zahl (mal -1) multiplizieren wird mit der aktuellen Reihe und addieren die Multiplizierte Reihe mit der oberen.

$$\begin{array}{rcl} 2079 & = & 1 \cdot 2079 + 0 \cdot 735 \\ 735 & = & 0 \cdot 2079 + 1 \cdot 735 \quad | \cdot -2 \\ 609 & = & 1 \cdot 2079 + -2 \cdot 735 \quad | \cdot -1 \\ 126 & = & -1 \cdot 2079 + 3 \cdot 735 \quad | \cdot -4 \\ 105 & = & 5 \cdot 2079 + -14 \cdot 735 \quad | \cdot -1 \\ 21 & = & -6 \cdot 2079 + 14 \cdot 735 \\ 0 & = & \end{array}$$

$$\rightarrow ggT(2079, 735) = 21$$

Ein Beispiel zur Ermittlung eines Invers: Inverse von  $\bar{5}$  in  $\mathbb{Z}_{26}$

$$\begin{array}{rcl} 26 & = & 1 \cdot 26 + 0 \cdot 5 \\ 5 & = & 0 \cdot 26 + 1 \cdot 5 \quad | \cdot -5 \\ 1 & = & 1 \cdot 26 + -5 \cdot 5 \quad | \text{ fertig, da 1 links herauskommt} \\ \dots \text{ daraus} & & \end{array}$$

- $ggT(26, 5) = 1$ , daher ist  $\bar{5}$  in  $\mathbb{Z}_{26}$  invertierbar  
... würde anstatt 1 in der letzten Zeil 0 herauskommen, gäbe es kein Invers zu dieser Zahl
- das Inverse von  $\bar{5}$  ist  $\bar{-5} \rightarrow -5 + 26 = 21$

### 11.6 Eulersche Phi-Funktion

Die Phi-Funktion  $\varphi(n)$  gibt die Anzahl der invertierbaren Elemente von  $\mathbb{Z}_n$ .

- Ist  $n$  prim, gilt  $\varphi(n) = n - 1$
- Ist  $n$  nicht prim  $\rightarrow$  Primfaktorzerlegung,  $\varphi$  mit jeder Primzahl durchführen; bei mehrfachen vorkommen von Primzahlen nur einmal  $\varphi$  verwenden, anschließend alle multiplizieren. Beispiel  
 $\varphi(810) = \varphi(2 \cdot 3 \cdot 3 \cdot 3 \cdot 5) = \varphi(2) \cdot \varphi(3 \cdot 3 \cdot 3 \cdot 3) \cdot \varphi(5) = 1 \cdot 2 \cdot 3 \cdot 3 \cdot 3 \cdot 4 = 216$   
... 216 Zahlen sind in  $\mathbb{Z}_{810}$  teilerfremd und 216 invertierbare Elemente.

**Das Berechnen von Invers ist leicht. Das Bestimmen von wie viele invertierbare Elemente es gibt ist bei Primzahlen einfach, sonst aber (bei großen Zahlen besonders) schwer, da man die Primfaktorzerlegung durchführen muss.**

## 11.7 Einweg- und Falltürfunktionen

Eine Funktion weist zu jeder *Definitionsmenge* eine *Wertemenge*. Für die Verschlüsselung ist dies jedoch ungünstig, da man von der verschlüsselten Nachricht einfach auf dem Klartext kommen würde.

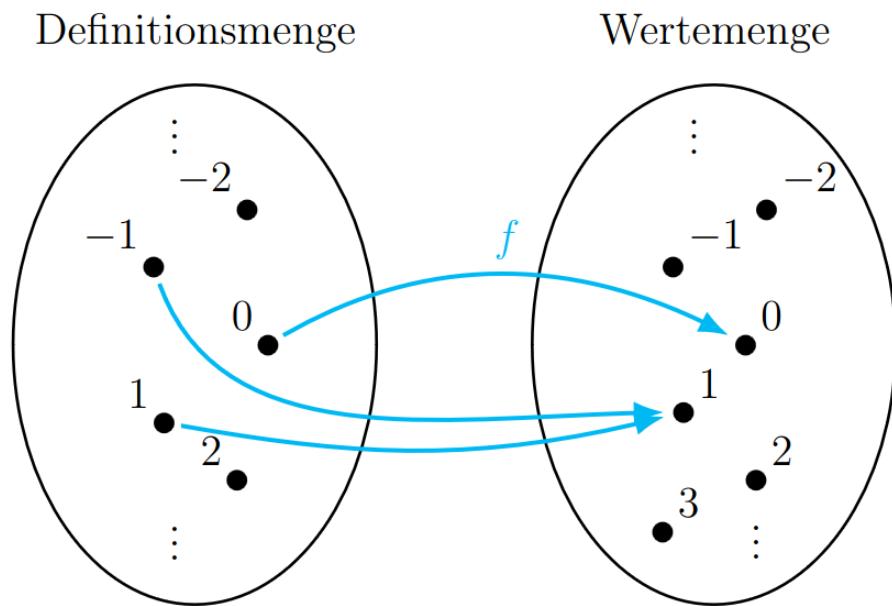
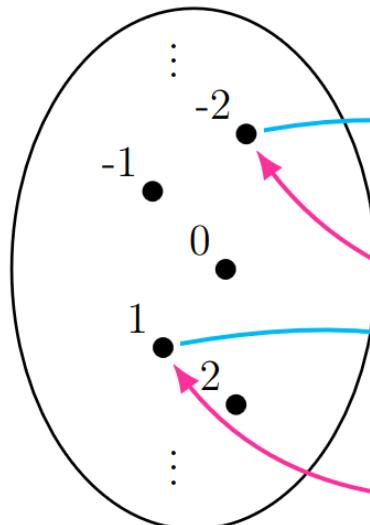


Abbildung 11.1: Definitionsmenge und Wertemenge einer normalen Funktion

Eine **umkehrbare Funktion** verwendet jede Wertemenge einmal. Daher ist das Umkehren aufgrund der Eindeutigkeit auch wieder einfach und für die Verschlüsselung ungünstig.

Definitionsmenge



Wertemenge

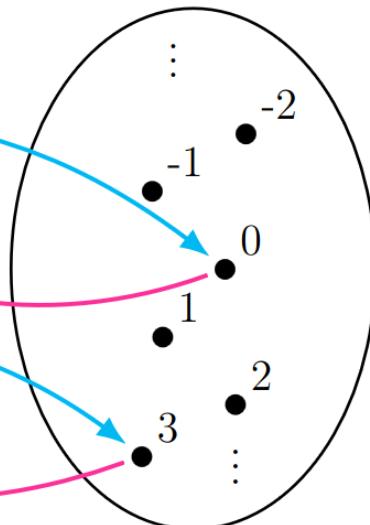
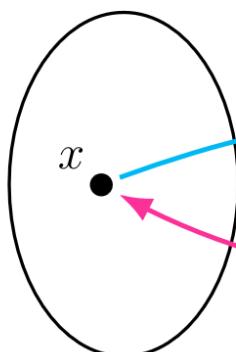


Abbildung 11.2: Definitionsmenge und Wertemenge einer umkehrbaren Funktion

Eine **Einwegfunktion** ist zwar theoretisch umkehrbar, dies ist aber sehr aufwändig.

Definitionsmenge



Wertemenge

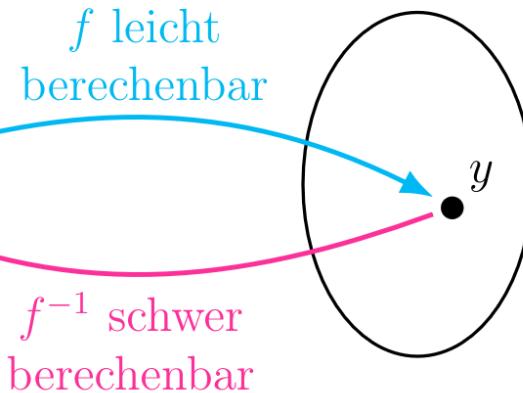


Abbildung 11.3: Definitionsmenge und Wertemenge einer Einwegfunktion

Zur Verschlüsselung kann eine Einwegfunktion jedoch auch nicht verwendet werden, da die Umkehrung sehr lange dauern würde (Beispiel Kommunikation zwischen *Alice* und *Bob*).

Eine **Falltürfunktion** ist allgemein sehr aufwändig zum umkehren, aber sehr einfach mit Zusatzinformation.

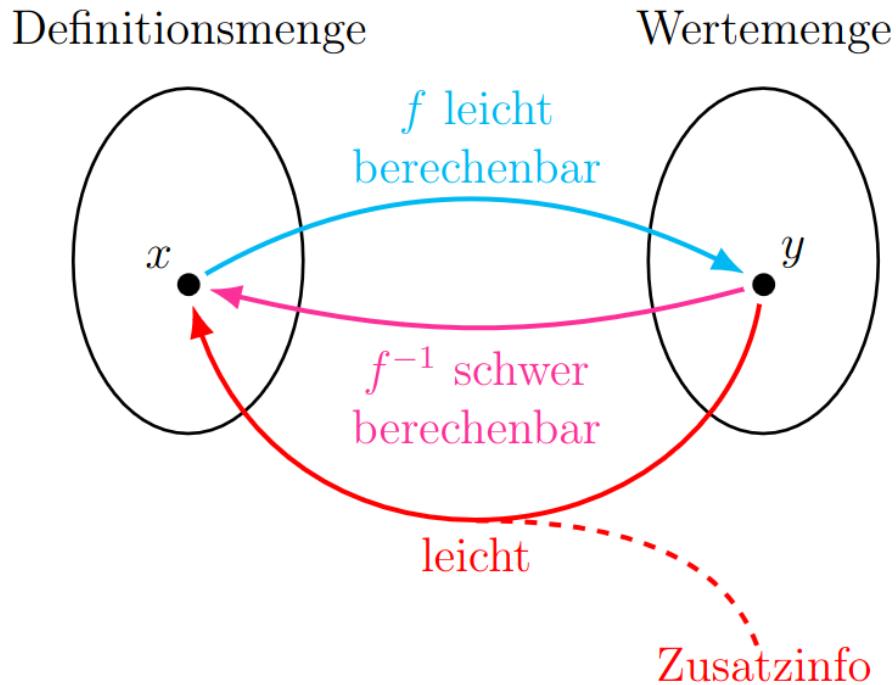


Abbildung 11.4: Definitionsmenge und Wertemenge einer Falltürfunktion

## 11.8 Modulares Potenzieren und diskreter Logarithmus

### 11.8.1 Modulares Potenzieren

$$\text{basis}^{\text{exponenten}} \bmod n$$

Bei Potenztabellen: Reihen mit 1 → invertierbar.

### 11.8.2 Diskreter Logarithmus

Log allgemein:  $b^x = a$

$x = \log_b(a)$  ... der Logarithmus von  $a$  zur Basis  $b$ . Logarithmusfunktion ist Umkehrfunktion von Exponentialfunktion

Diskrete Log:  $b^n = a$

$n = \log_b(a)$  ... der diskrete Logarithmus von  $a$  zur Basis  $b$ .

→ Berechnung von diskreten Logarithmus ist kompliziert. Mehrere Hochzahlen liefern dasselbe Ergebnis. Diskrete Log verwendet nur die kleinste Hochzahl.

Dadurch, dass nicht jedes Element invertierbar ist, haben viele Elemente **keinen** Logarithmus.

### 11.8.3 Aufwand des modularen Potenzieren

$174^{23^1} \text{ mod } 23 = 13^{23^1} \text{ mod } 23 = 13^{11} \text{ mod } 23 \dots$  sehr aufwendig → daher **Square-and-Multiply Algorithmus**

### 11.8.4 Square-and-Multiply Algorithmus

$$13^{11} \text{ mod } 23 = 13^{1011} \text{ mod } 23$$

$13^1$	$= 13^1$	$= 13$	$\xrightarrow{\text{mod } 23}$	13	$\xrightarrow{\text{Quadrieren}}$
$13^2$	$= 13^{10}$	$= 13 \cdot 13 = 169$	$\xrightarrow{\text{mod } 23}$	8	$\xrightarrow{\text{Quadrieren}}$
$13^4$	$= 13^{100}$	$= 8 \cdot 8 = 64$	$\xrightarrow{\text{mod } 23}$	18	$\xrightarrow{\text{Multiplizieren } 13}$
$13^5$	$= 13^{101}$	$= 18 \cdot 13 = 234$	$\xrightarrow{\text{mod } 23}$	4	$\xrightarrow{\text{Quadrieren}}$
$13^{10}$	$= 13^{1010}$	$= 4 \cdot 4 = 16$	$\xrightarrow{\text{mod } 23}$	16	$\xrightarrow{\text{Multiplizieren } 13}$
$13^{11}$	$= 13^{1011}$	$= 16 \cdot 13 = 208$	$\xrightarrow{\text{mod } 23}$	1	

1. Hochzahl in Binär umwandeln
2. 0 dranhängen → Quadrieren  
0 zu 1 umwandeln → Multiplizieren
3. Zwischenergebnisse modulo rechnen

Durch den Square-and-Multiply Algorithmus müsste man bei einer 2048 Bit Zahl **mindestens 2048 mal quadrieren und eventuell 2048 mal multiplizieren**.

### 11.8.5 Aufwand der Berechnung des diskreten Logarithmus

Beim diskreten Logarithmus muss man schlussendlich durch probieren. Derzeit gibt es keine effizienten Möglichkeiten oder Algorithmen zum berechnen des diskreten Logarithmus.

**Dadurch, dass das Berechnen des diskreten Logarithmus viel aufwändiger als wie das modulare Potenzieren, liegt eine kryptographische Einwegfunktion vor**

# 12 Diffie-Hellman Schlüsselaustausch

## 12.1 Ablauf

Der Diffie-Hellman Schlüsselaustausch (asymmetrisch) verwendet das **modulare Potenzieren**.

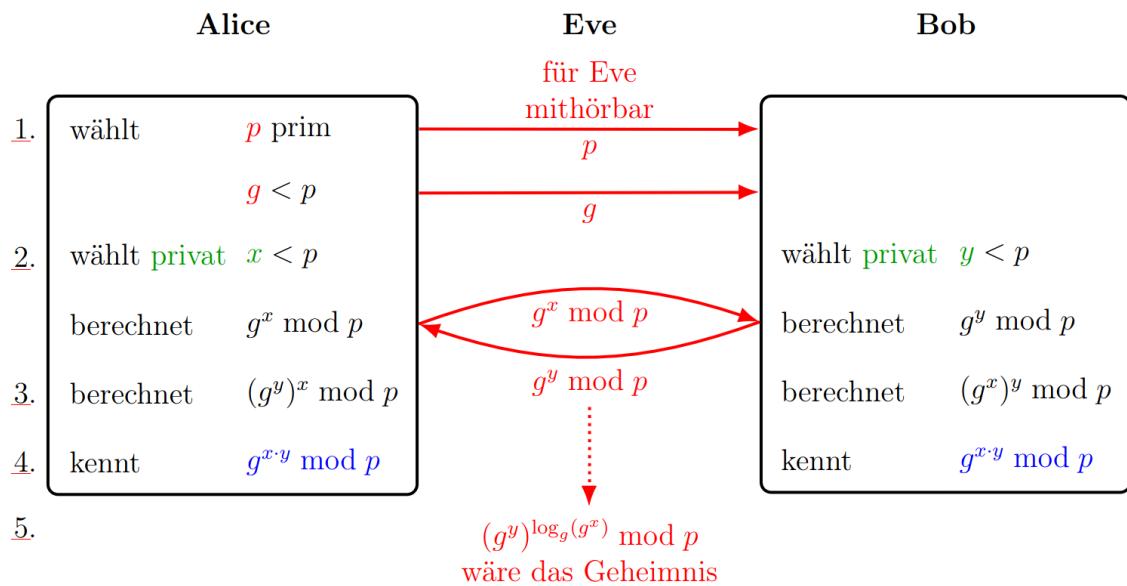


Abbildung 12.1: Diffie-Hellman Schlüsselaustausch

1. Es wird eine Primzahl  $p$  und (idealerweise) ein Generator  $g < p$  ausgewählt und veröffentlicht.
2. Beide Parteien wählen eine private Zahl  $x$  bzw.  $y$  welche  $< p$  sind und Potenzieren den Generator  $g$  hoch  $x/y$
3. Das Ergebnis wird ausgetauscht und wieder mit ihrer privaten Zahl ( $x/y$ ) potenziert.
4. da  $(g^x)^y = g^{x \cdot y} = (g^y)^x$  wurde der Schlüssel bei beiden Parteien generiert, ohne dass jemand anders es mitbekommen hat

5. Wenn Eve effizient den diskreten Logarithmus berechnen könnte, könnte sie die Verschlüsselung mit den gegebenen veröffentlichten Informationen knacken. **Da der diskrete Logarithmus enorm aufwendig ist, kann Eve die Verschlüsselung nicht knacken**

## 12.2 Allgemein

Diffie-Hellman tauscht also die Schlüssel aus, indem man sie mit gegebenen Informationen berechnet/generiert.

Diffie-Hellman gilt als sicher, wenn  $g$ ,  $x$ , und  $y$  mindestens 1024 Bit-Zahlen sind.

**Die Sicherheit von asymmetrischen Verfahren, basiert darauf, dass es (derzeit) keine effizienten Verfahren zur Umkehrung oder keine Quantencomputer gibt**

## 12.3 Man-in-the-Middle

Diffie-Hellman bietet keinen Schutz von Man-in-the-Middle Angriffe.

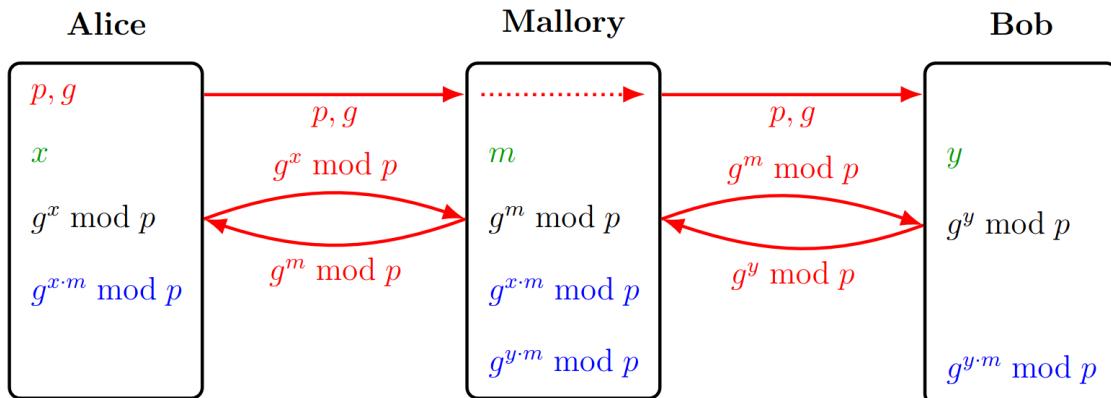


Abbildung 12.2: Diffie-Hellman Man-in-the-Middle

# 13 RSA

## 13.1 Ablauf

RSA (asymmetrisch) verwendet als Falltürfunktion die **Primzahlmultiplikation** und nutzt die Schwierigkeit der **Primfaktorzerlegung**.

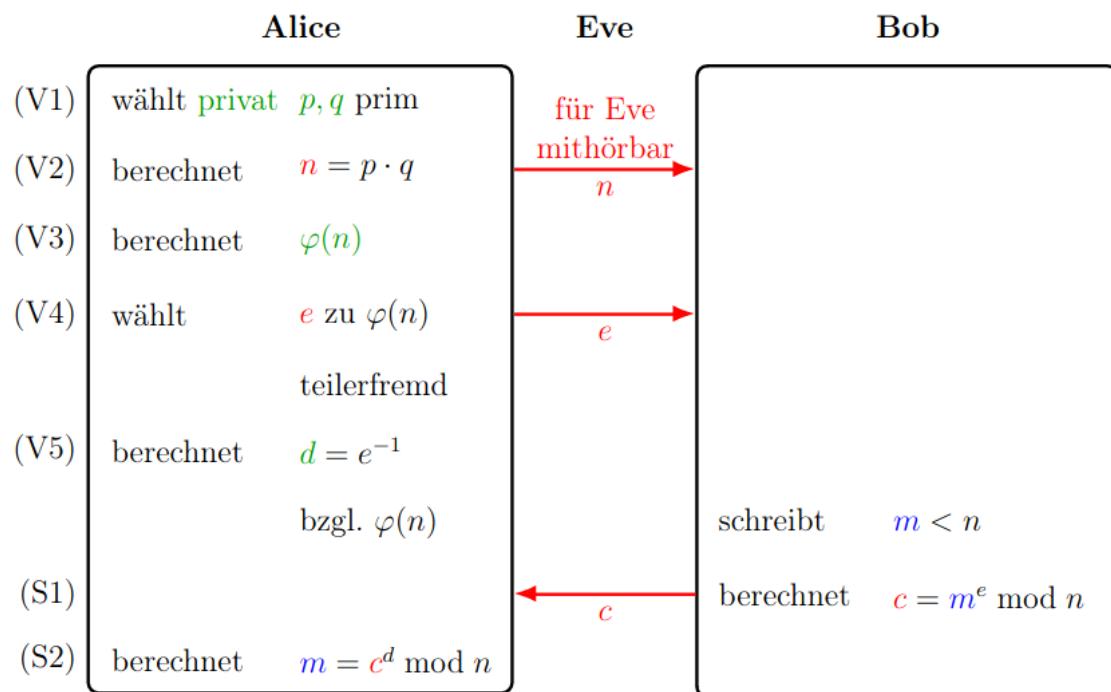


Abbildung 13.1: RSA Ablauf

- V1** Es werden zwei Primzahlen  $p$  und  $q$  ausgewählt
- V2** Modul  $n$  wird berechnet,  $n = p \cdot q$ , und veröffentlicht
- V3** Danach wird  $\varphi(n)$  berechnet, da  $p$  und  $q$  beide Primzahlen sind ist dies einfach:  
 $\varphi(n) = (p - 1) \cdot (q - 1)$
- V4** Der Verschlüsselungsexponent  $e$  wird gewählt, indem eine teilerfremde Zahl zu  $\varphi(n)$  ermittelt wird und wird veröffentlicht ( $e \dots \text{encrypt}$ ).

**V5** Da  $e$  in  $\mathbb{Z}_{\varphi(n)}$  invertierbar ist, kann man mit **erweiterten euklidischen Algorithmus** das Inverse  $d$  von  $e$  berechnen.  $d$  gilt dann als Entschlüsselungsexponent ( $d$  ... **decrypt**)

**S1** Verschlüsselung von  $m$  mittels  $c = m^e \bmod n$

**S2** Entschlüsselung von  $c$  mittels  $m = c^d \bmod n$

## 13.2 Allgemein

RSA ist natürlich anfällig für **Chosen-Plaintext Angriffe**, da  $e$  zum Verschlüsseln öffentlich ist. Lösung: großes  $n$ .

Angriff gegen RSA: **Faktorisierungsangriff**: Es wird versucht die Primfaktorzerlegung von  $n$  zu finden. Bei 512 Bit RSA ist dieser Angriff möglich → dementsprechend mindestens 1024 Bit verwenden.

Meist erfolgen aber Angriff über Seitenkanäle.

# 14 One-Time-Pad

Hier geht es um symmetrische Kryptosysteme.

## 14.1 Caesar-Verfahren

Sehr alt und bekannt. Kein Prinzip von Kerkchoff, sehr unsicher gegen Angreifer die das Verfahren kennen.

Aufgrund des kleinen Schlüsselraumes (26, effektiv 24) ist das Caesar Verfahren gegen Brute-Force schwach und aufgrund der Linearität auch gegen Wahrscheinlichkeitsanalyse schwach.

## 14.2 Monoalphabetische Substitution

Auf den ersten Blick scheint die Monoalphabetische Substitution eine Verbesserung des Caesar Verfahrens zu sein. Statt 26 Schlüssel gibt es  $26!$  ( $\approx 4 \cdot 10^{26}$ ) Schlüssel. Jedoch, wenn man einen Teil des Klartextes erfährt, erkennt man welche Buchstaben ersetzt wurden.

Die Monoalphabetische Substitution gegen Brute-Force sehr gut geschützt, jedoch umso einfacher mit einer Häufigkeitsanalyse zu knacken.

## 14.3 Vigenère Verfahren

Bei allen monoalphabetischen Verfahren ist das Problem, dass ein Klartextzeichen immer durch dasselbe Chiffrezeichen ersetzt wird. Dies ist beim Vigenère Verfahren anders → polyalphabetisch.

---

Klartext:	EINGEHEIMNIS	=	04	08	13	06	04	07	04	08	12	13	08	18	
Schlüssel:	HUNDHUNDHUND	=	07	20	13	03	07	20	13	03	07	20	13	03	$\oplus_{26}$
Chiffret.:	LCAJLBRLTHVV	=	11	02	00	09	11	01	17	11	19	07	21	21	

---

Abbildung 14.1: Vigenère Verfahren

**Schachstelle:** Mit dem Kasiski Angriff lässt sich die Schlüsselwortlänge herausfinden, und mit einer Häufigkeitsanalyse dann das Schlüsselwort herausfinden.

## 14.4 Vernam Verfahren

Weiterentwicklung des Vigenère Verfahren. Anstatt ein Schlüsselwort zu wiederholen, wird ein Text eines leicht verfügbaren Buches (Telefonbuch, Bibel,...) verwendet.

## 14.5 One-Time-Pad

Final Stufe: hier wird nun eine zufällige Kette als Schlüssel('wort') verwendet. Prinzipiell gutes Verfahren, jedoch: wirklich Zufällig Zeichenketten zu generieren ist schwer, weiters benötigt man bei einem  $n$  langen Text, einen  $n$  langen Schlüssel. **Das One-Time-Pad ist in der Theorie sehr sicher, aber schwer in Praxis umzusetzen.**

# 15 Design moderner Blockchiffren

Blockchiffre: wenn Klar- und Chiffretext aus Blöcke einer bestimmten Länge bestehen (meist 64 oder 128 Bit)

Prinzipien bei Blockchiffren

- **Nicht Empfohlen:** *Security by Intricacy*, Sicherheit durch Komplexität des Verfahrens
- **Empfohlen:** Systematische Aufbau der Chiffre + Öffentlichkeit des Verfahrens (Kerckhoff'sche Prinzip)

Bei symmetrischen Verfahren sollte es keinen besseren Angriff als wie Brute-Force geben. Ciphertext-Only aussichtslos, Chosen-/Known-Plaintext teilweise bekannt.

**Zufallsorakel:** kein erkennbarer Zusammenhang zwischen Eingabe und Ausgabe.

Claude Shannon Forderungen:

- **Konfusion:** Zusammenhang zwischen Klartext und Chiffretext nicht erkennbar machen, um die Ableitung des Schlüssels zu erschweren.
- **Diffusion:** Jedes Bit des Klartextes/Schlüssels beeinflusst möglichst viele Bits des Chiffretexts
- **Nebenforderungen:** effiziente (arbeitsspeicher-, zeit- und stromsparende) Umsetzbarkeit in Software und Hardware, daher einfache Operationen wie XOR, Bitshifts

**Bitpermutationen** sind in Hardware sehr einfach umzusetzen (Hardwarebahnen), in Software jedoch schwieriger.

Art und Reihenfolge soll nicht vom Schlüssel oder Klartext abhängen, da dies Seitenkanalangriffe ermöglicht.

Erreichen der Ziele:

- **iterierte Ausführung gleichartiger Runden:** Rundenschlüssel, abgeleitet vom Hauptschlüssel
- **Permutationen für Diffusion**
- **Substitutionen (S-Boxen) für Konfusion**

## 15.1 Data Encryption Standard (DES)

Jahr 1977 unter Beteiligung NSA. Anwendungen: Bankkarten

Einfache Operationen, besonders gute Hardwareimplementierung, Software langsamer.

Klartext- und Chiffretext sind 64 Bit Blöcke, enthält jedoch 8 Prüfbits → **56 Bit Schlüssel**. ( $2^{56} = 7 \cdot 10^{16}$ )

**Feistel-Chiffre mit 16 Runden:** Ver- und Entschlüsseln ist gleich.

**konstante S-Boxen** und **konstante Permutationen**. S-Boxen wurden auf Sicherheit untersucht und haben die Eigenschaft, dass bei Veränderung eines Eingangsbites mindestens 2 Ausgangsbits verändert werden.

Schlüssel mit XOR, da schnell.

### 15.1.1 Schlüsselraum

Da die effektive Schlüssellänge nur 56 Bit ist, war damals ein Brute-Force Angriff (z.B. durch NSA) erreichbar gewesen. RSA Brute-Force: alle Schlüssel durchprobieren bis sinnvoller Klartext herauskommt, man muss natürlich wissen, was Verschlüsselt wurde (Format, Inhalt Sprache, Codierung,...). Daher ist normaler DES **nicht mehr sicher**.

### 15.1.2 Blockgröße

Nachrichten werden in 64 Bit Blöcken unterteilt, ähnlich zu einer Monoalphabetischen Substitution mit 64 Bit Blöcke → Häufigkeitsanalyse?

- Bei gleichgekommen Klartexten, wiederholt sich ein Block nach  $2^{64}$  64 Bit Blöcke → 140 Mio TB, also Nein
- Jedoch ist ein 64 Bit Block 8 ASCII Zeichen. Aber trotzdem nein

**Schlüssellänge** beeinflusst Aufwand von Brute Force.

**Blockgröße** beeinflusst Aufwand von Häufigkeitsanalyse.

DES ist prinzipiell sicher, aber wegen 56 Bit Schlüssel einfach zu knacken.

## 15.2 Triple-DES, 3DES, DESede

Das DES prinzipiell, vom Design keine Schwächen bekannt sind, verwendet man ihn 3x hintereinander um das Problem der Schlüssellänge zu beheben.

Warum nicht 2x? Wegen **Meet-in-the-Middle Angriff**

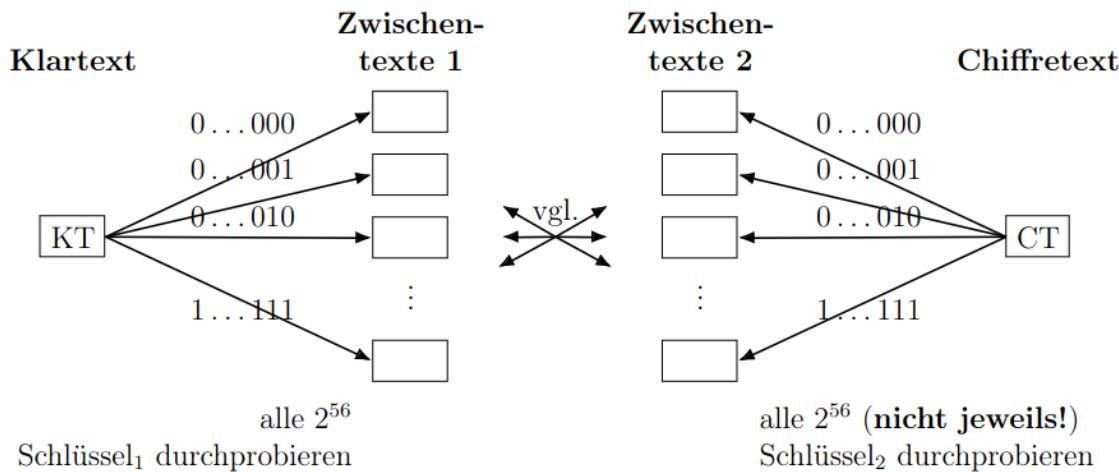


Abbildung 15.1: 3DES Meet-in-the-Middle Angriff

Ausgang: Known-Plaintext Angriff, Klartext-Chiffretext-Paar. Es wird der Klartext mit allen  $2^{56}$  Schlüsseln verschlüsselt und mit allen  $2^{56}$  der Chiffretext entschlüsselt und miteinander verglichen. Bei einer Übereinstimmung ist der Schlüssel bekannt. Man muss nur  $2^{56} + 2^{56} = 2^{57}$  Ver- und Entschlüsselungen durchführen → effektiv 57 Bit Schlüssellänge.

Deshalb wendet man DES 3-mal an:  $DES_{K1}(DES_{K2}(DES_{K1}(m)))$  Dadurch muss man  $2^{112} + 2^{56} = 2^{112}$  Ver- und Entschlüsselungen durchführen. Meet-in-the-Middle Angriff ist immer noch möglich aber enorm aufwendiger.

Da jedoch DES in Software langsam ist, wegen Speicherbedarf, Strombedarfs,... erhöht man nicht die Rundenzahl, sondern verwendet andere Verschlüsselungen wie Blowfish, AES,...

## 15.3 Advanced Encryption Standard (AES)

1997 kam es zu einer Ausschreibung für einen DES-Nachfolger. AES ist gut Ressourcenverbrauch und Leistung in Hard- und Software. AES ist kein Feistel-Chiffre sondern ein **SP-Netzwerk (Substitution-Permutation)**, deshalb verläuft die Entschlüsselung komplett anders (Erfinder müssen beachten dass die Entschlüsselung auch Effizient ist)

Schlüssellänge: 128, 192 oder 256 Bit

Abhängig davon ob 10, 12 oder 14 Runden 128-Bit Blöcker, mit  $4 \times 4$  Byte Matrizen. Eine Runde:

1. **SubBytes:** S-Box Substitutionen (Resistent gegenüber linearer und differentieller Kryptoanalyse)

2. **ShiftRows:** Einträge je Zeile werden 0, 1, 2, 3 Einträge nach links rotiert. (Zahlen aus selben Spalten in verschiedenen Spalten)
3. **MixColumn:** Zahlen in einer Spalte werden vermischt (lineare Operationen)
4. **AddRoundKey:** Rundenschlüssel mit XOR verknüpft

AES wird verwendet in:

- WPA2 (WLAN)
- KeePass
- LTE

# Abbildungsverzeichnis

1.1	Bustopologiearten . . . . .	2
1.2	CAN-Bus Aufbau . . . . .	7
1.3	I2C-Bus Bitsetzung . . . . .	9
1.4	I2C-Bus Steuersignale . . . . .	9
1.5	I2C Ablauf . . . . .	10
3.1	OSI-Modell Datenübertragung . . . . .	17
3.2	Glasfaserkabelarten . . . . .	19
3.3	Baum- und Stern topologie . . . . .	20
3.4	Ethernet Frame . . . . .	21
3.5	Layer 2 & 3 Adressierung . . . . .	22
3.6	ARP-Spoofing . . . . .	23
3.7	CIDR Beispiel . . . . .	26
3.8	VLSM Beispiel . . . . .	27
3.9	L4-Adressierung . . . . .	29
3.10	TCP 3-Way-Handshake . . . . .	30
3.11	TCP 2-Way-Handshake . . . . .	30
3.12	Layer 4 Segmentierung . . . . .	31
3.13	Request/Response Modell . . . . .	32
3.14	DNS-Hierarchie . . . . .	33
3.15	DHCP-Handshake . . . . .	33
3.16	Request/Response Modell . . . . .	34
3.17	Position der Wildcardmask . . . . .	37
3.18	Static NAT, 1:1 Mapping . . . . .	37
3.19	NAT mit PAT, n:1 Mapping . . . . .	38
5.1	IPv4-IPv6 Translation mit NAT64 . . . . .	44
5.2	IPv4-IPv6 Tunneling . . . . .	44
5.3	SLAAC . . . . .	45
6.1	Elektromagnetisches Spektrum . . . . .	49
6.2	WPA2 Personal Handshake . . . . .	52
7.1	IPS & IDS . . . . .	55



7.2 Honeypot . . . . .	55
7.3 VPN . . . . .	56
7.4 ESA & WSA . . . . .	57
8.1 Einwegfunktion . . . . .	58
8.2 PAP . . . . .	60
8.3 PAP . . . . .	61
8.4 Public Key Infrastruktur . . . . .	62
8.5 HTTPS Verbindungsaufbau . . . . .	63
9.1 John McCumber Cybersecurity Cube . . . . .	65
10.1 Grundlegende Situation der verschlüsselten Kommunikation . . . . .	67
10.2 Symmetrische Kryptosysteme . . . . .	70
10.3 Asymmetrische Kryptosysteme . . . . .	71
11.1 Definitionsmenge und Wertemenge einer normalen Funktion . . . . .	83
11.2 Definitionsmenge und Wertemenge einer umkehrbaren Funktion . . . . .	84
11.3 Definitionsmenge und Wertemenge einer Einwegfunktion . . . . .	84
11.4 Definitionsmenge und Wertemenge einer Falltürfunktion . . . . .	85
12.1 Diffie-Hellman Schlüsselaustausch . . . . .	87
12.2 Diffie-Hellman Man-in-the-Middle . . . . .	88
13.1 RSA Ablauf . . . . .	89
14.1 Vigenère Verfahren . . . . .	92
15.1 3DES Meet-in-the-Middle Angriff . . . . .	95

# I Quellcodeverzeichnis

1.1	UART Sender . . . . .	4
1.2	UART Receiver . . . . .	5
1.3	I2C #1 Master (Writer) . . . . .	10
1.4	I2C #1 Slave (Reciever) . . . . .	11
1.5	I2C #2 Master (Reader) . . . . .	11
1.6	I2C #2 Slave (Sender) . . . . .	11