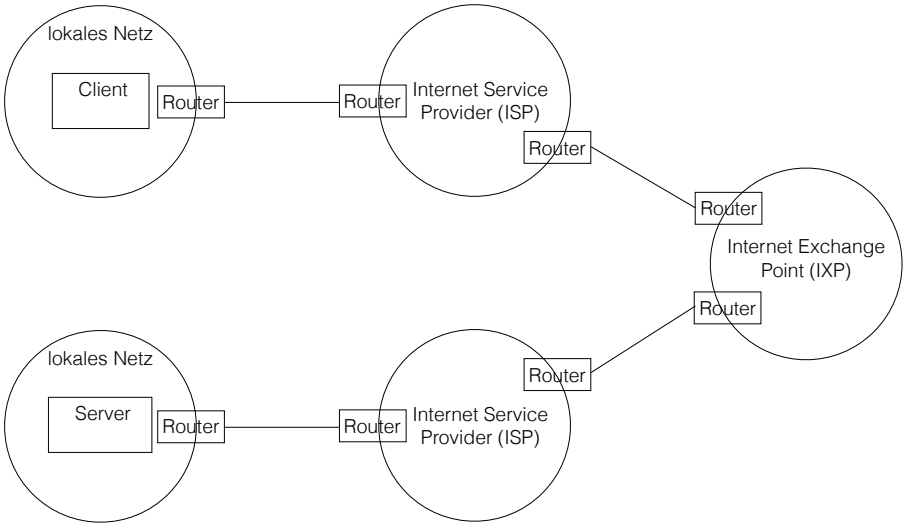
# 19 Internetsicherheit




Für unsere Untersuchung betrachten wir 3 Angreifertypen:

- Typ 1 hat Zugang zu einem Zwischenknoten und versucht
  - passive Angriffe durchzuführen (ausspähen)
    Verletzung des Schutzziels Vertraulichkeit
  - aktive Angriffe durchzuführen (manipulieren)
    Verletzung der Schutzziele Integrität und Verfügbarkeit
- Typ 2 versucht einen Endknoten (Client) anzugreifen
  - Eindringen in das lokale Netz
    Verletzung der Ziele Vertraulichkeit, Integrität, Verfügbarkeit
  - Störung der Funktionsfähigkeit (Denial of Service)
    Verletzung des Schutzziels Verfügbarkeit
- Typ 3 ist ein bösartiger Endknoten (Server), der versucht
  - die Identität eines vertrauenswürdigen Servers anzunehmen

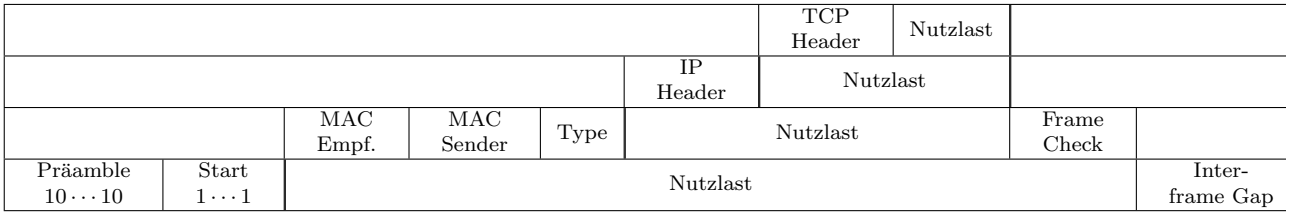
## 19.1 Angreifertyp 1: Kommunikationssicherheit

- Sicherheit spielte in der Anfangszeit des Internet keine Rolle
- Keine Mechanismen für Vertraulichkeit und Authentizität vorgesehen

TCP/IP-Referenzmodell

| OSI-Layer | TCP/IP-Layer | Beispiele |
|---|---|---|
| 5-7 | Application Layer | http, ftp, smtp, imap |
| 4 | Transport Layer | TCP, UDP |
| 3 | Internet Layer | IPv4, IPv6 |
| 1-2 | Link Layer | Ethernet, FDDI |

Typischer Aufbau:




- TCP-Header: Port Empfänger, Port Sender, Paketnummer
- IP-Header: IP-Adresse Empfänger, IP-Adresse Sender

Schutzmaßnahmen

- Auf Application Layer (anwendungsspezifisch):
  - S/MIME (Secure / Multipurpose Internet Mail Extensions)
  - pgp (Pretty Good Privacy)
  - ssh (Secure Shell)
- Auf Transport Layer (transportprotokollspezifisch)
  - tls (transport layer security)
  - Für alle Anwendungen, die TCP nutzen (http, ftp, smtp)

- Internet Layer (transportprotokollunabhängig)
  - IPSec (IP Security)

**Transport Layer Security**

TLS bietet

- Symmetrische Verschlüsselung der Nutzlast (AES, Triple DES)
- Datenauthentisierung der Nutzlast (HMAC)

<table>
<tr><td colspan="4">Application Layer (https, IMAPS, SMTPS, SFTP)</td></tr>
<tr><td>Handshake Protocol</td><td>Change Cipher Spec. Protocol</td><td>Alert Protocol</td><td>Application Data Protocol</td></tr>
<tr><td colspan="4">Record Protocol (Verschlüsselung und Datenauthentisierung)</td></tr>
<tr><td colspan="4">Transport Layer (TCP)</td></tr>
<tr><td colspan="4">...</td></tr>
</table>

Handshake-Protokoll

1. Client: Random Number $r_1$
2. Client → Server: client_hallo
   tls-version, time, $r_1$, session-id, cipher-suite (Möglichkeiten)
   cipher-suite: Algorithmen für Instanzauthentisierung, Schlüsseleinigung
   Verschlüsselung, Datenauthentisierung
   - Bsp. 1: TLS_RSA_with_AES_128_CBC_Sha256
   - Bsp. 2: TLS_DHE_RSA_with_AES_128_CBC_Sha256
3. Server: Random Number $r_2$
4. Server → Client: server_hallo
   tls-version, time, random $r_2$, session-id, cipher-suite (ausgewählt)
5. Server → Client: Server Certificate $C_S$ (für public key $pk_S$)
6. Server → Client: Demand Client Certificate (optional)

7. Client: Verify Server Certificate
   Rootzertifikate zum Validieren werden mit den Browsern ausgeliefert
8. Client → Server: Client Certificate $C_C$ (für public key $pk_C$) (optional)
9. Server: Verify Client Certificate (optional)
10. Client: Signatur über alle bisherigen Daten (mit $sk_S$) (optional)

11.a. Fall DHE: Client ↔ Server:

- Austausch eines Geheinmisses $g$ über authentisiertes Diffie-Hellman Signatur der Schlüsselanteile über $sk_S$ und $sk_C$ (optional)
- Ableitung eines Pre Master Key (PMK) aus $g, r_1, r_2$

11.b. Fall RSA: Client → Server:

- Client generiert PMK aus Zufall, $r_1, r_2$
- Client schickt PMK verschlüsselt (mit $pk_S$) an Server

12. Beide: Ableitung von Verschlüsselungs-, Authentisierungsschlüssel, IV
13. Client ↔ Server (Abschluss), Beide schicken
    - Change Cipher Spec (ab jetzt wird verschlüsselt und authentisiert)
    - Hashwert aller bisherigen Daten (zur Kontrolle)

**Sicherheitsbewertung:**

- Keine Replay-Attacken möglich (wegen $r_1, r_2$, gehen in Schlüssel ein)
- Authentisierung Server:
  - Fall RSA: Nur Server kann PMK entschlüsseln
  - Fall DHE: Schlüsselanteil Server ist authentisiert
- Authentisierung Client: (optional)
  - Fall RSA: Client signiert u.a. $r_2$ (challenge-response)
  - Fall DHE: Schlüsselanteil Client ist authentisiert
- Kein Schutz der Metadaten (wer komm. wann mit wem)