Modul 10:

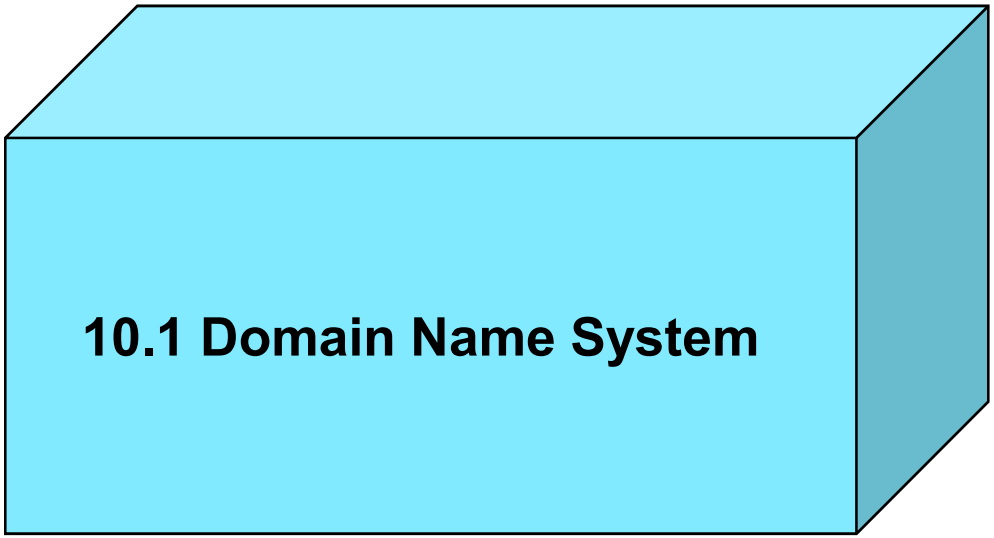
10.1 Domain Name System (DNS)

10.2 Hypertext Transfer Protocol (HTTP)

10.3 Sicherer Internet-Zugriff, Grundideen von HTTPS/ SSH

10.4 Anhang: Einrichten eines DNS Servers

© M. Leischner



© M. Leischner

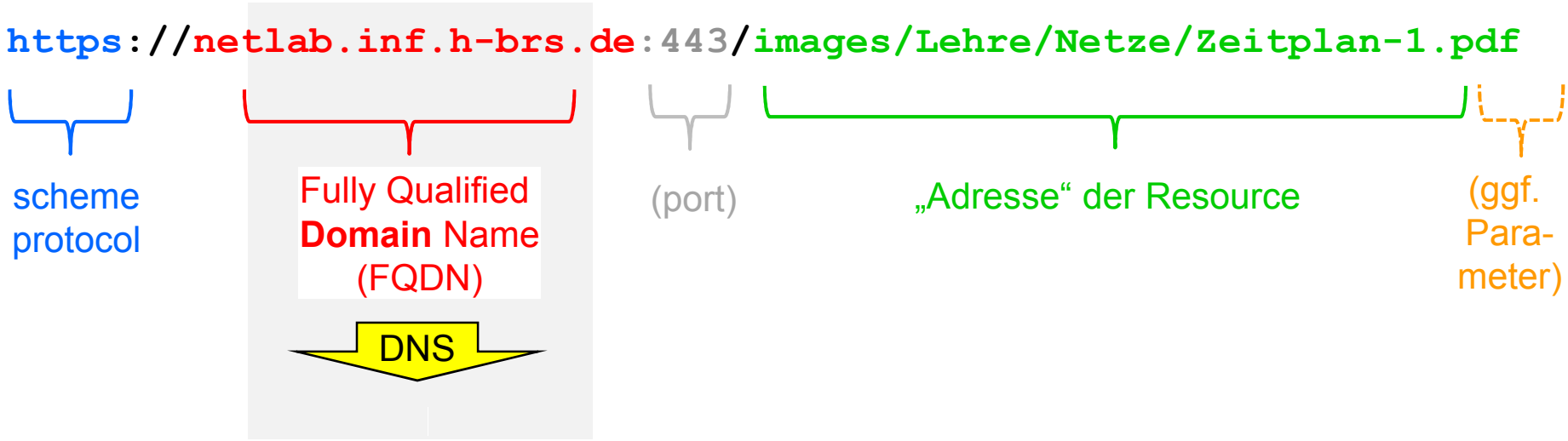
Überblick: Domain Name System

- **Aufgabe:**
  Namensauflösung Domain → IP
- **Bedeutung:**
  Extrem wichtiger Netzdienst, ohne diesen läuft praktisch nichts mehr
- **Technische Realisierung:**
  Weltweit, verteilte Anwendung
- **Organisation:**
  - hierarchisch strukturierter Namensraum
  - dezentrale Verwaltung
- **Beschrieben in**
  - RFC 1034: Domain names - concepts and facilities, November 1987,
  - seither nur "Updates"
- **Sicherheit:**
  - Vielfältige Angriffe („Falsche Auskunft“, Denial of Service, …)
  - Abwehrmaßnahmen (z.B. DNSSEC (DNS Security))

© M. Leischner

Namensraum: URI - URL - Domain

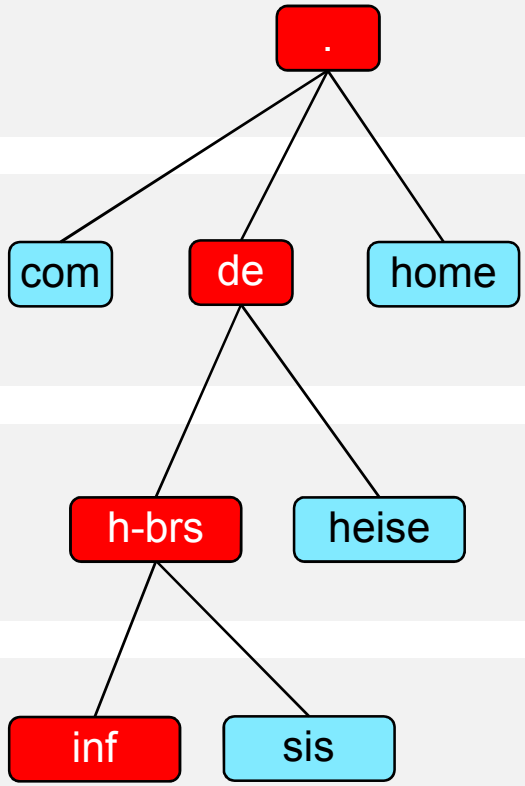
- **Jede Ressource im Internet wird kann durch einen vom Menschen lesbaren Identifikator identifiziert werden**
- **URI = Uniform Resource Identifier (siehe auch RFC 3986)**
- **URL = Uniform Resource Locator (häufig gleich verwendet, laut RFC 3986 URL = URI in einem Computernetzwerk)**
- **Grundprinzip, eingeführt 1994 durch Timothy John Berners-Lee (Erfinder des WWW)**



© M. Leischner

Struktur der Domain Names

DNS Root



Die Rootdomain ist leer

TLD, top level domain

- Generische Domänennamen (gTLD)
- CountryCode-Domänennamen (ccTLD)
- Private TLDs, z.B. home/corp/mail (seit 2018)

Internet Corporation for Assigned Names and Numbers (ICANN) koordiniert TLDs und DNS

SLD, second level domain

- Werden in der Regel von einem beim zuständigen NIC (denic.de) akkreditierten privatwirtschaftlichen Registrar (Internet Service Provider) verkauft und betreut

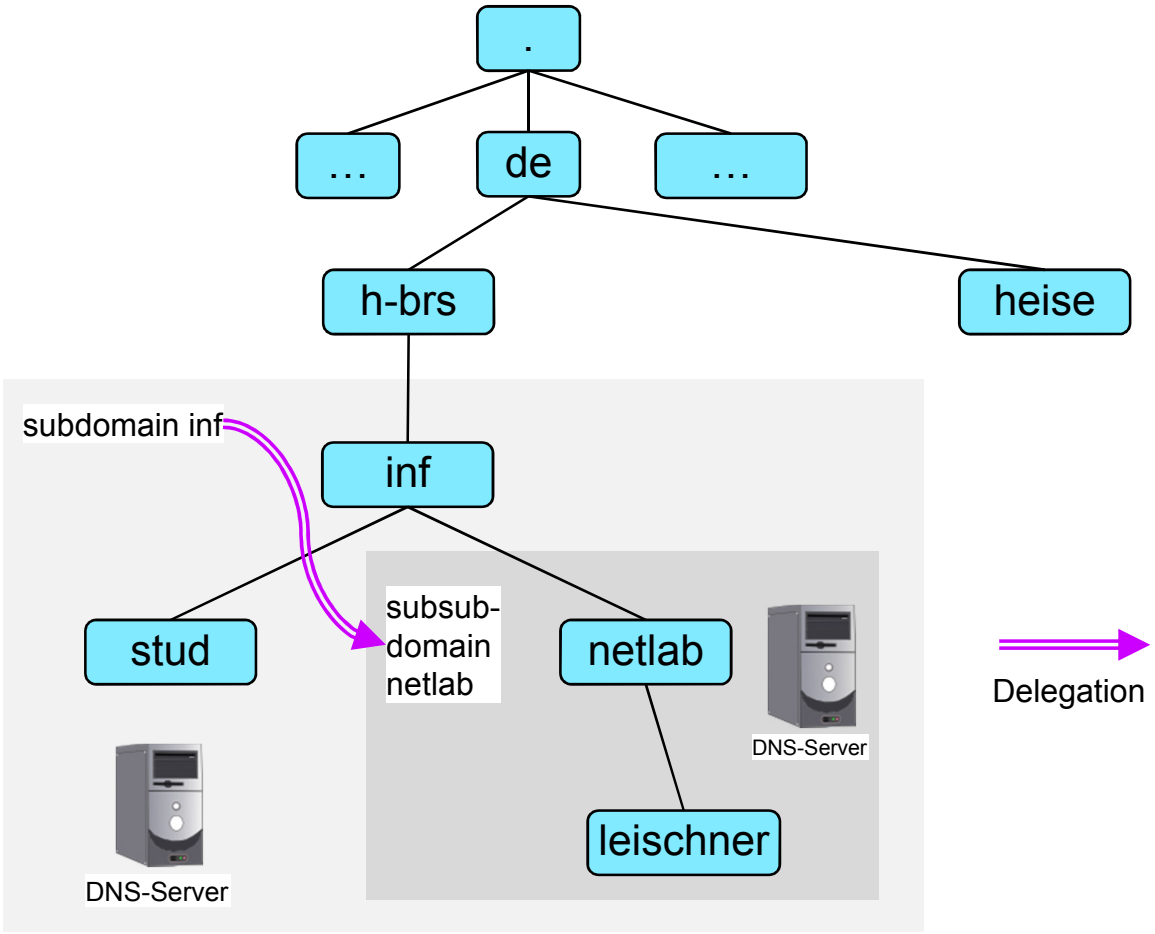
subdomain

- Werden durch den Kunden verwaltet

z.B. Fully Qualified Domain Name: netlab.inf.h-brs.de.

© M. Leischner

Domäne, Delegation, Zone und Zone-File



- Die technische und organisatorische Verantwortlichkeit („Authority“) für eine Domäne (Subdomäne, …) kann an einen anderen Administrator übergeben werden („Delegation“).
- Ein Administrator nutzt einen DNS-Server, um den Bereich, für den er verantwortlich ist zu verwalten („Zone“).
- Ein DNS-Server wird durch einen Zone-File konfiguriert, der alle notwendigen Konfigurations-informationen enthält. Findet eine Delegation statt, so muss auch diese im Zone-File beschrieben werden.

Eine Domäne kann in mehrere Zonen aufgeteilt werden. Jede Zone wird durch Name-Server (NS) verwaltet. Das Zone-File enthält „Ressource-Records“, z.B. Namen und die IPv4- bzw. IPv6 Adresse oder einen NS.

© M. Leischner

Resource Records, Aufbau und Beispiele, Forward- u. Reverse-Lookup

Format: <name> [<ttl>] [<class>] <type> [<rdlength>] <rdata>

- **<name>** Domänenname des Objekts, zu dem der Resource Record gehört
- **<ttl>** time to live (in Sekunden), Gültigkeitsdauer des Resource Records (optional)
- **<class>** Protokollgruppe des Resource Records
- **<type>** Typ des Resource Records
- **<rdlength>** Länge der Daten (optional)
- **<rdata>** resource data d.h. Daten, die den Resource Record näher beschreiben z.B. eine IP-Adresse

```
www.heise.de.           79251   IN      A       193.99.144.85
```

```
www.heise.de.           74961   IN      AAAA    2a02:2e0:3fe:1001:7777:772e:2:85
```

Für jede Domäne gibt es zwei Zonen: Forward-Lookup, Reverse-Lookup .
Eine Zonendefinition verweist auf einen Zone-File, z.B.

`fwd.netlab.inf.h-brs.de.` **oder** `rev.netlab.inf.h-brs.de.`

der die Ressourcen-Records der entsprechenden Zone enthält. Hier werden neue DNS-Einträge hinzugefügt.

Basics zu DNS-Abfrage

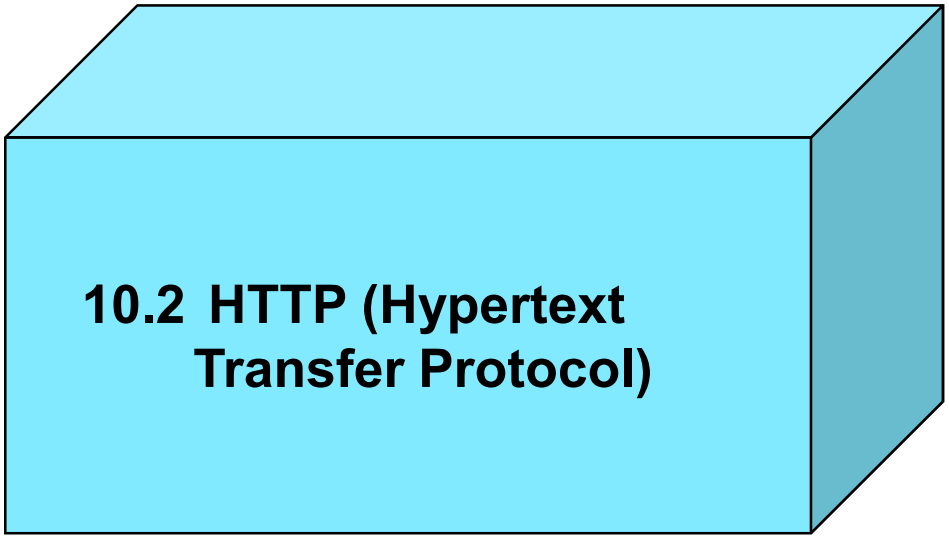
- **Einfaches Request/Response-Protokoll** (i.d.R vom „Resolver“ an Name-Server)
- **DNS-Anfrage über UDP Port 53. Maximal 512 Byte. Beispiele von Flags:**
  - **AA: authoritative answer**
  - **RD: recursion support desired**
  - **RA: recursion support available**
- **Rekursive Antwort auf eine Anfrage** (Name-Server liefert die Antwort, fragt u.U. weitere DNS-Server)**:**
  - **Falls Antwort im Cache, wird diese ausgeliefert**
  - **Andernfalls Anfrage an einen der 13 Root-Nameserver z.B. bezüglich „netlab.inf.h-brs.de“**
  - **Root-Server sendet Info (Name, IP-Adresse) zu de-Nameserver**
  - **Anfrage an de-Nameserver bezüglich netlab.inf.h-brs.de**
  - **de-Nameserver sendet Info (Name, IP-Adresse) bezüglich h-brs.de-Nameserver**
  - **…**
- **Iterative Antwort auf eine Anfrage:**
  - **Name-Server sendet Antwort, falls direkt möglich, sonst nur Info über weiteren Name-Server**

© M. Leischner

DNS Servertypen

- **DNS-Resolver**:
  Software zur Namensauflösung. Können einen Cache enthalten. Betriebssysteme (W10, Ubuntu besitzen einen DNS-Resolver)
- **Rekursiver DNS-Server**:
  Löst DNS-Anfrage bei Bedarf rekursiv auf. Betreibt meist zusätzlich einen Cache
- **Forwarding DNS-Server**:
  Ein DNS-Server, der alle Anfragen (an andere DNS-Server) weiterleitet. Beispiele: Fritzbox, Pi-Hole
- **Autoritativer DNS-Server**:
  Verantwortlich für eine Zone.
- **Primary / Slave DNS-Server**:
  In einer Domäne werden zwei DNS-Server betrieben. Falls der primäre DNS-Server ausfällt, übernimmt der Slave
- **Public / Private DNS-Server**
- **Caching DNS-Server:**
  Holt Informationen von anderen DNS-Servern und speichert diese in eine Cache zwischen

© M. Leischner



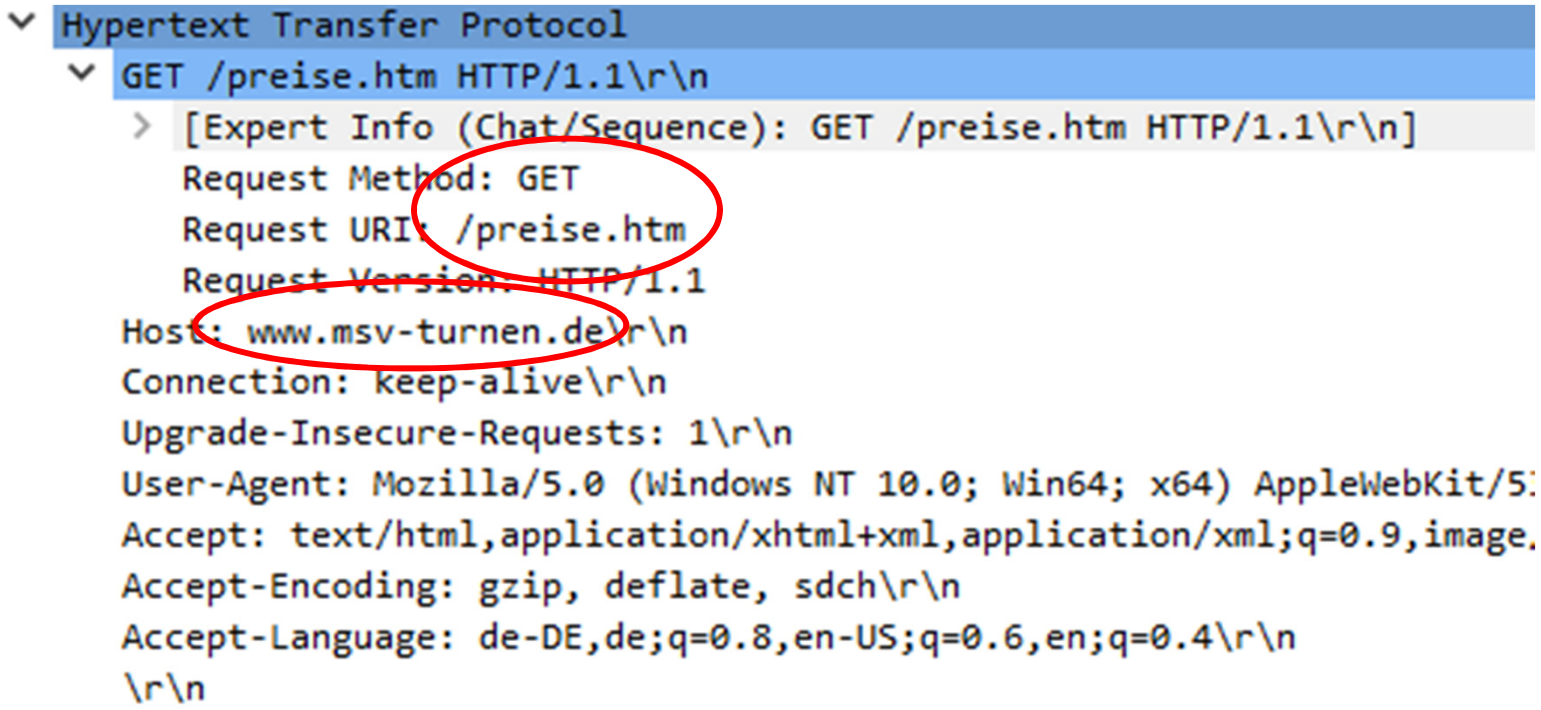
© M. Leischner

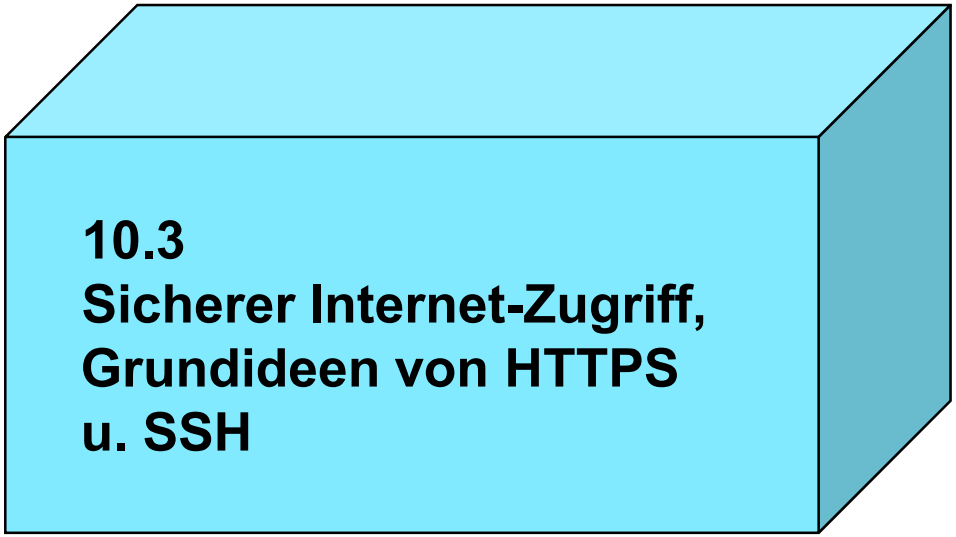
HTTP-Grundkonzept

- **HTTP setzt auf das Protokoll TCP auf.**
- **Der Web-Server benutzt standardmäßig den TCP-Port 80.**
- **HTTP ist ein zustandsloses Request/Response-Protokoll.**
- **HTTP ist textorientiert.**
- **Typische Request-Operationen sind**
  - **Get: Abruf des in der URL bezeichneten Dokuments**
  - **Put: Hochladen von Information auf den Server**
  - **Head: nur Header der Get Response soll an Client zurück gesendet werden**
  - **…**
- **Resonse enthält neben angeforderten Informationen im Header auch Statuscode** (z.B. 200 OK)
- **Designziele von HTTP sind:**
  - **Einfachheit: Das Protokoll soll nur wenige Ressourcen beanspruchen. (Davon entfernen wir uns gegenwärtig wieder.)**
  - **Geschwindigkeit: Das Protokoll soll so schnell wie möglich sein.**

© M. Leischner

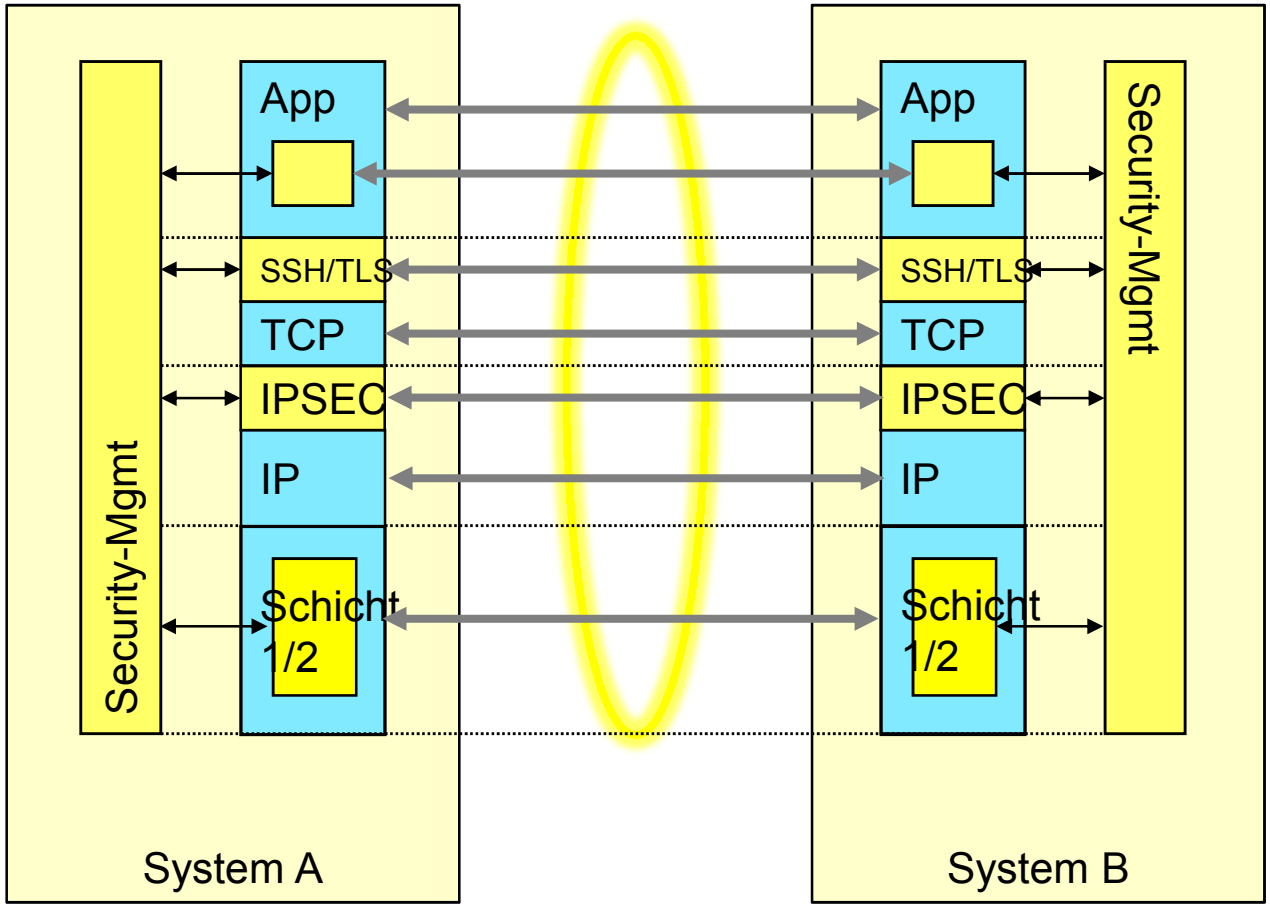
Beispiel: Get auf die Seite : http://www.msv-turnen.de/preise.htm





© M. Leischner

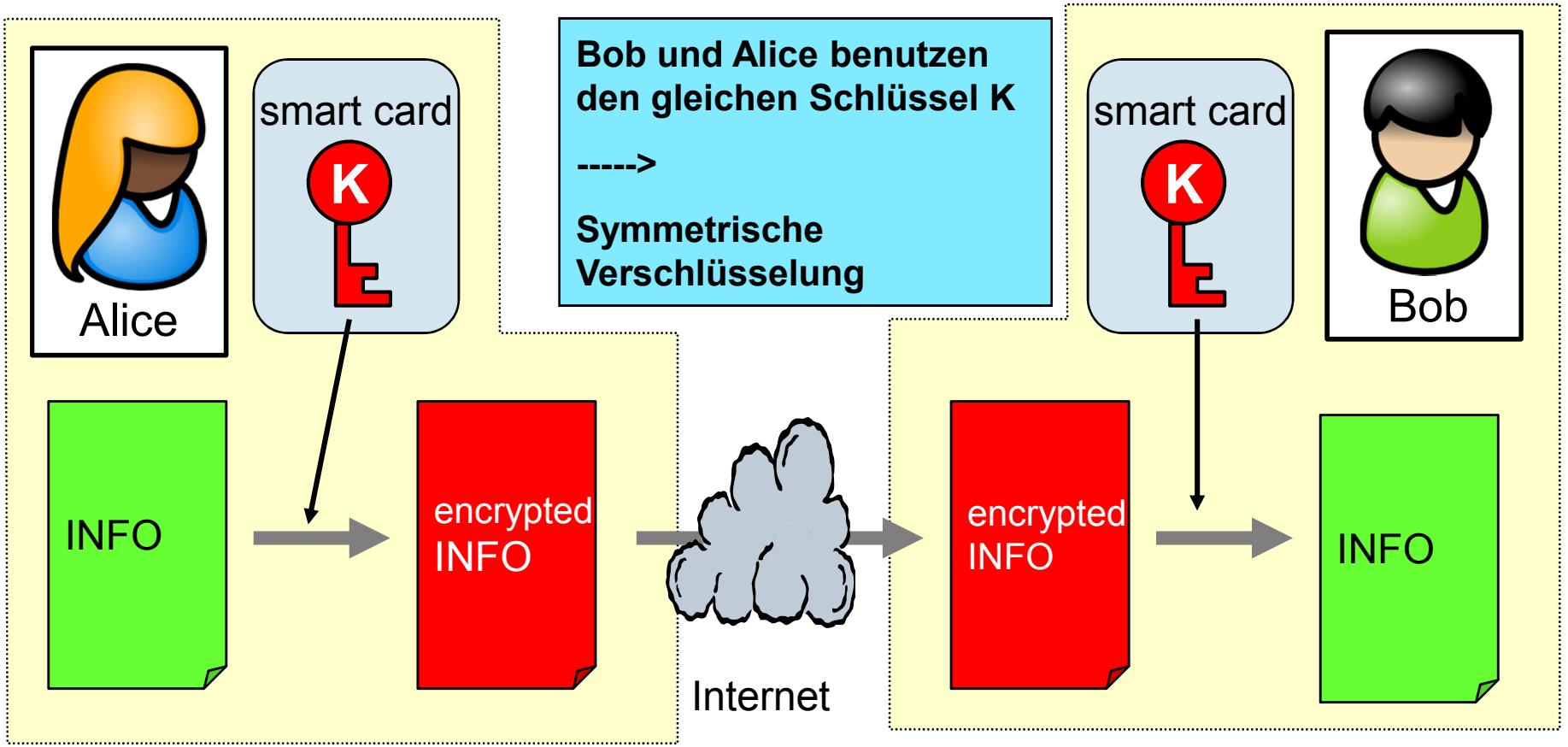
Sicherheit im Internet-Protokollstack



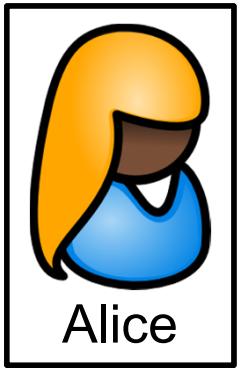
Ebenen der Sicherheit

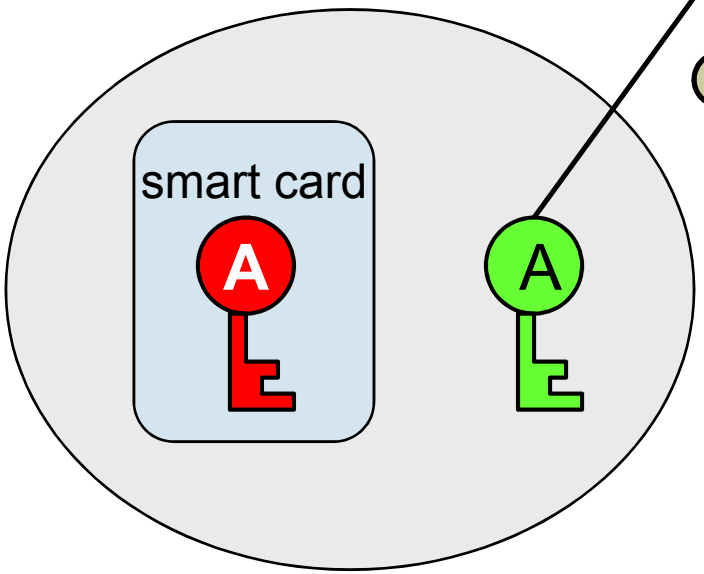
- **Sicherheit auf Applikationsebene, z.B. eCash, PGP. ("ich vertraue der Applikation")**
- **TLS: sichere Ende-zu-Ende-Verbindung ("ich vertraue dem sicheren Transport bis hin zu meiner Anwendung")**
- **SSH: Sicheres Einloggen in ein Remote-System**
- **IPsec sichere IP-Verbindung zwischen Systemen ("ich vertraue der Übermittlung über das Internet")**

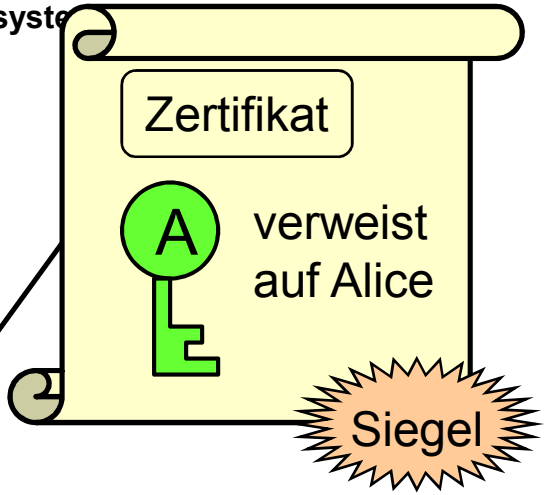
Krypto-Basic 1: Symmetrische Verschlüsselung



Krypto-Basic 2: Prinzip der asymmetrischen bzw. Public-Key-Kryptographie

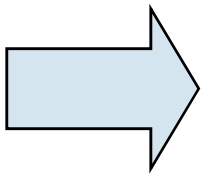






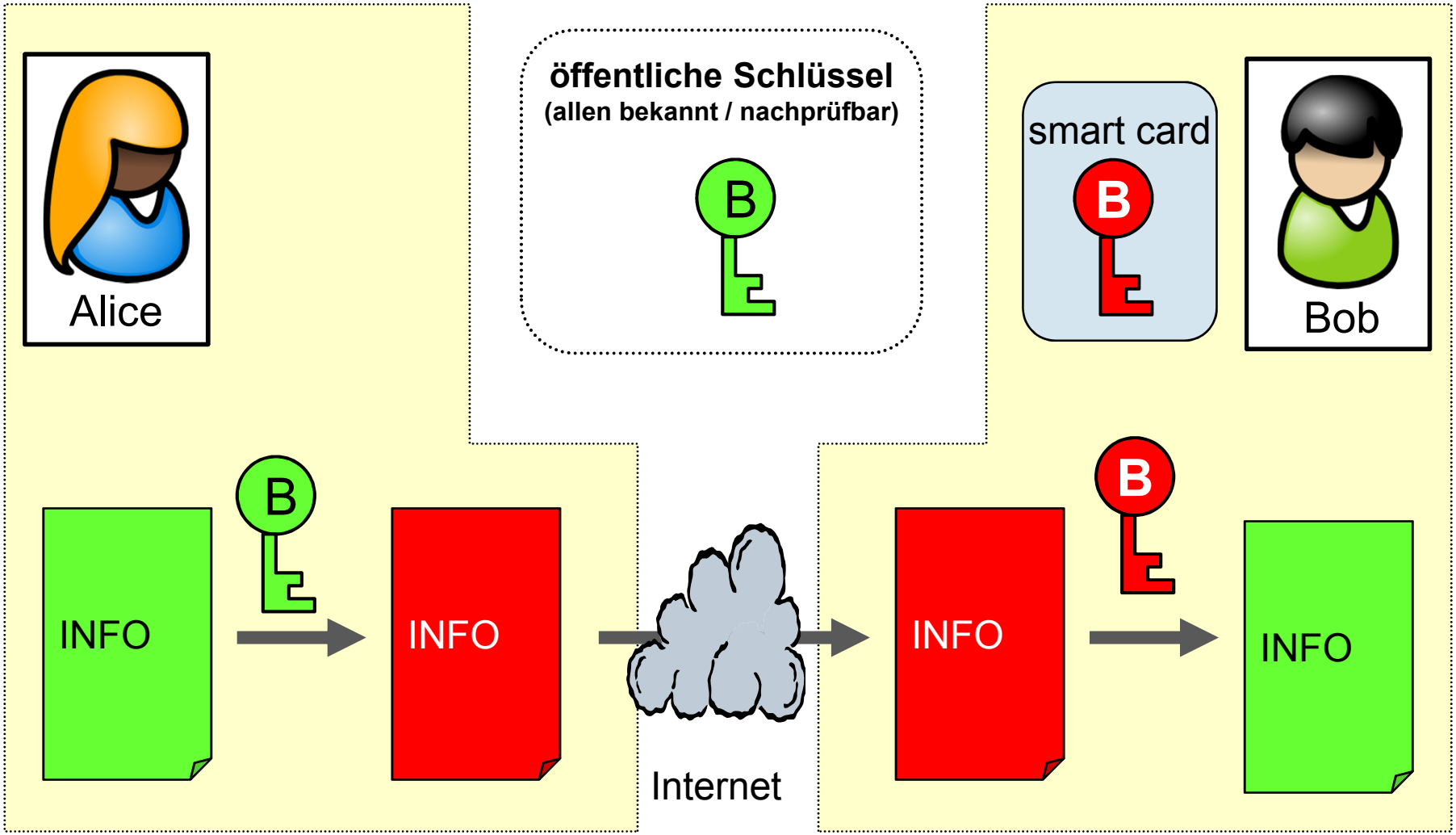
Alice hat zwei Schlüssel

- **einen secret key**
  (z.B. auf einer Chipkarte)
- **einen public key**
  (Dieser ist nicht geheim, kann **und sollte** in einem öffentlichen Verzeichnis gespeichert werden

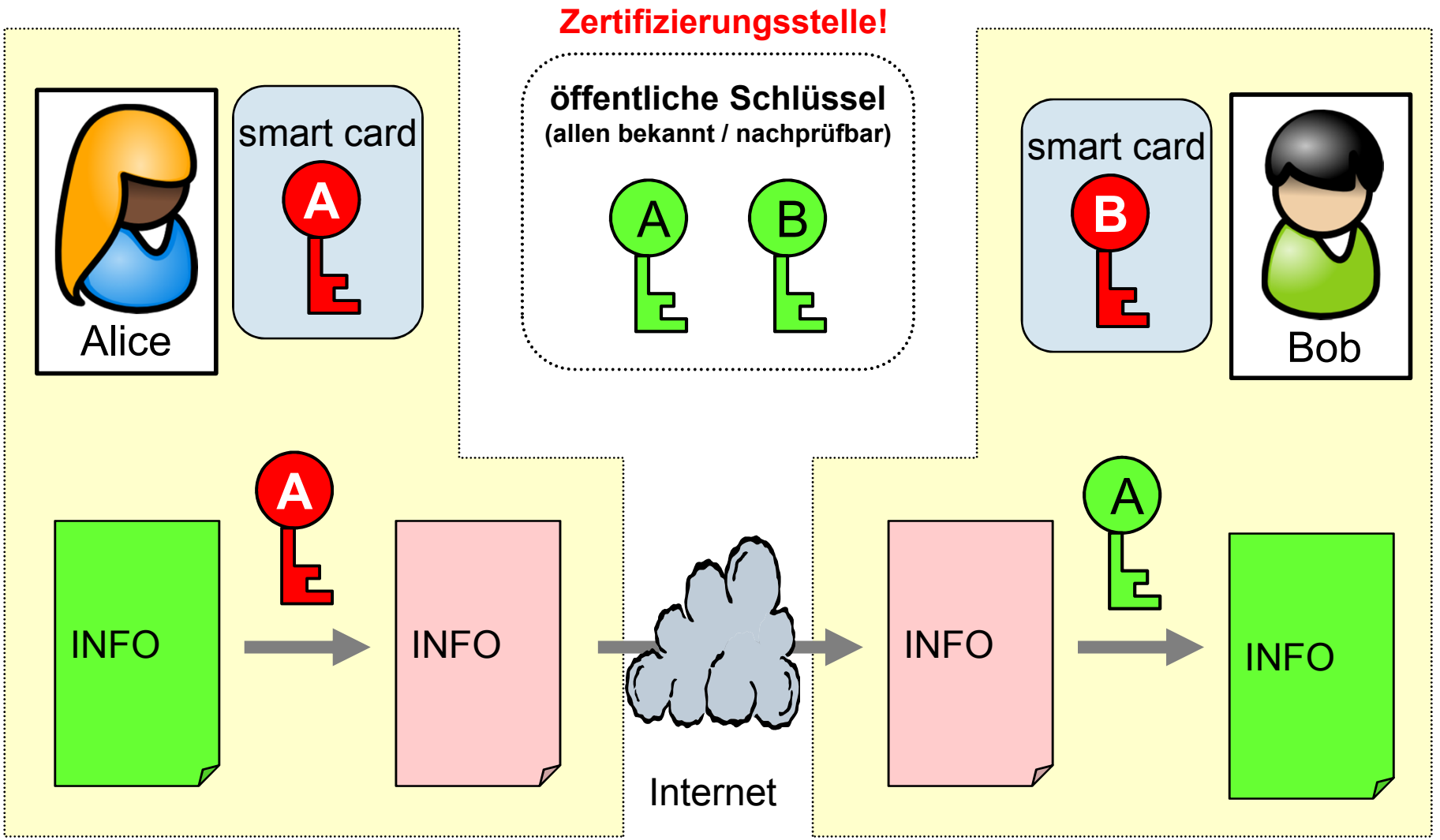


Um zu verschlüsseln und um zu entschlüsseln benötigt man verschiedene Schlüssel (= asymmetrisch)

Krypto-Basic 3: Asymmetrische Verschlüsselung



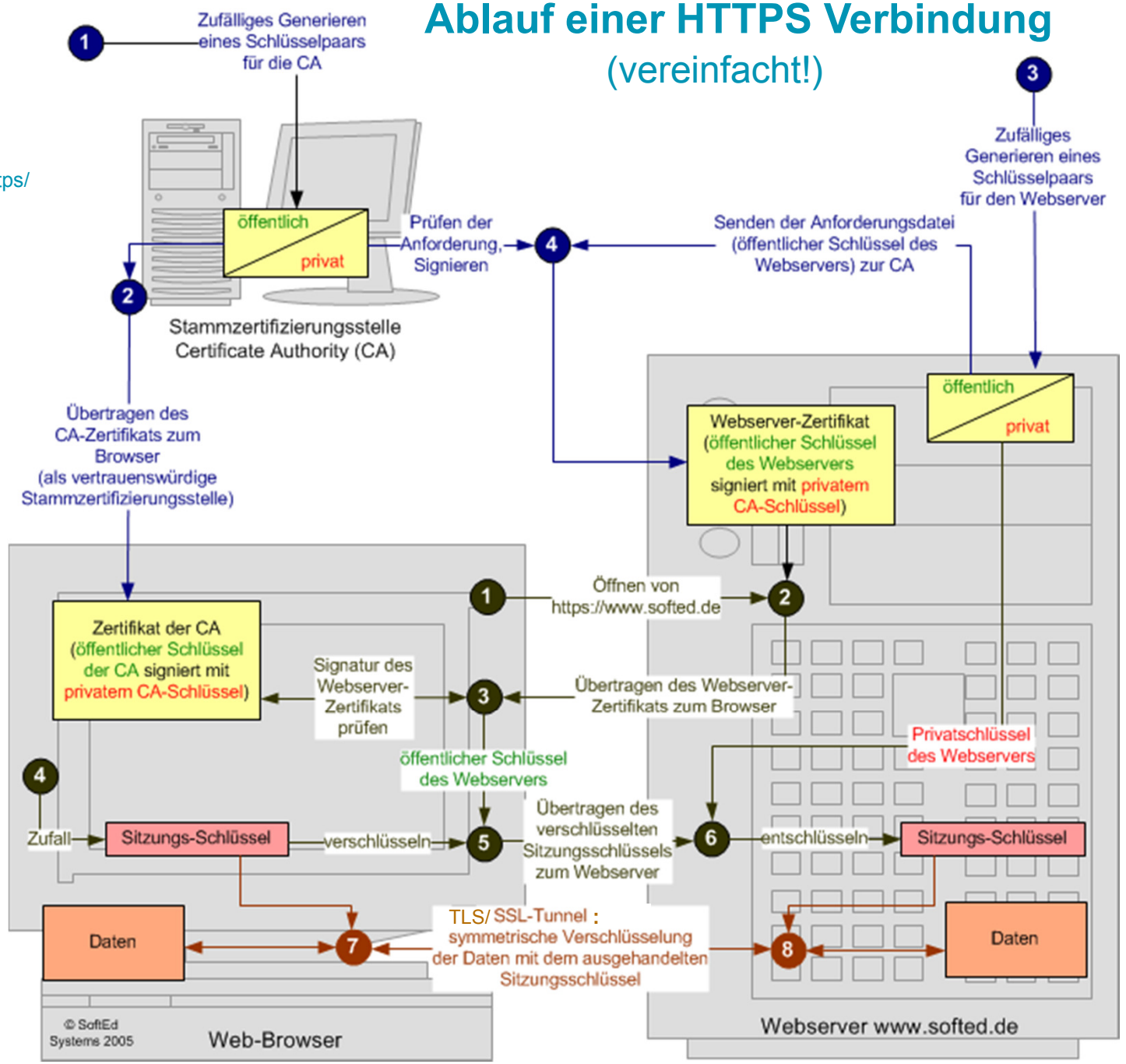
Krypto-Basic 4: Digitale Unterschrift (Authentifizierung)



Ablauf einer HTTPS Verbindung

(vereinfacht!)

Quelle:
https://www.softed.de
/blog/wie-funktioniert-https/



SSL = Secure Sockets Layer
TLS = Transport Layer Security

SSH – Secure Shell

- Haupteinsatz: Sicherer Fernzugriff auf entfernte Geräte aller Art.
  - Authentifizierung:
    Nur Befugte dürfen zugreifen.
  - Vertraulichkeit:
    Kein Unbefugter darf Daten auf der Verbindung einsehen.

- SSH läuft über TCP und benutzt standardmäßig den TCP-Port 22.
  - Allerdings wird der Port 22 von Hackern Tag und Nacht angegriffen. Die Hacker hoffen, einen nicht gut gesicherten SSH-Zugang zu finden.
  - Daher verwenden vorsichtige Menschen für SSH statt Port 22 einen anderen Port (z.B. 22022).
    Vorteil: höhere Sicherheit.
    Nachteil: Unorthodoxe Herangehensweise, die zu überraschenden bzw. lästigen Probleme führen kann („Ich habe zwar meinen SSH-Schlüssel, aber leider die Portnummer vergessen“).

© M. Leischner

SSH – Protokollablauf vereinfacht (Client/Server-Szenario)

Voraussetzung: Server besitzt asymmetrisches Schlüsselpaar (Key-S_{pub}, Key-S_{sec})

- **Schritt 1 – Client:**
  - **TCP-Verbindung aufbauen**
  - **Protokollversion austauschen**
- **Schritt 2 - Server:**
  - **sendet öffentlichen Server-Key Key-S_{pub} und**
  - **Liste der unterstützte Verschlüsselungsalgorithmen**
- **Schritt 3 - Client:**
  - **akzeptiert Server-Key (bereits bekannt bzw. Zertifizierungsstelle fragen)**
  - **wählt Verschlüsselungsalgorithmus**
  - **generiert symmetrischen Session-Key**
  - **sendet diesen Session-Key verschlüsselt mit öffentlichem Server-Key**
- **Schritt 4 - Server:**
  entnimmt Session-Key und schaltet auf Verschlüsselung um.
- **Bei SSH: Schritt 5 - Client:**
  authentifiziert sich in geeigneter Weise
  (das ist der entscheidender Punkt! Es könnte ja jeder kommen.)

© M. Leischner

SSH – Protokollablauf vereinfacht

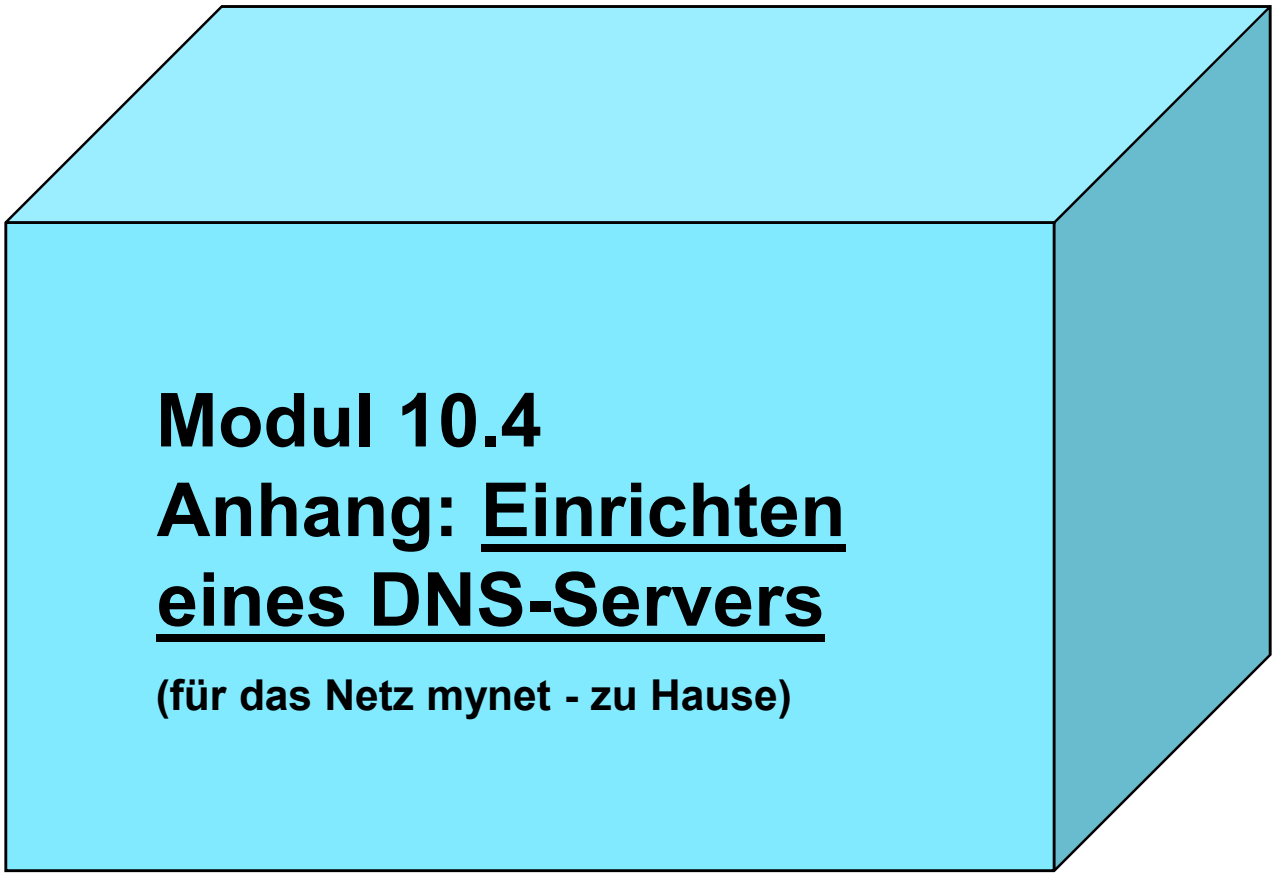
- **Methoden der Clientauthentifizierung**
  - **User-Passwort: User loggt sich über SSH als user mit Passwort PW-S ein**
  - **Authentifizierung über öffentliche Schlüssel (ähnlich digitale Unterschrift):**
    - **User erzeugt im Vorfeld ein asymmetrisches Schlüsselpaar ($Key\text{-}U_{pub}$, $Key\text{-}U_{sec}$) und hinterlegt im Vorfeld seinen öffentlichen Schlüssel $Key\text{-}U_{pub}$ beim Server.**
    - **Der Client verschlüsselt den öffentlichen Server-Schlüssel $Key\text{-}S_{pub}$ mit dem geheimen Userschlüssel $Key\text{-}U_{sec}$. und sendet das Ergebnis**
      $$ERG = enc_{Key\text{-}Usec}\ (Key\text{-}S_{pub})$$
      an den Server.
    - **Der Server besitzt den öffentlichen Schlüssel $Key\text{-}U_{pub}$ des Users. Mit diesem entschlüsselt er das Ergebnis ERG . Kommt als bei der Entschlüsselung $Key\text{-}S_{pub}$ heraus, weiß er, dass der Client den echten geheimen Schlüssel besitzt.**

© M. Leischner

Videobeispiele für Public-Key Verfahren

Quellen:

- **Public-Key-Verfahren: https://www.youtube.com/watch?v=pULfS1_-EQk (1:10min)**
- **What is Public Key Infrastructure (PKI) by SecureMetric: https://www.youtube.com/watch?v=i-rtxrEz_E8 (3:40)**



Modul 10.4
Anhang: Einrichten eines DNS-Servers

(für das Netz mynet - zu Hause)

© M. Leischner

Allgemein: Aufbau des Zone-Files

Mailadresse:
admin@mynet.home

Zone-File: Liste von Resource Records für eine Zone:

- **Ein SOA Resource Record (*Start of Authority*) ist der erste Record im Zone-File. Enthält globale Parameter für die Zone.**

```
@   IN   SOA    ns1.mynet.home. admin.mynet.home. (
                   2019041001 ; serial YYYYMMDDnn
                   14400      ; refresh (4 hours)
                   1800       ; retry   (30 minutes)
                   1209600    ; expire  (2 weeks)
                   3600   )   ; minimum (negative Caching)
```

- `serial`: Seriennummer (wichtig, damit Änderungen autom. übernommen werden).
- `refresh` / `retry` / `expire`: steuern Zusammenspiel zwischen Master / Slave
- `minimum`: (Gültigkeitsdauer eines Negative-Eintrags im Cache)
- `@` steht für die aktuelle Domäne (hier: `mynet.home.`)
- Das `@` kann auch an manchen Stelle durch ein Blank ersetzt werden. Beispiel folgt.

© M. Leischner

Allgemein: Aufbau des Zone-Files

- Ein NS Resource Record definiert einen Name Server für die Zone.
  - Ein Zone File kann mehrere NS Records enthalten.
  - Ein NS Record kann auch auf einen Nameserver eine Subdomäne verweisen.

```
;Name Server Information for domain mynet.home
        IN      NS      ns1.mynet.home.
        IN      NS      ns2.mynet.home.

;IP address of Name Server
ns1      IN      A       192.168.178.11
ns2      IN      A       192.168.178.12
```

- $ORIGIN setzt den Zonenamen für alle darauffolgenden Einträge

```
$ORIGIN subdomain.mynet.home.
```


© M. Leischner

Allgemein: Aufbau des Zone-Files

- Ein A Resource Record weist einem DNS-Namen eine IPv4-Adresse zu:

```
pihole.mynet.home    3600   IN    A      192.168.178.13
```

(Gültigkeitsdauer im Cache: 3600 Sek.)

Kürzere Schreibeweise für diesen Record:

```
pihole    3600   IN    A      192.168.178.13
```

Noch kürzer:

```
pihole    A      192.168.178.13
```

- Ein AAAA Resource Record weist einem DNS-Namen eine IPv6-Adresse zu:

```
dh.docklab.de          3600    N    AAAA     2001:638::1
```

- CNAME Resource Record: Verweist auf einen anderen Domänennamen.
- MX Resource Record: Gibt den Mailserver (FQDN) für die Domäne an.
- PTR Resource Record: Ordnet einer IP-Adresse einen Domänennamen zu (FQDN). Anwendung für Reverse Lookup.

```
13    IN     PTR   pihole.mynet.home. 178.168.192.in-addr.arpa.
```

Eintrag im Zone-File der Reverse-Lookup-Zone

© M. Leischner

DNS Software

- **BIND**
  Standard DNS-Server. Offizielle aktuelle ist Version `bind10`, eingesetzt wird aber `bind9`.
  `Bind9` ist auch die offizielle Version im Ubuntu 18.04-Repo.
- **PowerDNS**
  Open source, Konkurrent zu BIND, BIND-kompatible Datenspeicherung
- **Unbound**
  High-Performance DNS-Server. Modularer Aufbau.
- **Dnsmasq**
  Leichter, einfach zu konfigurierender DNS-Server. Wird im Pi_hole verwendet

- **Pi-Hole**
  Spezial-DNS-Server, der Werbung im ganzen Netz blockt.

© M. Leischner

Installation und Konfiguration von BIND

```
$ apt install bind9
```

Erzeugt im Verzeichnis `/etc/bind` u.a. die folgenden Konfigurationsfile:

- `named.conf` — Lädt nur die folgenden drei Config-Files.
- `named.conf.options` — Setzt Optionen für den DNS-Server
- `named.conf.default-zones` — Enthält standardmäßig gesetzte Zonen, z.B. localhost
- `named.conf.local` — Hier werden vom Administrator, die von ihm verwalteten Zonen definiert. (insbesondere Zonennamen + Zonenfiles)

Für jede Domäne gibt es zwei Zonen: Forward-Lookup, Reverse-Lookup .
Eine Zonendefinition verweist auf einen Zone-File, z.B.

`fwd.netlab.inf.h-brs.de.db` oder `rev.netlab.inf.h-brs.de.db`

der die Ressourcen-Records der entsprechenden Zone enthält. Hier werden neue DNS-Einträge hinzugefügt.

mynet-Konfiguration von `named.conf.options`

```
options {
        directory "/var/cache/bind";

        recursion yes;                  # enables resursive queries
        allow-recursion { trusted; };   # allows recursive queries from
                                        # "trusted" clients
        listen-on { 192.168.178.11; }; # ns1 private IP address
                                        # - listen on private network only
        allow-transfer { none; };       # disable zone transfers by default

        forwarders {
                192.168.178.1;
                1.1.1.1;
                8.8.4.4;
        };

        dnssec-validation auto;

        auth-nxdomain no;    # conform to RFC1035
        listen-on-v6 { any; };
}
```

© M. Leischner

Allgemeines Beispiel einer Master-Zonenbeschreibung

```
zone "netlab.inf.h-brs.de" IN {
    type master;
    file "fwd.netlab.inf.h-brs.de.db";
      # Forward zone lookup file with resource records
};
```

Zonentypen:

- **master:** Enthält alle Daten der Zone und liefert eine „authoritative answers“.
- **slave:** Replikation der Master-Zone.
- **forward:** Nur Weiterleitung an andere Server

mynet-Konfiguration von `named.conf.local` (Zonennamen + Zonenfiles)

```
zone "mynet.home" IN {          # Domain name
    type master;                # Primary DNS
    file "/etc/bind/fwd.mynet.home.db";
                                # Forward lookup file
    allow-update { none; };     # Since this is a primary DNS,
                                # it should be none.
};

zone "178.168.192.in-addr.arpa" IN { # Reverse lookup name,
                                # should match your network
                                # in reverse order
    type master;                # Primary DNS
    file "/etc/bind/rev. mynet.home.db";
                                # Reverse lookup file
    allow-update { none; };     # Since this is a primary DNS,
                                # it should be none.
};
```


© M. Leischner

mynet-Konfiguration für den Forward-Zone-File

```
$TTL 1d
@       IN      SOA     ns1.mynet.home. admin.mynet.home. (
                        18122001        ; Serial
                              3h        ; Refresh
                             15m        ; Retry
                              3w        ; Expire
                              3h )      ; Negative Cache TTL

;Name Server Information
@       IN      NS      ns1.mynet.home.
@       IN      NS      ns2.mynet.home.
;IP address of Name Server
ns1     IN      A       192.168.178.11
ns2     IN      A       192.168.178.12

;A - Record HostName To Ip Address
fritzbox        IN      A       192.168.178.1
ns1             IN      A       192.168.178.11
ns2             IN      A       192.168.178.12
pihole          IN      A       192.168.178.13
```

mynet-Konfiguration für den Reverse-Zone-File

```
$TTL 1d
@       IN      SOA     ns2.mynet.home. admin.mynet.home. (
                        19041001        ; Serial
                             3h         ; Refresh
                            15m         ; Retry
                             3w         ; Expire
                             3h )       ; Negative Cache TTL

;Name Server Information
@       IN      NS      ns1.mynet.home.
@       IN      NS      ns2.mynet.home.
;Reverse lookup for Name Server
11      IN      PTR     ns1.mynet.home.
12      IN      PTR     ns2.mynet.home.

;PTR Record IP address to HostName
1       IN      PTR      fritzbox.mynet.home.
11      IN      PTR      ns1.mynet.home.
12      IN      PTR      ns2.mynet.home.
13      IN      PTR      pihole.mynet.home.
```


© M. Leischner

Literatur

1. DNS for Rocket Scientists: http://www.zytrax.com/books/dns/, Zugriff am 09.04.2019.
2. Ron Aitchison: Pro DNS and BIND 10, Springer Science+Business Media, 2011
3. Wiki: Domain Name System, https://de.wikipedia.org/wiki/Domain_Name_System , Zugriff am 09.04.2019.
4. Wiki: Zonendatei, https://de.wikipedia.org/wiki/Zonendatei , Zugriff am 10.04.2019.

© M. Leischner