Computernetzwerke

Anwendungsschicht

Sebastian Bauer

Wintersemester 2022/2023

Computer Engineering Curriculum

Hybrides Referenzmodell: Anwendungsschicht

Outline

DNS – Einleitung

- IP-Adressen schwierig für Menschen (aber perfekt für Computer)
- Symbolische Namen für uns einfacher zu handhaben
- Wie werden symbolische Namen auf IP-Adressen (und umgekehrt) abgebildet?

DNS – Einleitung

- IP-Adressen schwierig für Menschen (aber perfekt für Computer)
- Symbolische Namen für uns einfacher zu handhaben
- Wie werden symbolische Namen auf IP-Adressen (und umgekehrt) abgebildet?

DNS – Domain Name System

- Spezifiziert in RFC 1034 und RFC 1035
- Löst lesbare Namen von Rechnern zu IP-Adressen auf: `htw-berlin.de` → `141.45.66.214`
- Komplementiert lokale Datenbank (`/etc/hosts`)

DNS – Konzepte

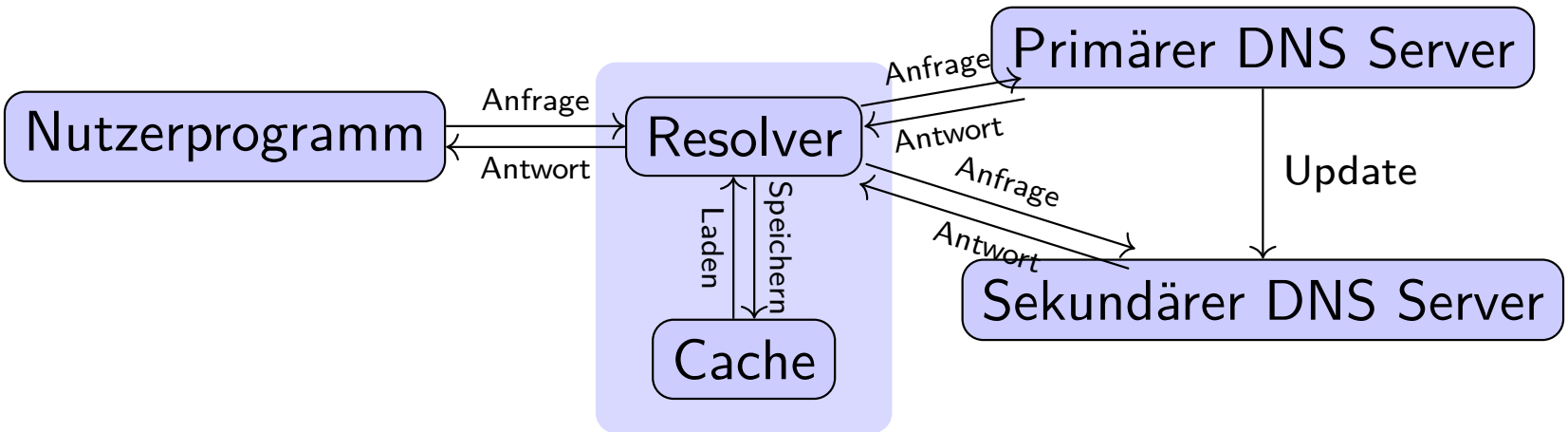
- Eine verteilte Datenbank:

DNS – Konzepte

- Eine verteilte Datenbank: verschiedene Namensserver verwalten unterschiedliche Domänen (Skalierbarkeit)
- Daten einer Domäne sind über Client/Server-Architektur für das gesamte Netzwerk verfügbar

  Namensserver Server der einen (kleinen) Teil der Datenbank verwaltet

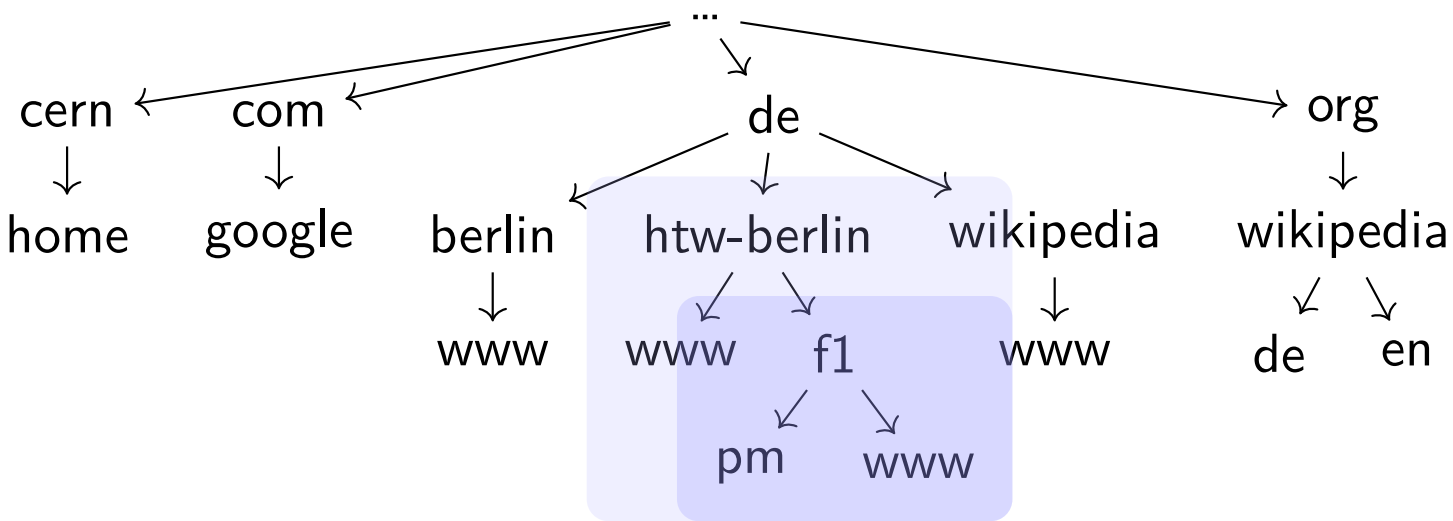
  Resolver Client der Domänennamen anfragt
- Robustheit ergibt sich aus Replikation und Caching



DNS – Aufbau des Domain-Namensraums

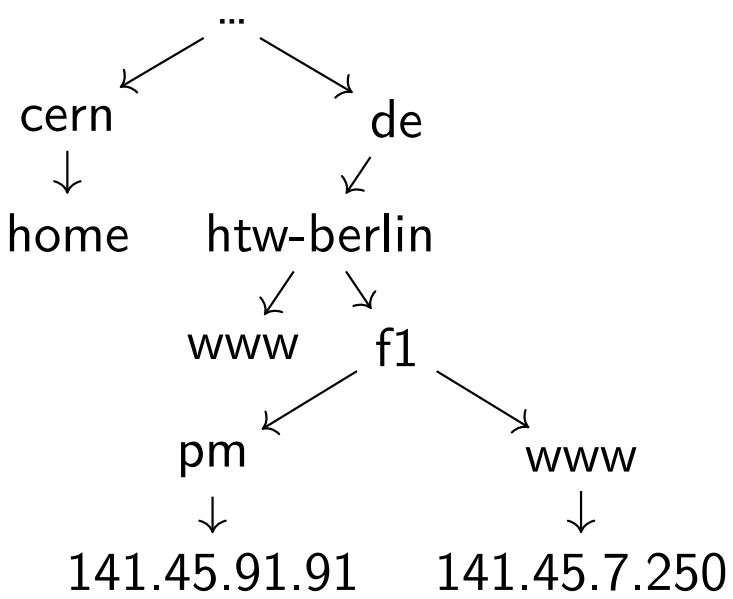
Baumförmige Struktur:

- Knoten (innere und Blätter) repräsentieren Labels
  - Zwischen 0 und 63 Zeichen, Umlaute seit 2004 (IDN)
  - Groß- und Kleinschreibung irrelevant
- Domain-Name eines Knotens ist Pfad zur Wurzel
- Jeder Unterbaum ist eine Domain (dt. *Domäne*)
- Jede Domäne kann Unterdomänen enthalten



DNS-Datenbank

- Blätter enthalten IP-Adressen und andere Daten

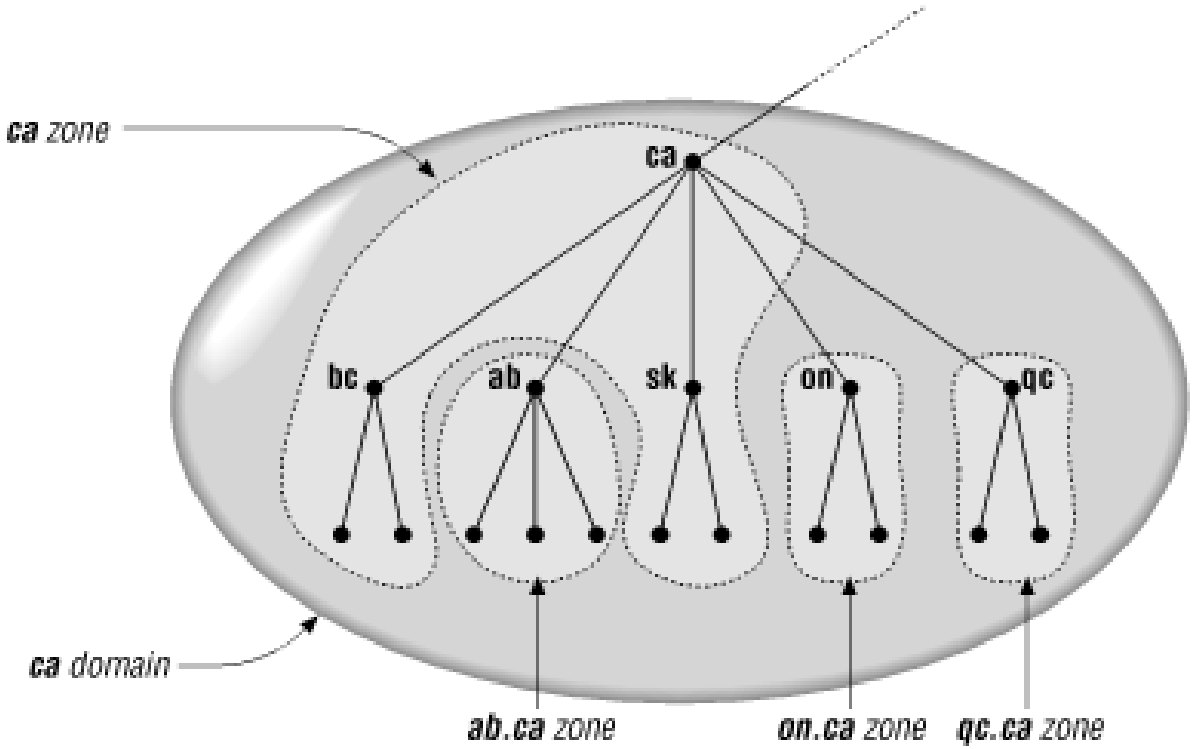


- Daten, die zu einem Domänennamen gehören, werden in Resource Record (RR) gespeichert

DNS – Zonen

Zonen

- sind Partitionen eines Domänenbaums, für die ein (primärer) Namensserver bestimmend ist



https://nnc3.com/mags/Networking2/dns/ch02_04.htm

DNS – Resource Records

- *Resource Records* liegen auf einem Namensserver
- Defacto-Standard Namenserversoftware ist BIND (seit 1986 entwickelt)

Resource Records können Textdateien sein, mit Format:

```
<name> [<ttl>] [<class>] <type> [<rdlength>] <rdata>

<name>     : Domainname
<ttl>      : Gültigkeitsdauer des Eintrags
<class>    : Klasse, meistens IN (Internet)
<type>     : Typ des Records (z.B. A)
<rdlength> : Länge der Daten
<rdata>    : Daten
```

DNS – Typen von Resource Records

A	Verweist auf eine IPv4-Adresse
AAAA	Verweist auf eine IPv6-Adresse
CNAME	Verweist auf einen anderen (kanonischen) Domainnamen
MX	Verweist auf einen Server für den Austausch von E-Mails (Mail-Exchange)
NS	Verweist auf einen Namensserver für diese Domain
SOA	Enthält Angaben zur Verwaltung der Zone (z.B. Seriennummer, E-Mail,...)

Weitere Records auf
https://de.wikipedia.org/wiki/Resource_Record

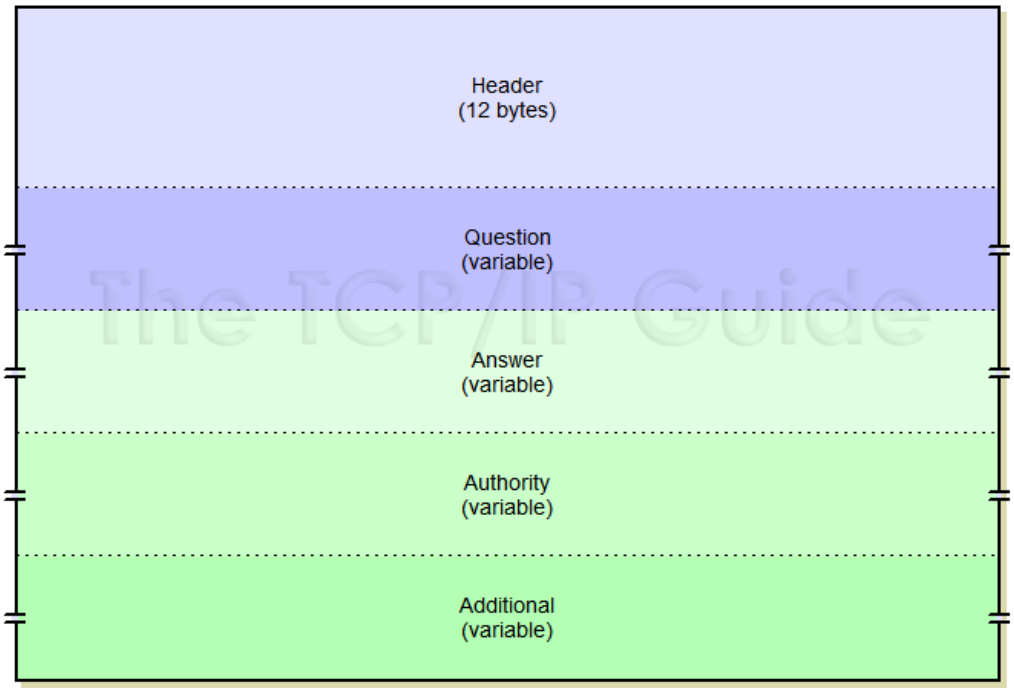
DNS – Beispiel

```
;name                    ttl   cl  type   rdata

www.htw-berlin.de.       2911  IN  CNAME  moehre.htw-berlin.de.
moehre.htw-berlin.de.    2297  IN  A      141.45.7.250

htw-berlin.de.            600  IN  MX     50 mail1.rz.htw-berlin.de.
mail1.rz.htw-berlin.de.   401  IN  A      141.45.10.101
```

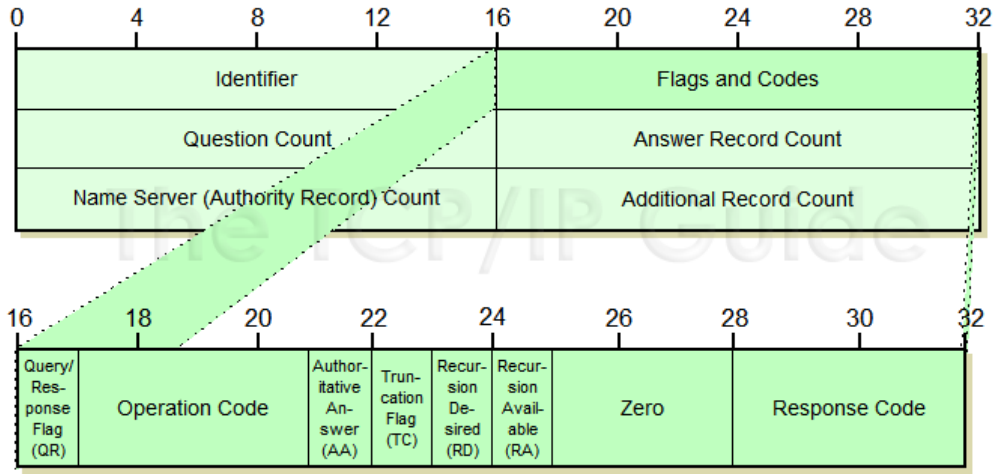
DNS – Protokoll

- Standartport: 53, für UDP und TCP
- Resolver ↔ DNS Server
  - UDP, falls Nachricht kleiner als 512 Bytes
  - Sonst: Nachricht bei 512 Bytes kappen und trotzdem via UDP senden, dann nochmal das gleiche via TCP

DNS – Nachrichtenformat



DNS – Header


0
4
8
12
16
20
24
28
32
Identifier
Flags and Codes
Question Count
Answer Record Count
Name Server (Authority Record) Count
Additional Record Count
16
18
20
22
24
26
28
30
32
Query/ Res- ponse Flag (QR)
Operation Code
Author- itative An- swer (AA)
Trun- cation Flag (TC)
Recur- sion De- sired (RD)
Recur- sion Avail- able (RA)
Zero
Response Code

Arten von Antworten auf Anfragen an Server

Autoritativ:

- Server holt Antwort aus der lokalen Zonendatei

Arten von Antworten auf Anfragen an Server

Autoritativ:

- Server holt Antwort aus der lokalen Zonendatei

Iterativ:

- Server kennt Antwort nicht selbst
- Schickt Verweis auf anderen Server
  (ich weiß es nicht, aber Server Y
  könnte es wissen)

Arten von Antworten auf Anfragen an Server

Autoritativ:

- Server holt Antwort aus der lokalen Zonendatei

Iterativ:

- Server kennt Antwort nicht selbst
- Schickt Verweis auf anderen Server (ich weiß es nicht, aber Server Y könnte es wissen)

Rekursiv:

- Server kennt Antwort nicht selbst
- Schickt eigene Anfrage an anderen Server
- Schickt Antwort dann an Client

Arten von Antworten auf Anfragen an Server

Autoritativ:

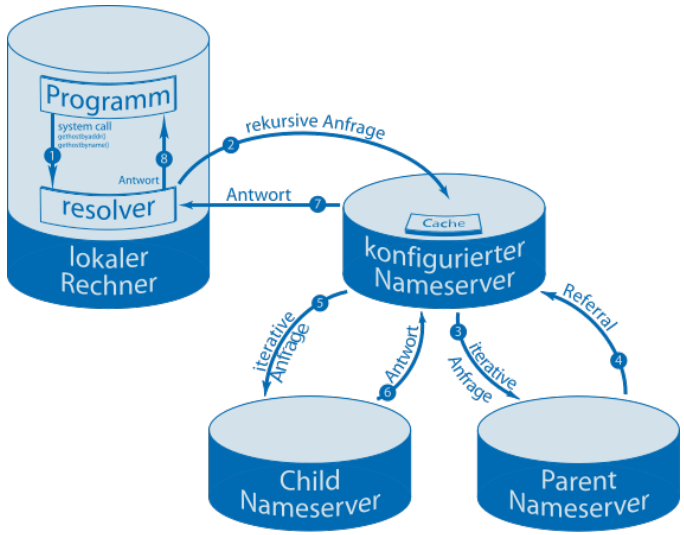
- Server holt Antwort aus der lokalen Zonendatei

Iterativ:

- Server kennt Antwort nicht selbst
- Schickt Verweis auf anderen Server (ich weiß es nicht, aber Server Y könnte es wissen)

Rekursiv:

- Server kennt Antwort nicht selbst
- Schickt eigene Anfrage an anderen Server
- Schickt Antwort dann an Client



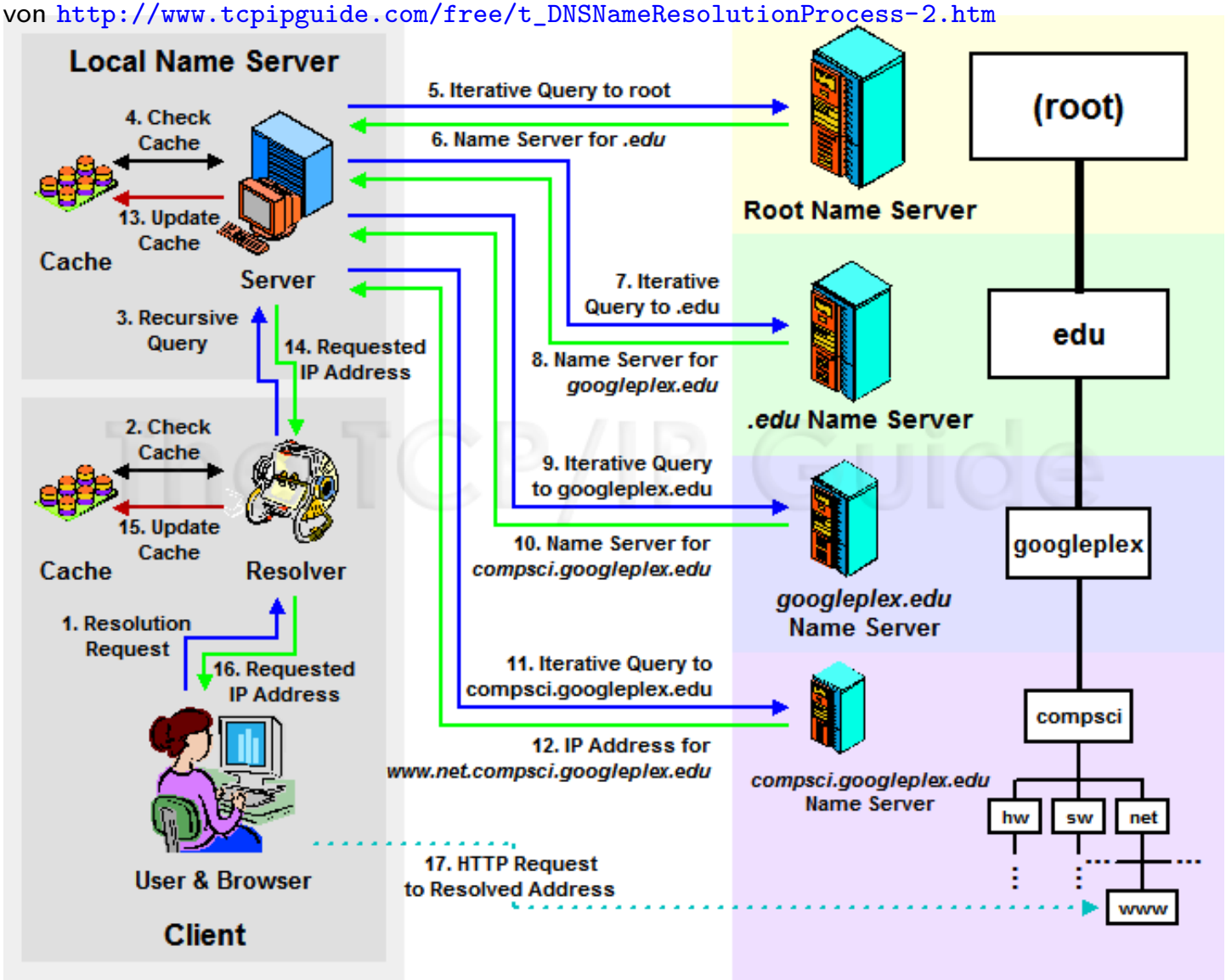
DNS – Rootserver

- Auflösen eines Domänennamens startet bei Wurzel (Rootzone)
- Zunächst werden Root-DNS-Server befragt
- Antwort erfolgt in der Regel iterativ (z. B. für `.de`-Domains ist u. a. `f.nic.de` zuständig)
- 13 Haupt-DNS-Root-Server gibt es, durch Anycast sind es physisch mehr (siehe https://root-servers.org/)

Woher kennt ein Rechner die DNS-Root-Server?

DNS – Beispielabfrage

von http://www.tcpipguide.com/free/t_DNSNameResolutionProcess-2.htm



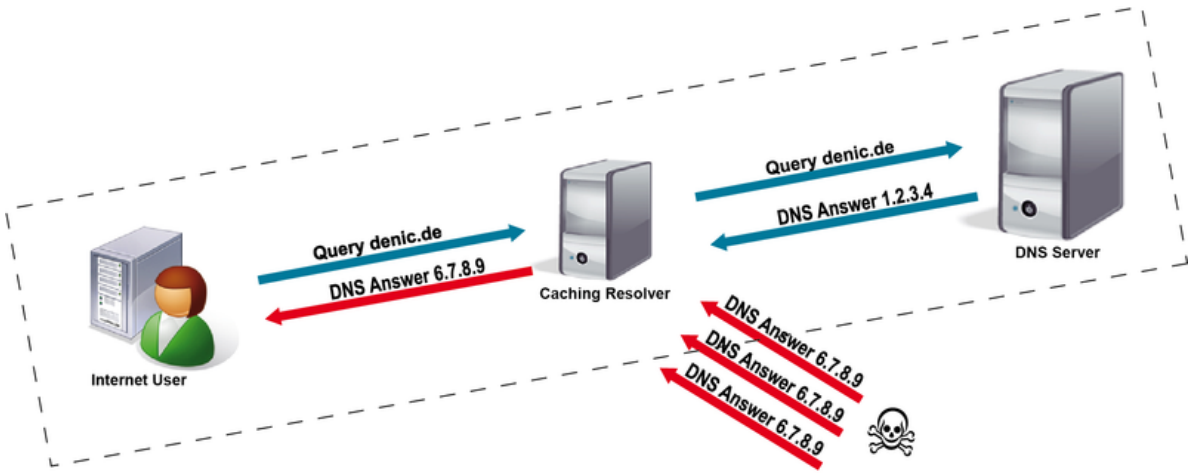
DNS-Tool: dig

```
$ dig -t MX htw-berlin.de
; <<>> DiG 9.11.5-P4-5.1+b1-Debian <<>> -t MX htw-berlin.de
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 40073
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;htw-berlin.de. IN MX

;; ANSWER SECTION:
htw-berlin.de. 600 IN MX 50 mail1.rz.htw-berlin.de.

;; ADDITIONAL SECTION:
mail1.rz.htw-berlin.de. 56 IN A 141.45.10.101

;; Query time: 31 msec
;; SERVER: ::1#53(::1)
;; WHEN: Thu Dec 05 22:20:55 CET 2019
;; MSG SIZE  rcvd: 83
```

```
$ dig +trace www.htw-berlin.de
; <<>> DiG 9.11.5-P4-5.1+b1-Debian <<>> +trace www.htw-berlin.de
;; global options: +cmd
. 42384 IN NS c.root-servers.net.
. 42384 IN NS d.root-servers.net.
. 42384 IN NS e.root-servers.net.
. 42384 IN NS f.root-servers.net.
. 42384 IN NS g.root-servers.net.
. 42384 IN NS h.root-servers.net.
. 42384 IN NS i.root-servers.net.
. 42384 IN NS j.root-servers.net.
[...]
;; Received 1097 bytes from ::1#53(::1) in 3 ms

de. 172800 IN NS a.nic.de.
de. 172800 IN NS f.nic.de.
de. 172800 IN NS l.de.net.
de. 172800 IN NS n.de.net.
de. 172800 IN NS s.de.net.
de. 172800 IN NS z.nic.de.
[...]
;; Received 777 bytes from 193.0.14.129#53(k.root-servers.net) in 35 ms

htw-berlin.de. 86400 IN NS dns-2.dfn.de.
htw-berlin.de. 86400 IN NS infobloxv.htw-berlin.de.
[...]
;; Received 653 bytes from 194.146.107.6#53(n.de.net) in 27 ms

www.htw-berlin.de. 3600 IN CNAME moehre.htw-berlin.de.
moehre.htw-berlin.de. 3600 IN A 141.45.7.250
htw-berlin.de. 28800 IN NS infobloxv.htw-berlin.de.
htw-berlin.de. 28800 IN NS dns-2.dfn.de.
;; Received 219 bytes from 193.174.75.54#53(dns-2.dfn.de) in 111 ms
```

Sicherheitsprobleme von DNS

Versendete Informationen von DNS sind manipulierbar, z. B. via Cache-Poisoning

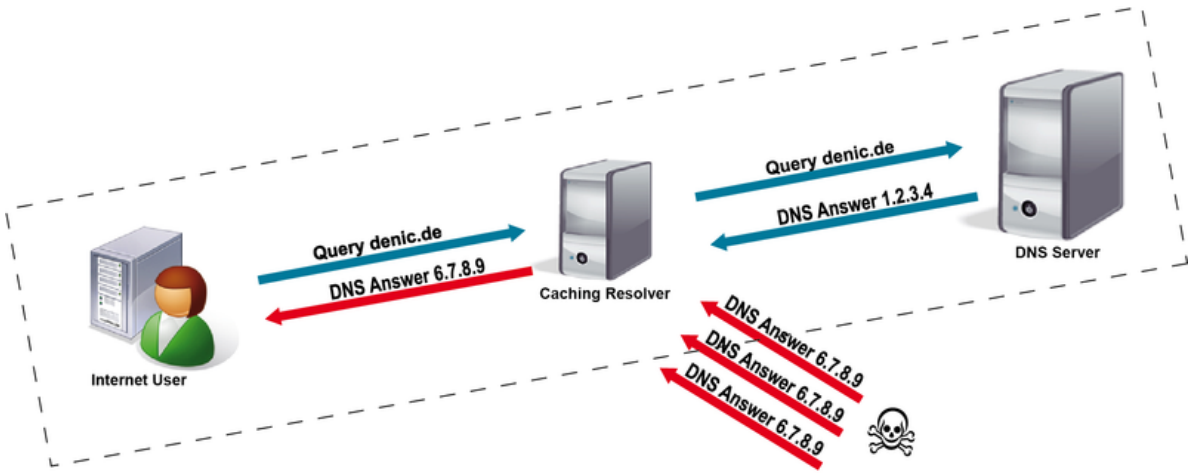


von https://www.denic.de/wissen/dnssec/

Welche Angriffsarten ermöglicht Cache-Poisoning?

Sicherheitsprobleme von DNS

Versendete Informationen von DNS sind manipulierbar, z. B. via Cache-Poisoning



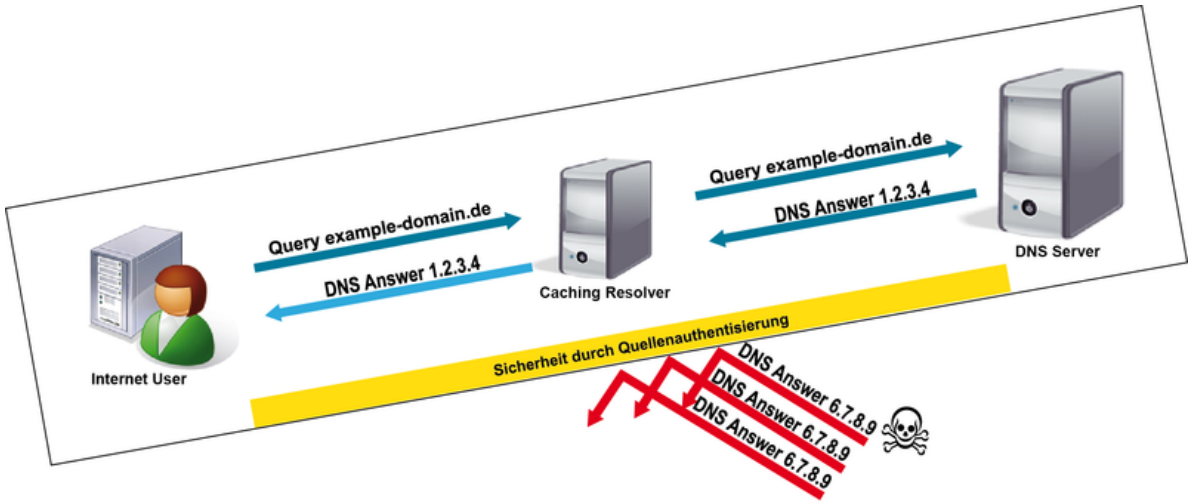
von https://www.denic.de/wissen/dnssec/

Welche Angriffsarten ermöglicht Cache-Poisoning?

- Phishing
- Denial-of-Service

DNSSEC

- DNSSEC sichert die Übertragung mit Hilfe digitaler Signaturen (Public-Key-Verfahren) ab
- Aber: Absicherung ≠ Vertraulichkeit



Funktionsweise von DNSSEC

- Kompatibel zur DNS: weitere Resource Records (RR)
- Jede Zone generiert ein Zonenschlüssel (aufgeteilt in privater und öffentlicher Schlüssel)
- Öffentlicher Teil landet im DNSKEY RR
- Jeder andere Eintrag der Zone wird mit privaten Schlüssel digital signiert, gespeichert im RRSIG
- Clients können jetzt überprüfen, ob signierter RR von der Zone kommt

DNSSEC – Beispiel

```
child.example. IN DNSKEY (
                        257          ; Schlüsselunterzeichnungs-Schlüssel
                        3            ; DNSSEC
                        1            ; RSA-Verschlüsselung
                        AQOW4333ZLdOHLRk+3Xe6RAaCQAOMhAVJu2T
                        xqmk1Kyc13h69/wh1zhDk2jjqxsN6dVAFi16
                        CUoynd7/EfaXdcjL )

www.child.example. 1285  A    1.2.3.15
www.child.example. 1285  IN                ; Klasse zu der der RR gehört
                         RRSIG             ; RR ist vom Typ RRSIG
                         A                 ; Signierter Typ ist A
                         3                 ; DSA-Encryption
                         3                 ; Name hat 3 Komponenten
                         1285              ; Original-TTL
                         (
                         20040327122207    ; Endzeitpunkt
                         20040226122207    ; Anfangszeitpunkt
                         22004             ; eindeutige Nummer/Key Tag
                         child.example.    ; Name des Unterzeichners
                         BMTLR80WnKndatr77...BtprR9SLKoZUiPWX ; Hash
                         )
```

DNSSEC – Probleme?

Wie viel bringt das bisher?

DNSSEC – Probleme?

Wie viel bringt das bisher? Rein gar nichts, da auch Zonenschlüssel gefälscht sein kann

Wie lässt sich die Korrektheit des Zonenschlüssels feststellen?

DNSSEC – Probleme?

Wie viel bringt das bisher? Rein gar nichts, da auch Zonenschlüssel gefälscht sein kann

Wie lässt sich die Korrektheit des Zonenschlüssels feststellen? Vertrauenskette (engl. *chain of trust*)

DNSSEC – Vertrauenskette

- Öffentliche Teil des Schlüssels einer Zone (A) wird durch DS RR der übergeordneten Zone (B) validiert
- Dieser RR wird mit Schlüssel von B signiert
- Schlüssel von B wird von der nächst übergeordneten Zone validiert
- usw.

DNSSEC – Vertrauenskette

- Öffentliche Teil des Schlüssels einer Zone (A) wird durch DS RR der übergeordneten Zone (B) validiert
- Dieser RR wird mit Schlüssel von B signiert
- Schlüssel von B wird von der nächst übergeordneten Zone validiert
- usw.

Zwei Fragen

1. Bis wohin geht das?

DNSSEC – Vertrauenskette

- Öffentliche Teil des Schlüssels einer Zone (A) wird durch DS RR der übergeordneten Zone (B) validiert
- Dieser RR wird mit Schlüssel von B signiert
- Schlüssel von B wird von der nächst übergeordneten Zone validiert
- usw.

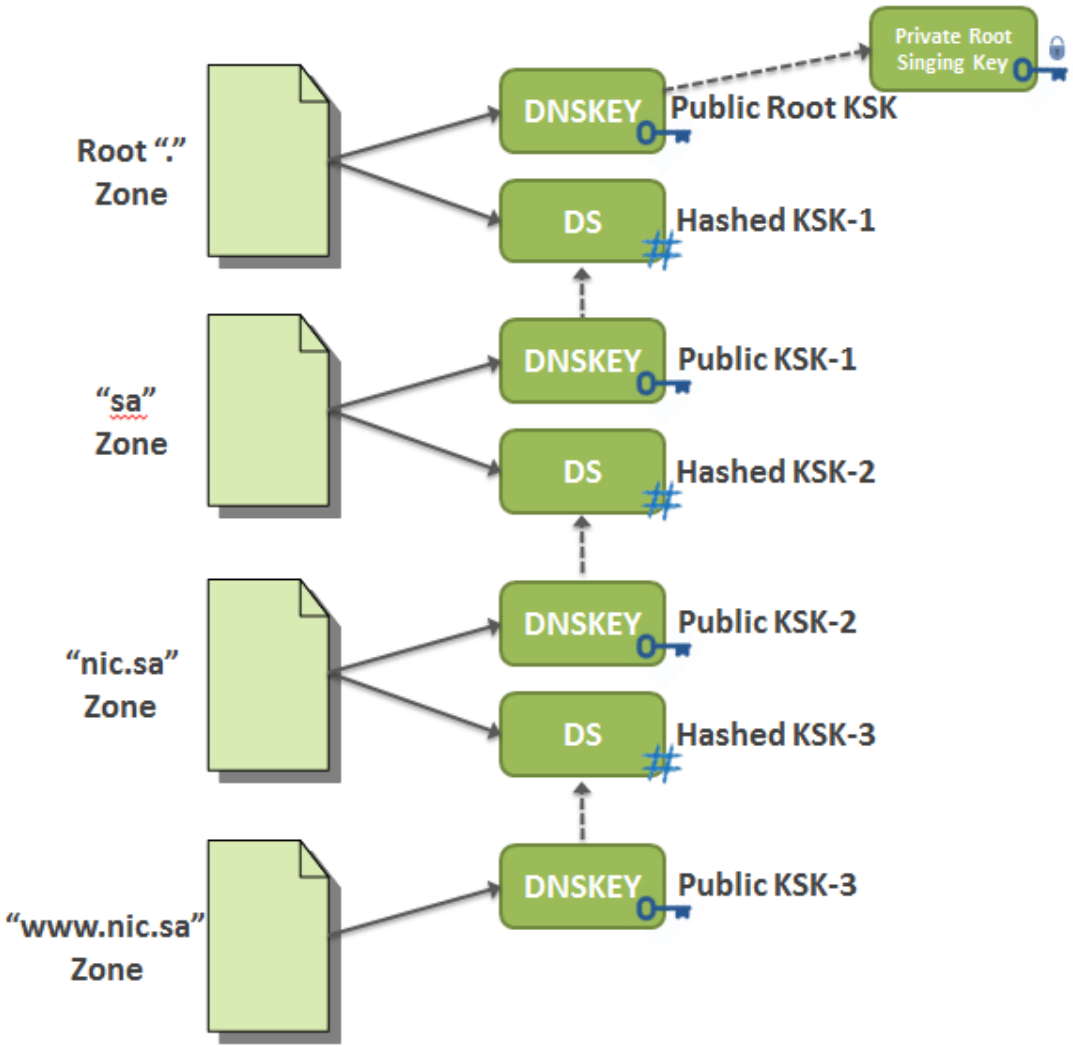
Zwei Fragen

1. Bis wohin geht das?
2. Wer validiert den Schlüssel der Root-Zone?

DNSSEC – Schlüsselvalidierungsbeispiel

```
filiale1.example.org.   NS nsf
filiale1.example.org.   DS        ; Typ
                        52037     ; Identifikationsnummer
                        1         ; Verschlüsselungsverfahren
                        1         ; Hash-Typ
                        378929E92D7DA04267EE87E802D75C5CA1B5D280
```

DNSSEC – Beispiel zur Vertrauenskette



Outline

DHCP – Einleitung

- Manuelle Konfiguration der Clients ist in größeren Netzwerken umständlich
- Dynamic Host Configuration Protocol (DHCP) erlaubt Clients, Netzwerkkonfiguration von einem Server zu beziehen
- Dazu zählen:
  - IP-Adresse
  - Netzmaske
  - Gateway / Router
  - DNS
  - ...
- Spezifiziert in RFC 2131

DHCP – Historisches

Vorläufer:

- RARP
  - Übermittelt nur IP-Adresse (keine andere Daten)
  - Spezifiziert in RFC 903
- BOOTP
  - Sitzt Anwendungsschicht (welcher Vorteil?), nutzt UDP
  - Weitere Netzwerk-Eigenschaften können übermittelt werden (z.B. Gateways)
  - Definiert in RFC 951
  - Erweitert in RFC 1497 (Subnetmasken, DNS, ...)

⇒ Im Gegensatz zu DNS wurde die „Technik“ im Laufe der Zeit überarbeitet.

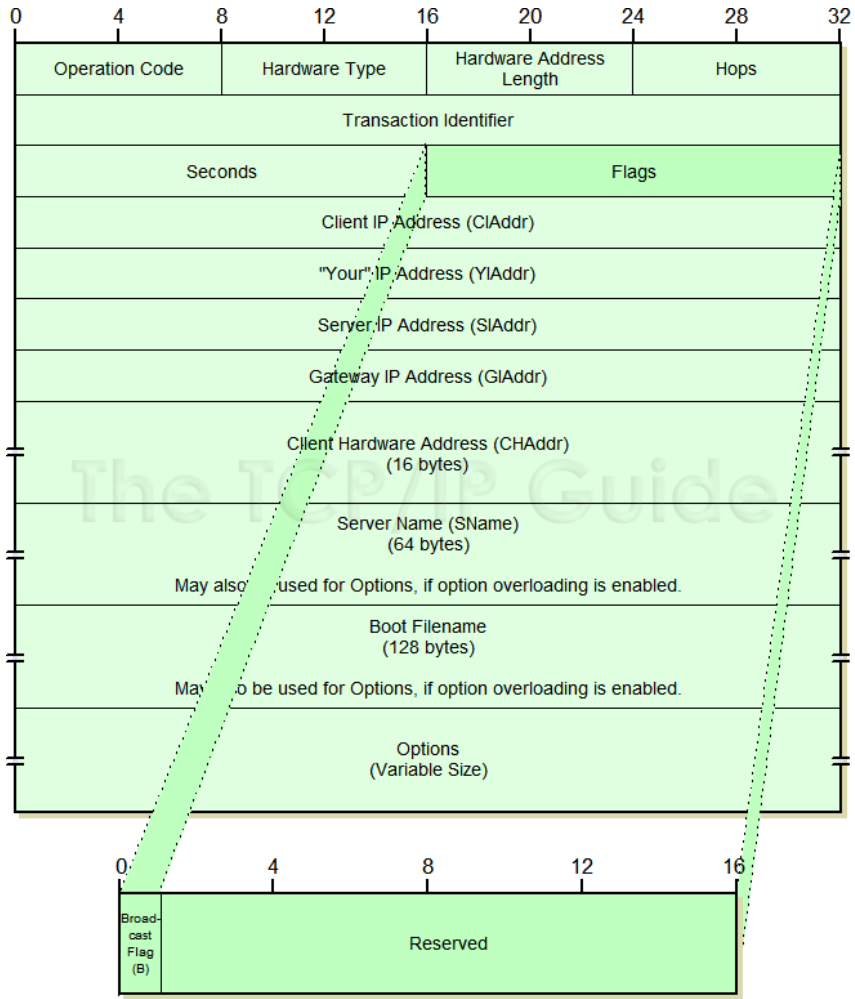
Betriebsmodi zur IP-Adressenzuweisung

Manuell Vergebene IP-Adresse wird fest an MAC Adresse gebunden. Sinnvoll bei Server, Drucker, etc.

Automatisch Unbekannte MAC-Adressen erhalten IP-Adresse aus definierten Bereich. Zuordnung bleibt auf Dauer bestehen.

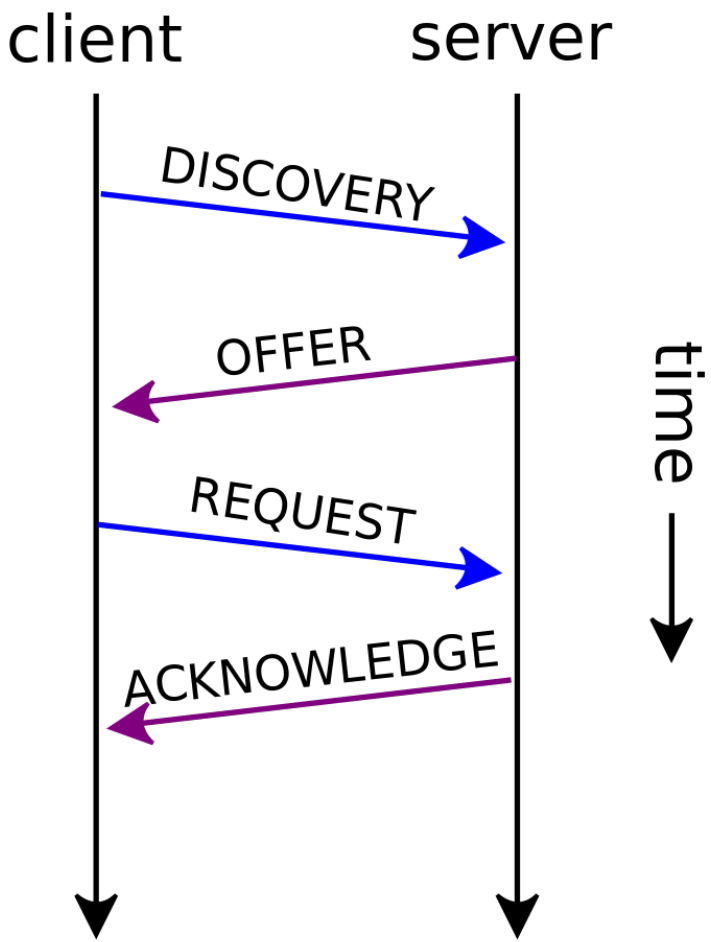
Dynamisch IP-Adresse wird nur einen bestimmt Zeit verliehen. Client muss sich kurz vor Ablauf der Zeit neu melden.

Aufbau eines DHCP-Paketes



Siehe auch https://en.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol.

Ablauf



By Gelmo96 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=38179484

Henne-Ei-Problem

1. Für den *Discovery* kennt der Client den Server evtl. nicht. An welche Adresse sind die Pakete zu senden?

Henne-Ei-Problem

1. Für den *Discovery* kennt der Client den Server evtl. nicht. An welche Adresse sind die Pakete zu senden?
2. Wie kann die *Offer* den Client erreichen, der seine IP-Adresse noch nicht kennt (zwei Möglichkeiten)?

DHCP-Server

Linux:

- `dnsmasq`: Tutorial
- `isc-dhcp-server`: Tutorial 1, Tutorial 2
- `udhcpd`: Tutorial

Sonstiges:

- Netzwerk- bzw. Internetverbindung teilen

DHCP und IPv6

- DHCPv6 ist Analogon zu DHCP für IPv6
- Alternative dazu ist Stateless Address Autoconfiguration (SLAAC):
  - Entspricht in etwa der Auswahl einer Link-Lokal-Adresse bei IPv4
  - Etwas einfacher: IPv6 Adresse enthält MAC-Adresse

Outline

Wozu?

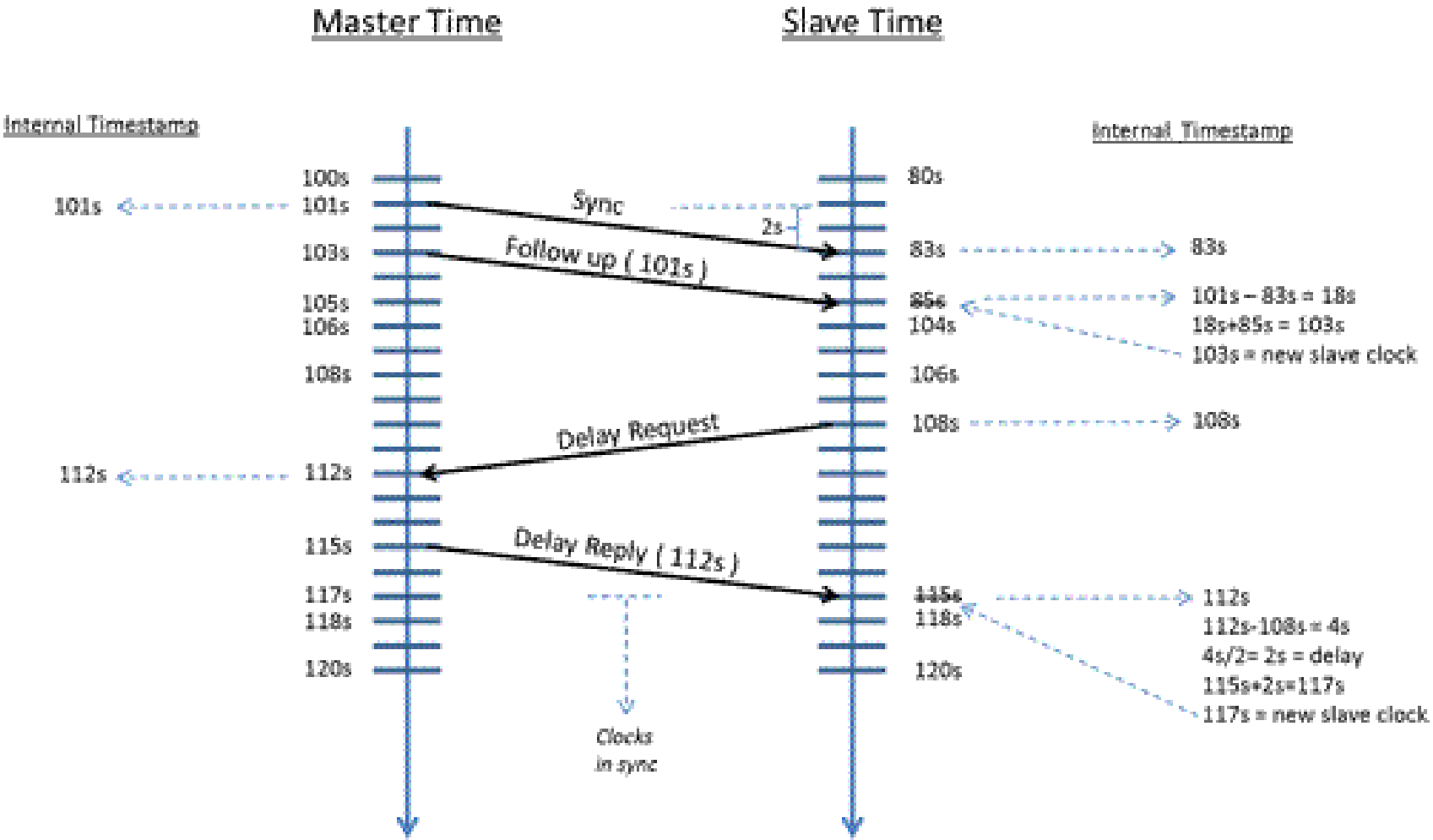
Gleich laufende Uhren essenziell für viele Anwendungen:

- Entfernungsmessung
- Positionsbestimmung
- Finanztransaktionen

Eigenschaften

- Hauptfokus: LAN (keine WANs)
- Ressourcenfreundlich
- Synchronisiert Uhren unterschiedlicher Qualität
- Protokollunabhängig
- Kein spezieller Software-Setup nötig
- Hohe Genauigkeit, falls Hardware mitspielt

Übersicht – Ablauf



Wie?

https://www.youtube.com/watch?v=Forh3XfD_Ec

Outline

Wozu noch eine Adressierung?

- Bisher: MAC-Adressen, IP-Adressen, Ports
- DNS erlaubt Namen für IP-Adressen (=Geräte) zu nutzen

Wozu noch eine Adressierung?

- Bisher: MAC-Adressen, IP-Adressen, Ports
- DNS erlaubt Namen für IP-Adressen (=Geräte) zu nutzen
- Aber wie können Nutzer einzelne Ressourcen bzw. Dienste kommunizieren und ansprechen?

Wozu noch eine Adressierung?

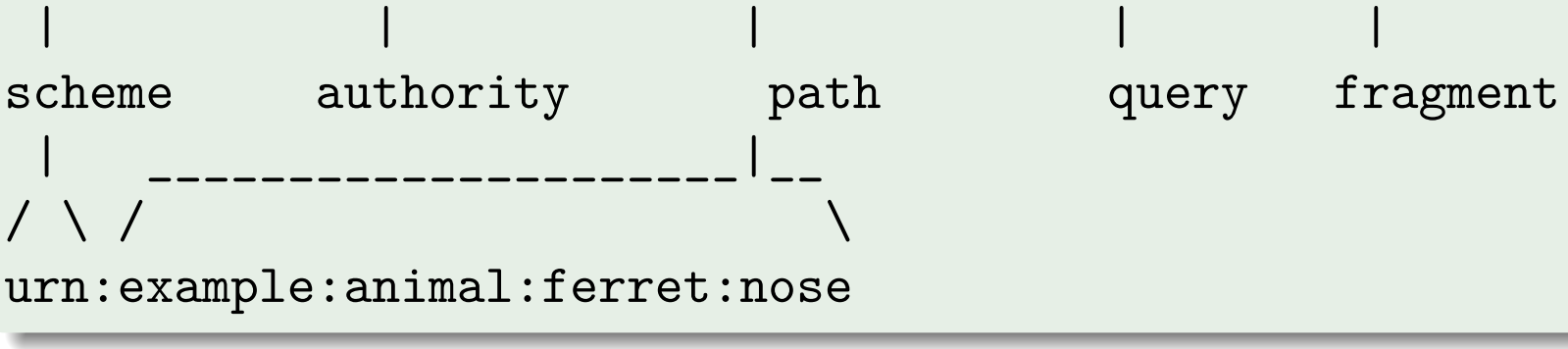
- Bisher: MAC-Adressen, IP-Adressen, Ports
- DNS erlaubt Namen für IP-Adressen (=Geräte) zu nutzen
- Aber wie können Nutzer einzelne Ressourcen bzw. Dienste kommunizieren und ansprechen?
- Hierfür gibt es Uniform Resource Identifer (URI)
- Erste Definition in RFC 1738, aktuelle in RFC 3986.

URI – Grammatik

URI – Uniform Resource Identifer

```
URI = scheme:[//authority]path[?query][#fragment]
```

Zwei Beispiele



Scheme (Protokoll)

- Ist der Teil vor dem Doppelpunkt
- Definiert Kontext des URIs, damit Bedeutung des weiteren Teils

http	Webseite über HTTP (URL)
https	Webseite über gesichertes HTTP (URL)
ftp	Datei bzw. Verzeichnis auf FTP-Server (URL)
file	Datei bzw. Verzeichnis auf lokalen Rechner (URL)
urn	Ortsunabhängiger eindeutiger Bezeichner für eine Ressource, z.B. `urn:isbn:1-4398-3665-5`
doi	Digital object identifier
mailto	E-Mail-Adresse

Authority (Zuständigkeit)

```
authority = [userinfo@]host[:port]
userinfo = username[:password]
```

- Teil der URI bei Schemas wie `http` und `ftp`
- Host:
  - IP-Literal mit eckigen Klammern (IPv6)
  - IP-Literal mit Punkt-Notation (IPv4)
  - Domainname
- Port: Optionale Angabe einer Portnummer

Beispiele

1. `de.wikipedia.org`
2. `user@example.com:8080`
3. `192.0.2.16:80`
4. `[2001:db8::7]`

Path (Pfad)

- Identifiziert abzufragende Ressource (mgl. nur teilweise)
- Ist oftmals hierarchisch organisiert (wie Dateinamen und Verzeichnisse)
- Konkretes Format hängt vom Schema ab

Beispiele

```
https://www.htw-berlin.de/studieninteressierte/
ldap://[2001:db8::7]/c=GB?objectClass?one
mailto:John.Doe@example.com
tel:+1-816-555-1212
```

Query (Abfrage)

- Spezifiziert Ressourcen genau, wenn Pfadangabe alleine nicht reicht
- Syntax ist nicht genau spezifiziert, bei HTTP(s) sind es aber oft Schlüssel-Werte-Paare

Beispiele

```
https://htw.de/index.php?tag=nw&order=newest#top
ldap://[2001:db8::7]/c=GB?objectClass?one
```

Fragment

- Positionsangabe innerhalb der Ressource
- Beispielsweise Anker bei HTML-Dokumenten

Beispiele

```
https://htw.de/index.php?tag=nw&order=newest#top
```

Outline

Spezifikationen

- HTTP/1.0: RFC 1945
- HTTP/1.1: RFC 2616
- HTTP/2: RFC 7540
- HTTP/3: HTTP Draft basierend auf QUIC Draft