Rechnernetze, Übung 9

Henry Haustein

Aufgabe 1

(a) Interpretation:

- *www*: hostname
- *inf*: subdomain
- *tu-dresden*: domain
- *de*: toplevel-domain

(b) Client sendet Namen, zu dem er die IP wissen will. DNS-Server organisiert die IP, indem er selbst in seiner Tabelle nachschaut, oder, wenn es keinen passenden Eintrag gibt, fragt er einen übergeordneten DNS-Server. Hat der DNS-Server die IP beschafft, sendet er sie an den Client zurück.

(c) `dig www.example.com A` liefert

```
; <<>> DiG 9.10.6 <<>> www.example.com A
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44774
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.example.com.   IN  A

;; ANSWER SECTION:
www.example.com.  2197  IN  A 93.184.216.34

;; Query time: 62 msec
;; SERVER: 208.67.222.222#53(208.67.222.222)
;; WHEN: Thu Jun 24 13:23:31 CEST 2021
;; MSG SIZE  rcvd: 60
```

und damit die IP-Adresse 93.184.216.34.

`dig tu-dresden.de MX` liefert

```
; <<>> DiG 9.10.6 <<>> tu-dresden.de MX
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26730
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1
```

```
6
7  ;; OPT PSEUDOSECTION:
8  ; EDNS: version: 0, flags:; udp: 4096
9  ;; QUESTION SECTION:
10 ;tu-dresden.de.      IN   MX
11
12 ;; ANSWER SECTION:
13 tu-dresden.de.     2932   IN   MX   20 mailin5.zih.tu-dresden.de.
14 tu-dresden.de.     2932   IN   MX   20 mailin6.zih.tu-dresden.de.
15
16 ;; Query time: 62 msec
17 ;; SERVER: 208.67.222.222#53(208.67.222.222)
18 ;; WHEN: Thu Jun 24 13:24:39 CEST 2021
19 ;; MSG SIZE  rcvd: 94
```

und damit die Adressen mailin5.zih.tu-dresden.de und mailin6.zih.tu-dresden.de.

`dig tu-dresden.de` liefert

```
1  ; <<>> DiG 9.10.6 <<>> tu-dresden.de
2  ;; global options: +cmd
3  ;; Got answer:
4  ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41062
5  ;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
6
7  ;; OPT PSEUDOSECTION:
8  ; EDNS: version: 0, flags:; udp: 4096
9  ;; QUESTION SECTION:
10 ;tu-dresden.de.      IN   A
11
12 ;; ANSWER SECTION:
13 tu-dresden.de.     600 IN   A 141.76.39.140
14
15 ;; Query time: 62 msec
16 ;; SERVER: 208.67.222.222#53(208.67.222.222)
17 ;; WHEN: Thu Jun 24 13:26:17 CEST 2021
18 ;; MSG SIZE  rcvd: 58
```

und `dig www.tu-dresden.de` liefert

```
1  ; <<>> DiG 9.10.6 <<>> www.tu-dresden.de
2  ;; global options: +cmd
3  ;; Got answer:
4  ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61835
5  ;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1
6
7  ;; OPT PSEUDOSECTION:
8  ; EDNS: version: 0, flags:; udp: 4096
9  ;; QUESTION SECTION:
10 ;www.tu-dresden.de.    IN   A
11
12 ;; ANSWER SECTION:
13 www.tu-dresden.de.  4450   IN   CNAME tucms.wcms.tu-dresden.de.
```

```
14 tucms.wcms.tu-dresden.de. 600 IN  A 141.76.39.140
15
16 ;; Query time: 62 msec
17 ;; SERVER: 208.67.222.222#53(208.67.222.222)
18 ;; WHEN: Thu Jun 24 13:27:19 CEST 2021
19 ;; MSG SIZE  rcvd: 87
```

Ein CNAME Resource Record (CNAME RR) ist im Domain Name System (DNS) dazu vorgesehen, einer Domäne einen weiteren Namen zuzuordnen. Die Abkürzung "CNAME"" steht für *canonical name* (engl. *canonical* = anerkannt) und bezeichnet daher den primären, quasi echten Namen. (`https://de.wikipedia.org/wiki/CNAME_Resource_Record`).

`dig _ldap._tcp.google.com SRV` liefert

```
1  ; <<>> DiG 9.10.6 <<>> _ldap._tcp.google.com SRV
2  ;; global options: +cmd
3  ;; Got answer:
4  ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53873
5  ;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
6
7  ;; OPT PSEUDOSECTION:
8  ; EDNS: version: 0, flags:; udp: 4096
9  ;; QUESTION SECTION:
10 ;_ldap._tcp.google.com.   IN  SRV
11
12 ;; ANSWER SECTION:
13 _ldap._tcp.google.com.  86400 IN  SRV 5 0 389 ldap.google.com.
14
15 ;; Query time: 63 msec
16 ;; SERVER: 208.67.222.222#53(208.67.222.222)
17 ;; WHEN: Thu Jun 24 13:31:41 CEST 2021
18 ;; MSG SIZE  rcvd: 85
```

(d) 1 Tag = 86400 Sekunden. Damit ist die Tabelle

NAME	TTL	CLASS	TYPE	VALUE
jupiter	86400	IN	A	117.186.1.1
jupiter	86400	IN	AAAA	2001:db8:85a3:8d3::1
saturn	86400	IN	A	117.186.1.2
saturn	86400	IN	AAAA	2001:db8:85a3:8d3::2
rn-edu.de.	86400	IN	NS	sonne
sonne	86400	IN	A	117.186.1.3
sonne	86400	IN	AAAA	2001:db8:85a3:8d3::3
_sip._tcp.rn-edu.de.	86400	IN	SRV	0 0 5060 mond.rn-edu.de.
_sip._udp.rn-edu.de.	86400	IN	SRV	0 0 5060 mond.rn-edu.de.
mond	86400	IN	A	117.186.1.4
mond	86400	IN	AAAA	2001:db8:85a3:8d3::4

Aufgabe 2

Bei Base64 wird das ASCII-Alphabet, in dem jedes Zeichen aus 8 Bit besteht (insgesamt $2^8 = 256$ mögliche Zeichen), in ein Alphabet transformiert, in welchem die Zeichen nur 6 Bit benötigen (insgesamt $2^6 = 64$ mögliche Zeichen). Dazu nimmt man 3 Zeichen des ASCII-Alphabetes (24 Bit) und wandelt diese in 4 Zeichen des Base64-Alphabets um (24 Bit). Falls solche Gruppen aus 3 Zeichen nicht voll werden, füllt man diese einfach auf 24 Bit auf. Bei 20.000 Byte Binärdatei gibt es 6.666 volle Gruppen und 1 Gruppe, die mit einem Zeichen aufgefüllt werden muss, also insgesamt 6.667 Gruppen. Jede Gruppe wird durch 4 Zeichen des Base64-Alphabets repräsentiert, sodass wir auf insgesamt 26.668 Byte kommen. Nach je 76 Byte wird ein CR-LF gesendet, das sind insgesamt 351 á 2 Byte. Insgesamt werden nun also 27.370 Byte gesendet.

Moderne Webanwendungen nutzen Base64-Codierung zur Übertragung von Binärdaten zwischen Client und Browser. Base64-Codierung kann dabei clientseitig im Browser (JavaScript) durchgeführt werden.

Aufgabe 3

(a) DNS-Request stellen, Daten vom Webserver holen und darstellen

(b) URL: gaia.cs.umass.edu/cs453/index.html
HTTP-Version: 1.1
persistente Verbindung: ja, wird angefragt
IP ist nicht Bestandteil von HTTP

(c) Statuscode 200 → Dokument wurde gefunden
Datum der Antwort: Tue, 07 Mar 2006 12:39:45GMT
letzte Modifikation: Sat, 10 Dec 2005 18:27:46GMT
Größe in Bytes: 3874
ersten 5 Bytes: ¡!doc
persistente Verbindung wurde aufgebaut

(d) Protokoll: HTTP, Dienst: Programm, welches Funktionen über Schnittstellen bereitstellt

Aufgabe 4

Tabelle

Absender	**Ziel**	**Protokoll**	**Methode/Antwort**	**Inhalt**
Client	DNS-Server	DNS	query	URL vom Webserver (www.heise.de)
DNS-Server	Client	DNS	query response	IP vom Webserver
Client	Webserver	TCP	SYN	
Webserver	Client	TCP	SYN, ACk	
Client	Webserver	TCP	ACK	
Client	Webserver	HTTP	GET /tools	Ziel-URL
Webserver	Client	TCP	ACK	
Webserver	Client	HTTP	200 OK	Dokumentdaten
Client	Webserver	TCP	ACK	
Webserver	Client	TCP	ACK	

Client	Webserver	TCP	TCP Keep-alive
Webserver	Client	TCP	TCP Keep-alive
...	...	...	...
Webserver	Client	TCP	FW
Client	Webserver	TCP	ACK

Aufgabe 5

(a) Schichten:

- Sicherungsschicht: Quelle und Ziel sind der vorangegangene und nachfolgende Knoten, Protokoll z.B. HDLC oder PPP
- Vermittlungsschicht: Quelle und Ziel sind der vorangegangene und nachfolgende Knoten, Protokoll z.B. IP
- Transportschicht: Quelle: 141.76.42.42, Ziel: 141.30.111.111, Protokoll: z.B. TCP

Auf dem Rückweg vertauschen sich Quelle und Ziel.

(b) ?

Aufgabe 6

(a) Manager $\leftarrow$ GET $\leftarrow$ Agent
Manager $\rightarrow$ RESPONSE $\rightarrow$ Agent

(b) Manager $\leftarrow$ GETBULK $\leftarrow$ Agent
Manager $\rightarrow$ RESPONSE $\rightarrow$ Agent

(c) Manager $\rightarrow$ TRAP $\rightarrow$ Agent

(d) Manager $\leftarrow$ SET $\leftarrow$ Agent
Manager $\rightarrow$ RESPONSE $\rightarrow$ Agent