Theorie 2

Fachgruppe Telekommunikationsnetze (TKN)

10. November 2023

Einleitung

Die folgenden Aufgaben werden gemeinsam im Tutorium bearbeitet. In der Veranstaltung Rechnernetze wird die SI-Notation verwendet. Beispiele für Präfixe: m = 10^{-3}, k = 10^{3}, M = 10^{6}, ki = 2^{10}, Mi = 2^{20}. „B“ bezeichnet Bytes, „bit“ Bits.

Übung 1

Beantworten Sie folgende Fragen rund um den TCP/IP Stack:

1. Welche Layer gibt es im TCP/IP Modell?
2. Angenommen Sie öffnen einen Browser und senden eine Web-Anfrage (HTTP) über TCP über IP über Ethernet. In welcher Reihenfolge werden die Header gesendet?
3. Angenommen Sie haben zwei Browser gleichzeitig geöffnet und rufen vom gleichen Server gleichzeitig über HTTP das selbe Dokument ab.
   a) Wie unterscheidet der Server zwischen den beiden Anwendungen?
   b) Was bringt das *Hourglass Modell* zum Ausdruck?

Lösung

1. Anwendungen, Transport, Internet, Netzzugang (Link Layer)
2. Ethernet, IP, TCP (Den HTTP Header würde man nicht als klassischen Header ansehen)
   a) Port auf dem Client-System.
   b) Im Modell des Internets ist es egal, welche Transport-, Applikations- bzw. Sicherungs- und Bitübertragungs-Schicht Implementierungen zum Einsatz kommen. Solange IP unterstützt wird, können verschiedene Protokolle oberhalb und unterhalb genutzt werden.

Übung 2

Die Zuordnung von Domainnamen zu IP-Adressen erfolgt im Internet mit Hilfe des Domain Name Systems (DNS). Beschreiben Sie anhand der folgende Fragen den Aufbau und die Funktionsweise von DNS:

1. Was ist der Unterschied zwischen einem Namen und einer Adresse?
2. Wie sind die Verantwortlichkeiten im Namenssystem aufgeteilt?
3. Wie funktioniert das rekursive bzw. iterative Auflösen von Namen mit DNS?
4. Welche Funktion erfüllen die TTL-Einträge?
5. Überprüfen Sie mittels des Kommandozeilen-Tools `dig` (Linux/OS X) bzw. `nslookup` (Windows), welche DNS-Server für die Domain `tu.berlin` sowie deren Subdomains zuständig sind und welche TTL diese Einträge haben.

Lösung

1. Namen *identifizieren* eine Entität, Adresse beschreiben, wo sie zu finden sind. Im Schichtenmodell kann ein Wert auch beide Rollen einnehmen. Im Sinne der Anwendungsschicht ist `93.184.216.34` die Adresse zum Namen `example.com`, in der Netzzugangsschicht allerdings ist es ein Name der beispielsweise eine MAC-Adresse zugeordnet wird.

2. Letzten Endes sind die root-Nameserver im DNS verantwortlich. Diese delegieren die Verantwortung für Teile des Namensraums an andere Server, deren Adresse sie stehts kennen. Dadurch ist ein root-Nameserver stehts in der Lage eine beliebige Anfrage, zumindest indirekt, also durch Rückfragen bei dei den entsprechend verantwortlichen Nameservern, zu erfüllen.

3. Eine iterative Anfrage beantwortet ein Nameserver mit der Antwort, sofern er diese kennt (`ANSWER SECTION`), oder einem Verweis auf einen Nameserver, der für diesen Bereich des Namensraums verantwortlich ist (`AUTHORITY SECTION` & `ADDITIONAL SECTION`). Durch den hierarchischen Aufbau des DNS ist dies im Zweifel ein root-Nameserver, oder aber eine *näherer* Server, im Sinne der Hierarchie.

   Bei einer rekursiven Anfrage hingegen übernimmt der angefragte Nameserver die Verantwortung, die Anfrage zu erfüllen. Kennt er die Antwort also nicht, stellt er eine entsprechende Anfrage an den nächsten ihm bekannten Nameserver. Dies ist genau derjenige, an den der Client bei einer iterativen Anfrage verwiesen worden wäre.

4. Die TTL-Einträge schränken die Gültigkeit von Einträgen ein. Je höher diese sind, desto weniger Last erfahren die Nameserver, da Anfragen seltener wie-

derholt werden müssen. Kürzere Einträge erlauben die kurzfristige Änderung von Werten, ohne den Dienst zu beeinträchtigen.

5.
```
$ dig tu.berlin

; <<>> DiG 9.18.8 <<>> tu.berlin
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 65290
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 7

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1220
; COOKIE: cb6e8e1460a0a5b0c506c252637e4ee24a8fe03d5d85c7b6 (good)
;; QUESTION SECTION:
;tu.berlin.                  IN     A

;; ANSWER SECTION:
tu.berlin.            28800  IN     A      130.149.8.20

;; AUTHORITY SECTION:
tu.berlin.            28800  IN     NS     ns.tu-berlin.de.
tu.berlin.            28800  IN     NS     dns-3.dfn.de.
tu.berlin.            28800  IN     NS     dns-2.dfn.de.

;; ADDITIONAL SECTION:
dns-3.dfn.de.         60140  IN     A      193.174.75.58
dns-2.dfn.de.         60170  IN     A      193.174.75.54
ns.tu-berlin.de.      28800  IN     A      130.149.7.7
dns-3.dfn.de.         60140  IN     AAAA   2001:638:d:b103::1
dns-2.dfn.de.         60170  IN     AAAA   2001:638:d:b102::1
ns.tu-berlin.de.      28800  IN     AAAA   2001:638:809:7::7

;; Query time: 0 msec
;; SERVER: 130.149.7.7#53(130.149.7.7) (UDP)
;; WHEN: Wed Nov 23 17:48:31 CET 2022
;; MSG SIZE rcvd: 287
```

Die Nameserver `ns.tu-berlin.de` sowie `dns-2.dfn.de` und `dns-3.dfn.de` sind verantwortlich, diese Einträge sind je acht Stunden (28 800 s) gültig.

Übung 3

Diskutieren Sie Vor- und Nachteile der Client-Server-Architektur, auch im Vergleich zu Peer-to-Peer Systemen.

Lösung

Vorteile:

- Lastverteilung auf mehrere Geräte mgl.
- Clients können auf die Dienste des Servers aus Entfernung zugreifen
- Client und Server können separat entworfen werden
- Einfaches, wohlbekanntes Programmierparadigma: Funktionsaufruf
- Auf den Server kann gleichzeitig von mehreren Clients zugegriffen werden
- Einfache Ermöglichung der Modifikation des Servers

Nachteile:

- Server kann Engpass werden
- Signifikante Latenz der Kommunikation möglich
- Ressourcen (z.B. Festplattenplatz) der Clients werden evtl. verschwendet
- Finden des Servers für einen bestimmten Service nicht trivial
- Der Service kann deaktiviert sein wenn ein Server ausfällt

Übung 4

Nutzen Sie das Beispiel des Abrufs von E-Mails durch Clients bei einem Server, um die folgenden Fragen für das Client-Server-Prinzip im Allgemeinen zu beantworten:

1. Muss der betreffende Rechner (Client/Server) immer angeschaltet und ans Netz angeschlossen sein? Was hätte es für Konsequenzen, wenn beide Kommunikationspartner gleichwertig sind, also es keinen expliziten Server bzw. Client gibt?
2. Welche Art von Adresse haben Client und Server? Was hätte es für Konsequenzen, wenn beide Kommunikationspartner gleichwertig sind, also es keinen expliziten Server bzw. Client gibt?
3. Mit wem kommunizieren Clients in der Regel direkt? Mit wem der Server?

Lösung

1. Client: Nein; Server: Ja. Für jede Mail müssten beide Peers immer online sein.
2. Server: öffentliche Adresse; Client: ggf. private Adresse mit NAT.

3. Clients: Server; Server: Clients, Server

Übung 5

Welche Protokolle verwenden Portnummern? Welchem Zweck dienen Ports dabei?

Lösung

In TCP und UDP erlauben Ports die Zuordnung der eingehenden Datenpakete zur richtigen Applikation. Dadurch lassen sich mehrere Datenströme auf einem Host *multiplexen*.

Übung 6

Angenommen Sie möchten ein Programm schreiben, das eine Funktion auf einem entfernten Server per Socket-API aufruft und dabei als Argument ein Paar aus ID und Namen übergibt. Das Programm speichert das Paar intern wie hier dargestellt:

```
1  typedef struct ValueType {
2      int         id;
3      const char* name;
4  } ValueType;
```

1. Kann der dargestellte `struct ValueType` direkt an `send()` übergeben werden? Spielt die Prozessorarchitektur hierbei eine Rolle?
2. Funktionsaufrufe auf entfernten Systemen werden oft als sog. Web Services über HTTP ausgeführt (siehe REST). Wie löst HTTP die Probleme aus der ersten Teilaufgabe?
3. Wie können komplexe Datenstrukturen übertragen werden, bspw. verkettete Listen oder Binärbäume?

Lösung

1. Nein, da sonst einfach der Pointer `name` übertragen werden würde, anstatt der eigentliche Wert. Es muss also ein sog. *marshalling* des Datentyps stattfinden, also ein überführen des Datentyps in eine Folge von Bytes. Dafür muss vorher ein Format definiert werden, was beiden Systemen bekannt ist. Dabei muss insbesondere auf die Endianness von Integern geachtet werden, also in welcher Reihenfolge die einzelnen Bytes einer Zahl übertragen werden.
2. HTTP ist ein textbasiertes Protokoll. Zahlen werden also als Text übetragen. Allerdings gibt es auch hier verschiedene Möglichkeiten Zeichen zu enkodieren (UTF-8, ASCII, etc.). Welcher Zeichensatz verwendet wird, legt der

Server fest, wobei der Client mitteilt, welche Zeichensätze er akzeptiert.

3. Für komplexe Datenstrukturen muss auch ein Marshalling-Format festgelegt werden. Dabei muss die Datenstruktur in eine *flache* Darstellung überführt werden und ggf. Metadaten hinzugefügt werden (z.B. Referenzen auf andere Elemente) damit die Struktur wieder rekonstruiert werden kann im entfernten System.

Übung 7

Beantworten Sie im Kontext vom Hypertext Transfer Protocol (HTTP) folgende Fragen:

1. Welche HTTP Methoden gibt es und was machen diese?
2. Welche HTTP Methoden sind idempotent?
3. Was sind persistente und nicht-persistente Verbindungen?
4. HTTP ist ein zustandsloses Protokoll. Was bedeutet das? Mit welcher Technik können Server z.B. trotzdem den Warenkorb einer Nutzers speichern?

Lösung

1. **Methode** Semantik

   GET Abfragen einer Ressource

   PUT Erstellen oder Ersetzen einer Ressource (unter einer gegebenen URI) mit der mitgeschickten Repräsentation in der Nachricht

   POST Generell: Anhängen/erweitern von Daten einer bestehenden Ressource oder erstellen einer neuen Ressource die noch nicht definiert ist. *Jedoch: Semantik laut RFC 7321 nicht näher festgelegt.*

   DELETE Entfernen einer Ressource

   HEAD Identisch zu GET, außer dass der Server nur mit einem Header antwortet

2. Alle, außer POST.
3. Bei persistenten Verbindungen wird die einer HTTP-Anfrage zugrundeliegende TCP-Verbindung vom Server offen gehalten auch nach dem sie ausgeführt wurde. Danach kann Sie wieder verwendet werden für Folgeanfragen um Ressourcen vom gleichen Server abzufragen (z.B. eingebettete Bilder, Skripte, Stylesheets, etc.). Dadurch spart man sich den zeitaufwändigen Verbindungsaufbau gleich mehrfach.
4. Zustandslos bedeutet in diesem Fall, dass der Server keine Informationen über den Client-State speichert und verwaltet. Stattdessen muss der Client

bei jeder Anfrage bzw. bei jedem Aufruf explizit Informationen zu seinem Status mitübergeben. Um Informationen, wie zum Beispiel einen Warenkorb, zu speichern, kann der Server den Warenkorn in einer Datenbank mit einer eindeutigen ID speichern. Wenn der Client also den ersten Artikel zum Warenkorn hinzufügt, kann der Server eine neue Bestellung in der Datenbank anlegen, und die entsprechende ID dem Client als Antwort zusenden. Möchte der Client nun den nächsten Artikel dem Warenkorb hinzufügen, muss er die entsprechende Warenkorb ID mit übergeben, damit der Server den Artikel der Bestellung in der Datenbank hinzufügen kann. Bei jedem weiteren Aufruf wird vom Client nun immer diese ID als Zusatzinformation mitübergeben, wodurch der Server den Warenkorn bestimmen kann.

Übung 8

Gegeben sei ein RESTful Webservice, der Teilnehmer in einer Veranstaltung verwaltet. Der Service läuft auf dem Rechner mit dem DNS-Namen `api.tkn.tu-berlin.de`. Folgendes Verhalten ist bereits vorgegeben und implementiert, dabei sind `<courseName>` und `<student>` jeweils Platzhalter für die jeweilige Veranstaltung bzw. den Teilnehmer:

Ressource	Methode	Verhalten
`/courses/<coursename>`	`GET`	gibt Liste der Teilnehmer zurück
`/courses/<courseName>`	`DELETE`	Löscht die Veranstaltung
`/courses/<courseName>/<student>`	`GET`	Gibt bestimmten Teilnehmer zurück
`/courses/<courseName>/<student>`	`DELETE`	Löscht Teilnehmer

1. Ist eine der Ressourcen eine Collection-URL und falls ja, geben sie die komplette URL an (inkl. verwendender Zugriffsmethode und Hostnamen)?

2. Der Webservice soll von Ihnen nun um die Funktionalität erweitert werden, die Klausurnote eines Teilnehmers zu ändern. Welche HTTP-Methode muss dazu auf welche URL aufgerufen werden?

3. Was wäre allgemein der Inhalt (Payload) der Anfrage aus Teilaufg. 2.

4. Der Webservice soll von Ihnen nun um die Funktionalität erweitert werden, einen neuen Teilnehmer zur Veranstaltung hinzuzufügen, welcher vorher noch nicht angemeldet war. Welche HTTP-Methode muss dazu auf welche URL aufgerufen werden?

Lösung

1. Collections sind Gruppierungen von Resourcen. Somit ist `/courses/<courseName>` eine Collection URL.

2. PUT auf `/courses/<courseName>/<student>`
3. Eine Repräsentation der kompletten Resource (also des Studenten), inklusive der geänderten Note.
4. POST auf `/courses/courseName`

Übung 9

In dieser Aufgabe versuchen wir nachzuvollziehen wie ein Aufruf von YouTube abläuft und wie das verwendete CDN arbeitet. Rufen Sie dazu ein YouTube Video auf, während Sie den Netzwerktraffic mit Wireshark mitschneiden.

Hinweis: Durch gecachten Inhalt kann es sein, dass sie nicht die volle Anfrage sehen, falls Sie die Website bereits besucht haben. Um dies zu vermeiden starten Sie gegebenenfalls das System vor dem Test neu. Eure Teampartner erhalten unter Umständen andere Ergebnisse, vergleicht diese.

1. Welche DNS Anfragen werden ausgelöst? Welcher der Hosts liefert vermutlich die eigentliche Website aus? Welcher Host stellt die Videodaten bereit?
2. Wieso werden mehr Anfragen als nur `youtube.com` gestellt?
3. Welche CDN Mechanismen werden verwendet?
4. In der Vorlesung wurde besprochen, dass große Datenmengen (wie z.B. Videos) möglichst nah beim Nutzer gespeichert werden um Netzwerkkapazität zu sparen. Versuchen Sie herauszufinden, wo sich der Server befindet, der diese Daten ausliefert. [1] Überprüfen Sie ob dieser Standort plausibel ist, beispielsweise indem Sie die Latenz via `ping` testen.

Lösung

1. Zum Beispiel:
   - `www.youtube.com`
   - `ocsp.pki.goog`
   - `fonts.googleapis.com`
   - `fonts.gstatic.com`
   - `r4---sn-i5heen7z.googlevideo.com`
   - `i.ytimg.com`
   - `i1.ytimg.com`
   - `yt3.ggpht.com`

[1] Bspw. über dieses Online-Tool: `https://www.ip-tracker.org/locator/ip-lookup.php`

- `lh6.googleusercontent.com`
- `i9.ytimg.com`
- `www.youtube.com` (CNAME für `youtube-ui.l.google.com`) liefert inital die Website aus.
- `X.googlevideo.com` liefert das Video (wenn das Video abgespielt wird entsteht viel Netzwerkverkehr mit der entsprechenden IP Adresse).

2. Anfragen an weitere Hosts werden gestellt um Inhalt (Fonts, Werbung, Bilder, Videos) nachzuladen.
3. DNS-basierte Umleitung (verschieden DNS Records für unabhängige Anfragen) und URL Rewriting (`r4---sn-i5heen7z.googlevideo.com`)
4. Laut dem Tracking Tool befindet sich diese (vermutlich) in Kalifornien. Bei den meisten Servern `ping` ergibt sich allerdings eine Latenz von lediglich einigen Millisekunden. Ein Server, der innerhalb von 10ms Antwort kann höchstens 1500km Entfernt sein - wenn die Kommunikation mit Lichtgeschwindigkeit stattfindet und keine Verarbeitungsverzögerung auftritt. Der Server muss also entgegen der Aussage des Tools in einem Rechenzentrum in Europa stehen.