Freie Universität Berlin

Prof. Dr. Matthias Wählisch, Marcin Nawrocki, M.Sc.

Telematics Project overview, WS21/22

Projekt 1

Tutorium 01
Materialien: Latex, Python

David Ly & Jonny Lam & Thore Brehmer

4. Dezember 2021

---

1 Components

In unsere Abgabe benutzen wir folgende Komponenten:

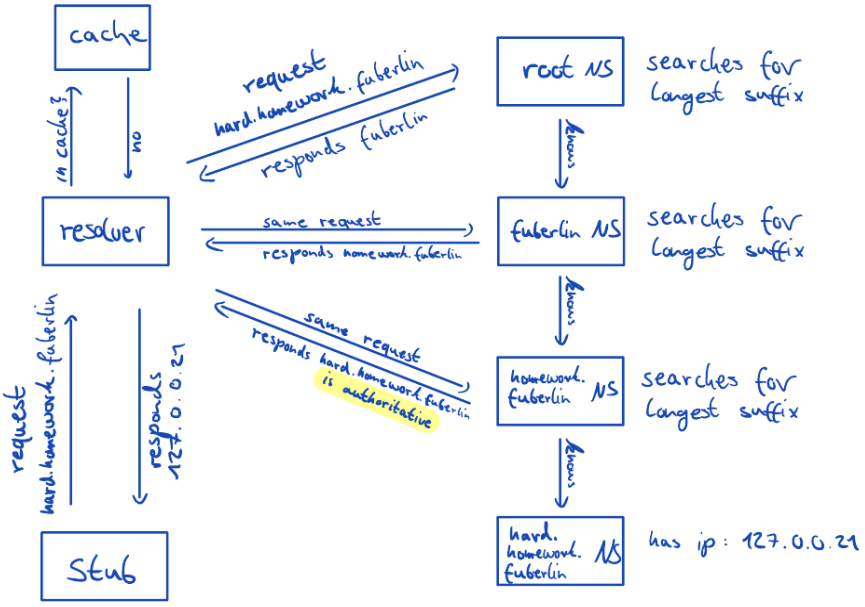
- **stub_resolver**
  - Unser Stub Resolver (Client) fragt für Domains die zugehörige IP-Adresse beim Rekursiven Revolver an. Die Antwort vom Rekursiven Resolver wird vom Stub Resolver interpretiert. (Erfolgreich, Fehlgeschlagen usw.)
- **recursive_revolser**
  - Der Rekursiven Resolver empfängt Anfragen von Stub Resolvern und Antworten von Name Servers. Anfragen von Stub Resolvern werden versucht zu bearbeiten, indem sie weiter an den Name Server(Root) gesendet werden. Darauf werden die Name Server antworten. Diese Antwort muss entweder weiter bearbeitet werden (neue Anfrage an einen Naming Server senden) oder sie ist abgeschlossen (authoritative) und kann an den Stub Resolver zurück gesendet werden.
- **nameserver**
  - Die Name Server enthalten für ihren Domain Namen eine IP-Adresse und kennen den Domain Namen und die IP-Adressen von unterlegenden Name Servern. Wenn sie eine Anfrage erhalten antworten sie mit dem Name Server den sie kennen, welcher das längste Suffix der Anfrage gleicht.
- **loghanlder**
  - Besitzt eine Methode zum erstellen von log Dateien.
- **cache**
  - Besitzt einen dictionary in welchen man DNS Anfragen hinzufügen kann und das längste Präfix aus einer Anfrage ausgeben kann, welches sich bereits im dictionary befindet. Außerdem startet es beim erstellen des Speichers einen Thread, welcher beim überschreiten der ttl einer DNS Anfrage, diese löscht. Wir im Rerkusiven Resolver benutzt.
- **dns.py**

  - Besitzt nützliche Funktionen für das senden und empfangen von json Dateien über udp, sowie für das erstellen von DNS Anfragen und DNS Antworten.

- **server_table.json**
  - Enthält die in der Aufgabe gestellten Name Server
- **main.py**
- - Das main.py Skript lädt, aus der server_table.json Datei, die Name Server und startet diese. Sie startet auch den Rekursiven Resolver und schickt mehrere Test DNS Anfragen, über den Stub Resolver, an den Rekursiven Resolver.
- **dns.flags**
  - Wir benutzen die uns vorgegebenen und die im Tutorium gezeigten Flags. Die Flags dns.count.auth_rr und dns.count.add_rr, vom Tutorium, haben wir jedoch ausgelassen. Außerdem benutzen wir die Flag dns.id (Transaction ID), womit sich unser Rekursiver Resolver merken kann, welche Anfrage von welchem Stub Resolver kam. (Falls mehrere Stub Resolver gleichzeitig anfragen würden)

2 Ablauf Visualisiert

Aus Darstellungsgründen werden Anfragen und Antworten als Domain Name und IP-Adresse dargestellt. Eigentlich sind dies jedoch DNS-Anfragen mit Flags. Außerdem kennen sich Name Server nur mit der Tiefe 1. (Also kennt, in diesem Beispiel, der root Name Server den homework.fuberlin Name Server nicht.)



3 Milestones

- **Implemented:** Your stub resolver is able to (directly) request an A record from the authorative server.
- **Implemented:** Your recursive resolver is able to discover the authoritative server of a name, and resolve the A record for this name.
- **Implemented:** Your stub resolver is able to resolve any name in the list via the recursive resolver and profits from faster replies in the case of cache hits at the recursive resolver.
- **Not Implemented:** Your DNS implementation is used by an application (see HTTP proxy below).

4 Tests: main.py Ausgabe

```
ALL SERVERS STARTED
RESOLVER: STARTED

Test 0 Milestone 1 stub_resolver
 This dns request is send from the stub_resolver directly to the name_server(root)
 stub_resolver asks for:fuberlin
 stub_resolver interprets resp as:127.0.0.26
 Elapsed time:0.10104513168334961

TEST 1 Milestone 2 rekursive_resolver
 This dns request and the following are requested by the stub_resolver and solved with the
     rekursive_resolver
 This dns request should be successful and take around ~ 0.3sec as the rerkusive_resolver
    has to send 3 requests
 stub_resolver asks for:easy.homework.fuberlin
 stub_resolver interprets resp as:127.0.0.20
 Elapsed time:0.30408525466918945

TEST 2
 This dns request should be successful and take around ~ 0.1sec as the rerkusive_resolver
    has to send 1 requests
 stub_resolver asks for:telematik
 stub_resolver interprets resp as:127.0.0.12
 Elapsed time:0.1020052433013916

TEST 3 Milestone 3 cache
 This dns request should be successful and take around ~ 0.2sec as the rerkusive_resolver
    has to send 3 requests, but one was already cached
 stub_resolver asks for:shop.router.telematik
 stub_resolver interprets resp as:127.0.0.18
 Elapsed time:0.20456624031066895

TEST 4 Milestone 3 cache
 This dns request should be successful and take around ~ 0.0sec as the hole request was
    already cached
 stub_resolver asks for:shop.router.telematik
 stub_resolver interprets resp as:127.0.0.18
 Elapsed time:0.0007107257843017578

TEST 5
 This dns request should fail because stub_resolver requests a not supported function
 stub_resolver asks for:easy.homework.fuberlin with dns_qry_type=3
 stub_resolver interprets resp as:Not Implemented
 Elapsed time:0.10173559188842773

TEST 6
 This dns request should fail because the nameserver does not find this domain
 stub_resolver asks for:thisdomain.does.not.exist.fuberlin
 stub_resolver interprets resp as:Non-Existent Domain
 Elapsed time:0.10197663307189941

TEST 7
 Sleep 4 seconds
 This dns request should be successful and take around ~ 0.3sec as the resolver has to
    send 3 requests and the previous accessed cache was removed (ttl)
 stub_resolver asks for:shop.router.telematik
 stub_resolver interprets resp as:127.0.0.18
 Elapsed time:0.3042137622833252

Done. Press Crtl+c to exit
```