



## 4. Übung zur Internet-Praktikum

### Aufgabe 1: (100 Punkte) Vereinfachter DNS-Client, Teil 2:

In dieser Aufgabe geht es darum, den DNS-Client von Übungsblatt 3 so zu erweitern, dass DNS-Anfragen an einen DNS-Server gesendet und dessen Antwort ausgewertet werden können.

Das Antwortpaket eines DNS-Servers kann beliebig viele Antworten in den Bereichen **ANSWERS**, **AUTHORITIES** und **ADDITIONALS** haben. Die konkrete Anzahl trägt der Server in die entsprechenden Felder im Header ein. Eine einzelne Antwort wird durch einen Resource Record (RR) kodiert:

<b>multiple octets</b>	<b>16</b>	<b>16</b>	<b>32</b>	<b>16</b>	<b>multiple octets</b>
<b>Name</b>	<b>Type</b>	<b>Class</b>	<b>TTL</b>	<b>Rdlength</b>	<b>Rdata</b>

Genauereres findet sich in **RFC 1035, Abschnitt 4.1.3**.

- (a) Erweitere das DNS-Client-Programm vom letzten Aufgabenblatt um die Fähigkeit, DNS-Anfragen an einen DNS-Server zu senden und dessen Antwortpaket empfangen zu können. Dazu müssen die erstellten DNS-Anfragen per UDP-Socket an Port 53 eines DNS-Servers gesendet und das Antwortpaket über den selben Socket empfangen werden. Das Antwortpaket des Servers ist in eine Datei zu schreiben.

Auf den Praktikums-Rechnern kann man unter `/home/inetprak/DNS/dns\_request.pm` ein Perl-Modul finden, mit dem ein korrekter DNS-Request generiert werden kann. Dort findet sich auch ein Beispielprogramm (`dns.pl`).

Abzugeben sind:

- Der Quelltext Deines Programmes
  - Eine Datei mit dem Antwortpaket des DNS-Servers, wenn Du Dein Programm für `www.lrz-muenchen.de` aufrufst. Als Nameserver kannst Du z. B. `131.159.14.1` oder `129.187.10.25` verwenden.
- (b) Für diesen Aufgabenteil soll der DNS-Client so erweitert werden, dass die Server-Antwort ausgegeben werden kann. Dein Programm muss für diesen Aufgabenteil nur Antworten vom Type A und Class IN unterstützen. Andere Fälle brauchen nicht betrachtet zu werden. Es reicht außerdem, nur den ersten Resource Record im **ANSWERS**-Bereich zu bearbeiten; **AUTHORITIES** und **ADDITIONALS** können ignoriert werden.
- Dein Programm soll zunächst den kompletten Header des Antwortpaketes auswerten und die einzelnen Felder ausgeben. Hierbei muss darauf geachtet werden, dass man bei eventuellen Fehlercodes vom Server die weitere Bearbeitung mit einer passenden Fehlermeldung beendet.
- Anschließend ist der **QUESTIONS**- und **ANSWERS**-Bereich auszugeben, wobei man von genau einer Question und einer Answer ausgehen kann. Zur Kontrolle kannst Du z. B. die Ausgabe des Programms `dig` verwenden.

Ein Problem ergibt sich noch daraus, dass DNS bei den Antworten einen Kompressionsalgorithmus verwendet, um bei DNS-Namen Platz zu sparen. In diesem Aufgabenteil musst Du noch nicht verstehen, wie diese Kompression funktioniert. Gehe einfach davon aus, dass das Name-Feld im ANSWERS-Bereich durch die Kompression genau 2 Bytes lang wird und gib anstatt des richtigen DNS-Namens einfach die Zeichenkette `compressed` aus.

Abzugeben sind:

- Der Quelltext Deines Programmes
- Die Ausgabe Deines Programmes, wenn Du es für `www.lrz-muenchen.de` aufrufst. Als Name-server kannst Du z. B. `131.159.14.1` oder `129.187.10.25` verwenden.

(c) Erweitere Dein Programm, so dass es auch mit komprimierten DNS-Namen bei den Antworten zurechtkommt. Eine genaue Erklärung des Algorithmus findest Du in **RFC 1035, Abschnitt 4.1.4**.

Abzugeben sind:

- Der Quelltext Deines Programmes
- Je eine Datei mit der gesendeten Anfrage und der empfangenen Antwort.
- Die Ausgabe Deines Programmes, wenn Du es für `www.lrz-muenchen.de` aufrufst.

(d) Dein Programm ist nun so zu vervollständigen, dass es alle Antworten (ANSWERS, AUTHORITIES und ADDITIONALS) ausgibt. Dabei soll es bei der Ausgabe zusätzlich zu Type A auch Type CNAME und NS unterstützen. Beim QUESTIONS-Bereich kann weiterhin von genau einer Question ausgegangen werden.

Abzugeben sind:

- Der Quelltext Deines Programmes
- Die Ausgabe Deines Programmes für `www.heise.de` und `www.in.tum.de`.

(e) Beschreibe den Unterschied zwischen rekursiven und iterativen DNS-Anfragen.

Wo werden in der Praxis typischerweise rekursive, wo iterative Anfragen benutzt?

Abzugeben sind:

- Eine Datei mit Deiner Beschreibung (`.txt`, `.dvi`, `.ps` oder `.pdf`)

(f) Erweitere Dein Programm um die Möglichkeit von iterativen Anfragen und die Fähigkeit, mit entsprechenden Antworten umzugehen. Sollte also in einem Antwortpaket nicht die angefragte Information enthalten sein, so muss Dein Programm selbständig beim nächsten Server in der Kette weiterfragen. Gib alle Antwortpakete, die Du auf dem Weg zum Ziel erhältst, in Dateien aus.

Abzugeben sind:

- Der Quelltext Deines Programmes
- Die Ausgabe Deines Programmes, die eine iterative Auflösung von `www.heise.de` zeigt.

**Details zur Abgabe der Aufgaben: siehe FAQ (unterhalb <http://www.net.in.tum.de/teaching/SS05/inetprak/>).**

**Abgabedatum: 10.5.2005 23:59h s.t.**