TECHNISCHE UNIVERSITÄT BERLIN
Fakultät IV – Elektrotechnik und Informatik
Fachgebiet Intelligente Netze und Management verteilter Systeme
Prof. Anja Feldmann, Ph. D.
Harald Schiöberg, Bernhard Ager, Vlad Manilici

3rd Assignement Protocol Design WS 08/09

Question 1: (20 points) Latency and transmission delay

(a) Estimate, based on the speed of light ($c = 300.000\,\text{km/s}$), the theoretical minimum that it takes to transmit data from here to

- Paris
- Washington, D.C, using a transatlantic cable
- Washington, D.C, using a geo-stationary satellite

Rough estimates of the distances are fine.

(b) Calculate the transmission delay for transmitting 400 and 1500 Bytes via a 56kbit Modem, DSL, 10Mbit/s Ethernet and Gigabit Ethernet.

Which number dominates the delay on the different media and distances?

Submit: Your explanations and calculations as file. (ps/eps/txt/html)

Question 2: (80 points) Simplified DNS-Client (part one)

This question is about implementing a simplified DNS client. The DNS protocol is specified in the **RFCs 1034 and 1035**. To solve this question it is not necessary to fully read the two RFCs. It should be sufficient to recap the basics of DNS, e.g. using the textbook (Kurose/Ross, see web-site). For the implementation details we will point you to the significant parts of the RFC in the respective questions. The RFCs are available online, e.g., via `http://www.rfc-editor.org/`.

To get started you should have a look at the program `dig`. Your tool should in the end output the same data as `dig` (but you don't have to imitate the output format!)

A DNS packet always has the following structure (**RFC 1035, Sec. 4.1**):

Header
Questions
Answers
Authoritys
Additionals

(a) Write a program that builds a DNS header. The length of a DNS header is always 12 Bytes and has the following format:

16	1	4	1	1	1	1	3	4	16	16	16	16
ID	**QR**	**Opcode**	**AA**	**TC**	**RD**	**RA**	**Z**	**Rcode**	**Qdcount**	**Ancount**	**Nscount**	**Arcount**

Detailed explanation of the fields can be found in **RFC 1035, Sec. 4.1.1**

Your program should contain a dedicated function that fills in the fields of the header according to the passed parameters. You can check the correctness of your program by calling it with different parameters and reviewing the output file with a hex editor. On the course machines the programm `hexdump` can be used.

Submit:

- The program source code
- The resulting output file, when calling your function with these parameters: `ID` = 1000, `QR` = 1, `AA` = 1, `RD` = 1, `Qdcount` = 1, all other parameters = 0.

(b) Now extend your program to construct a DNS query packet and output it to a file. In addition to the function from part (a), you will need a new function that fills the fields of the query according to its parameters. **Attention:** For the header you should not just use the header fields from part (a) but your program should choose sensible values. You should pay attention to the fact, that DNS names in the `Qname` field are coded in a special format. See **RFC 1035, Sec 2.3.1 and Sec 3.1**. The DNS protocol knows a plethora of different queries. For this assignment it is sufficient to support `Qtype A` and `Q`class IN. Explanations on the different types and classes can be found in **RFC 1035, Sec 3.2.2-5**.

Submit:

- The source code of your program
- The resulting output file, when you call your programm with `www.heise.de`.

(c) Describe the difference between recursive and iterative DNS queries.

When are recursive or iterative queries used in practice?

Submit: Your explanations as a file.

For details on the submission: see online FAQ.

Submission Deadline: Wed, 12. Nov. 2008, 23:59h

—End of questions for 3rd assignement—

Preview on Assignment 4

*For your reference here comes the second part of the DNS assignment, which **will be handed out next week**. It is here that you can better estimate what comes next, and write your code in a way that it can be easily extended. If you want to, you can even start with these questions, since next weeks assignment is a bit lenghty, but do not submit any solutions to these questions yet.*

- Now you should send the DNS request to a DNS server and parse the corresponding reply packet. This may contain arbitrary many answers in sections `ANSWERS`, `AUTHORITYS` and `ADDITIONALS`. The exact number is inserted in the corresponding header fields by the server. A single answer is encoded in a Rescource Record (RR):

multiple octets	16	16	32	16	multiple octets
Name	**Type**	**Class**	**TTL**	**Rdlength**	**Rdata**

Details can be found in **RFC 1035, Sec. 4.1.3**. For now your program needs only support for answers with type `A` and class `IN`, other cases can be ignored. It is furthermore still sufficient to process only the first RR in the `ANSWERS` section; `AUTHORITIES` and `ADDITIONALS` can be ignored.

Your program must first parse the complete header of the answer packet, and print the individual fields to standard out. Errorcodes sent by the server must be treated properly and the program must terminate with a suitable error message.

Subsequently your program must output the `QUESTIONS` and `ANSWERS` sections, where you can assume exactly one question and one answer. The output format of your program should be roughly similar to the `dig` command.

An additional challenge is, that DNS uses a compression algorithm for names. For this part of the question you need not understand the compression algorithm. Just assume, that the `Name` field in the `ANSWERS` section is exactly two bytes long due to compression, and output the string `compressed` instead of the real DNS name.

Submit:

– The source code of your program

  - The output of your program, if you query for `www.heise.de`. You can use `130.149.220.253` or `130.149.2.12` as nameserver.

- Extend your program to cope with compressed DNS names. A detailed explanation of the algorithm can be found in **RFC 1035, Sec. 4.1.4**.

  Submit:

  - The source code of your program
  - The output of your program, if you query for `www.heise.de`.

- Add support for `AUTHORITYS` and `ADDITIONALS`. Add these sections to your program's output. Add support for the types `CNAME` and `NS`. You can still assume exactly one RR in the `QUESTIONS` section. You output format should further on be roughly similar to `dig`.

  Submit:

  - The source code of your program
  - The output of your program for `www.heise.de` and `www.google.de`.

- Extend your program with the option to perform iterative queries, and to deal with the related answers. Output all answer packets. The query should be done automatically, i.e. you should need to run your program only once to do a full iterative query.

  Submit:

  - The source code of your program
  - The output of your program, while doing an iterative resolution of `www.heise.de`.