

Introduction to Web Science

Assignment 3

Prof. Dr. Steffen Staab

staab@uni-koblenz.de

René Pickhardt

rpickhardt@uni-koblenz.de

Korok Sengupta

koroksengupta@uni-koblenz.de

Institute of Web Science and Technologies

Department of Computer Science

University of Luxembourg

Submission until: November 16, 2016, 10:00 a.m.

Tutorial on: November 18, 2016, 12:00 p.m.

The main objective of this assignment is for you understand different concepts that are associated with the "Web". In this assignment we cover two topics: 1) DNS & 2) Internet.

These tasks are not always specific to "Introduction to Web Science". For all the assignment questions that require you to write a code, make sure to include the code in the answer sheet, along with a separate python file. Where screen shots are required, please add them in the answers directly and not as separate files.

Delta Group Members

Oana Dumitrasc

odumitrasc@uni-koblenz.de

Alisa Becker

alisabecker@uni-koblenz.de

Omar Aly

oaly@uni-koblenz.de

1 DIG Deeper (5 Points)

Assignment 1 started with you googling certain basic tools and one of them was "dig".

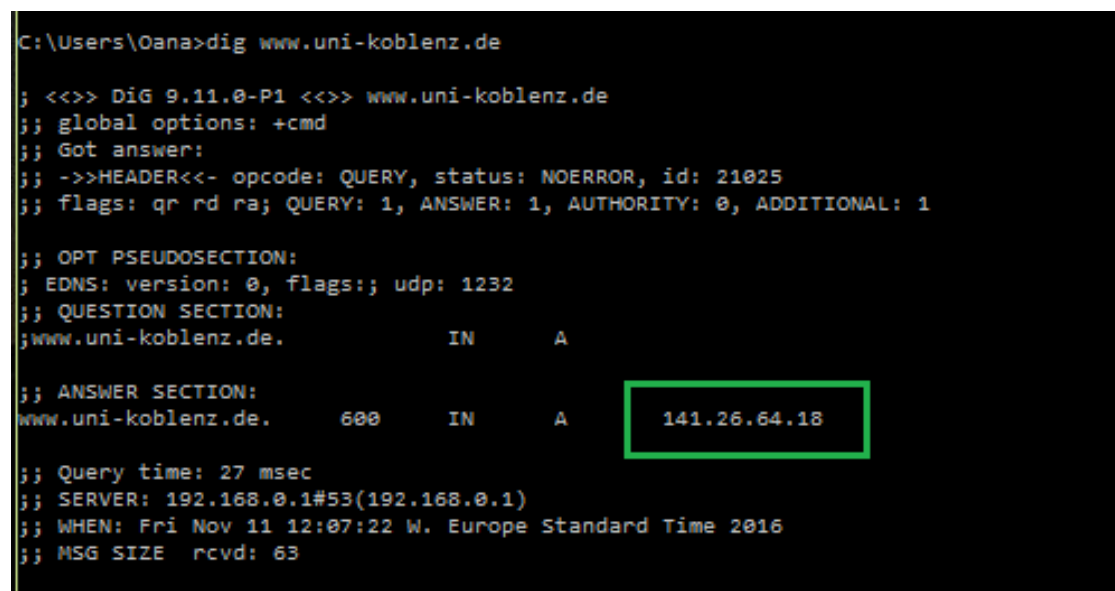
1. Now using that dig command, find the IP address of www.uni-koblenz-landau.de
2. In the result, you will find "SOA". What is SOA?
3. Copy the SOA record that you find in your answer sheet and explain each of the components of SOA with regards to your find. Merely integrating answers from the internet won't fetch you points.

Try the experiment once from University network and once from Home network and see if you can find any differences and if so, clarify why.

Answers:

From Home Network:

1. Figure 1 - IP address of www.uni-koblenz-landau.de from the Home Network



```
C:\Users\Oana>dig www.uni-koblenz.de

; <<>> DiG 9.11.0-P1 <<>> www.uni-koblenz.de
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21025
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;www.uni-koblenz.de.          IN      A

;; ANSWER SECTION:
www.uni-koblenz.de.        600     IN      A      141.26.64.18

;; Query time: 27 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: Fri Nov 11 12:07:22 W. Europe Standard Time 2016
;; MSG SIZE rcvd: 63
```

Figure 1: IP address from Home Network

2. SOA is an abbreviation for Start of Authority. The SOA record is the most important part of the Zone file and this file can only contain one such record. With the help of this set of data, resources are provided for the Domain Name System and used to validate domains on the Internet. Through this record the Domain Administrator gives information about domain: how often it is updated, when it was last updated, when to check back for more info, what is the admin's email address and so on.

```
C:\Users\Oana>dig SOA www.uni-koblenz.de

; <<>> DiG 9.11.0-P1 <<>> SOA www.uni-koblenz.de
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30428
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;www.uni-koblenz.de.                IN      SOA

;; AUTHORITY SECTION:
uni-koblenz.de.      10800  IN      SOA      ns1.uni-koblenz.de. root.ns1.uni-koblenz.de. 2016102831 14400 900 172800 28800

;; Query time: 23 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: Fri Nov 11 12:01:25 W. Europe Standard Time 2016
;; MSG SIZE rcvd: 92
```

Figure 2: dig SOA - www.uni-koblenz-landau.de - Home Network

3. SOA record - explanation for each of the components:

```
C:\Users\Oana>dig soa +multiline www.uni-koblenz.de

; <<>> DiG 9.11.0-P1 <<>> soa +multiline www.uni-koblenz.de
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60916
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;www.uni-koblenz.de.                IN      SOA

;; AUTHORITY SECTION:
uni-koblenz.de.      7605  IN      SOA      ns1.uni-koblenz.de. root.ns1.uni-koblenz.de. (
                        2016102831 ; serial
                        14400      ; refresh (4 hours)
                        900        ; retry (15 minutes)
                        172800     ; expire (2 days)
                        28800      ; minimum (8 hours)
                        )

;; Query time: 19 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: Fri Nov 11 12:54:40 W. Europe Standard Time 2016
;; MSG SIZE rcvd: 92
```

Figure 3: dig SOA +multiline www.uni-koblenz-landau.de - Home Network

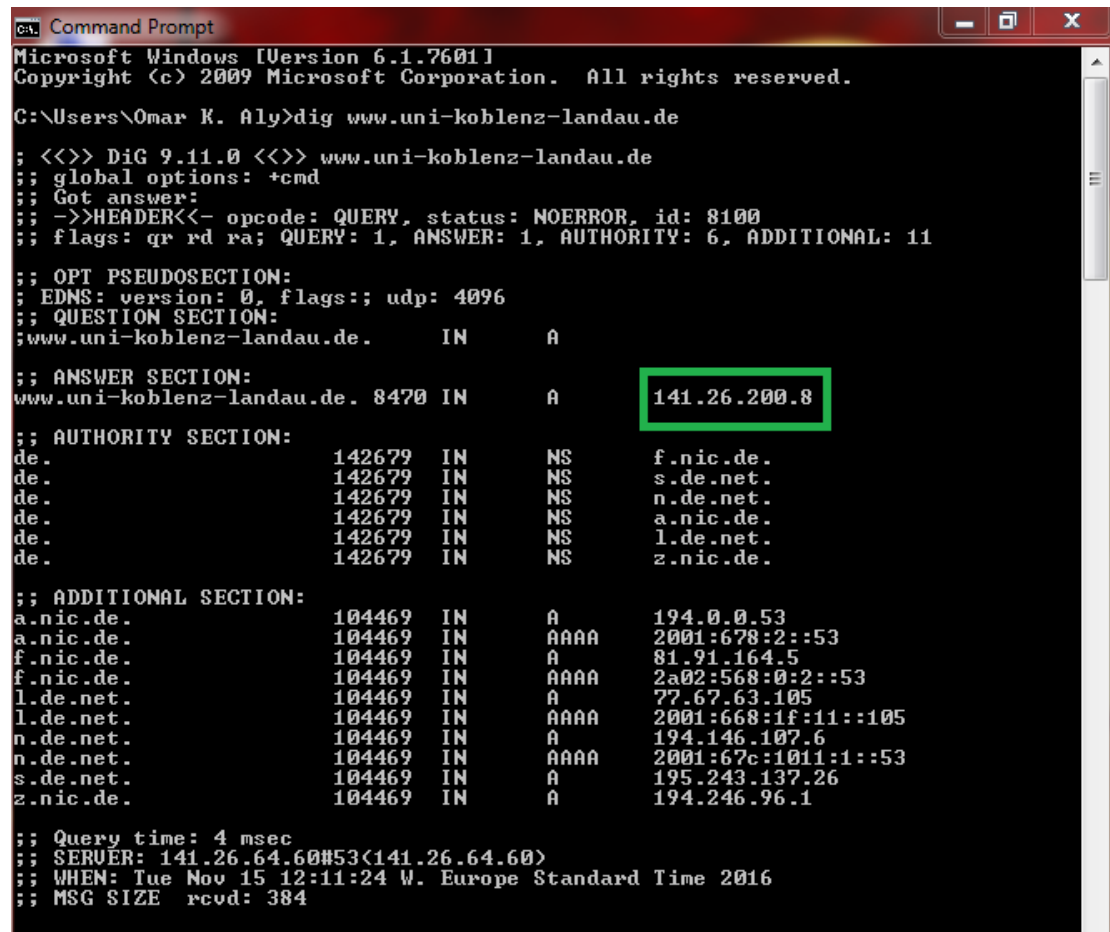
```
1: ;; AUTHORITY SECTION:
2: uni-koblenz.de.      7605 IN SOA ns1.uni-koblenz.de. root.ns1.uni-koblenz.de. (
3:                      2016102831 ; serial
4:                      14400      ; refresh (4 hours)
5:                      900        ; retry (15 minutes)
6:                      172800     ; expire (2 days)
7:                      28800      ; minimum (8 hours)
8:                      )
```

- Name - uni-koblenz.de - is the main name of this zone
- TTL – 7605 – TTL defines the duration in seconds that the record may be cached by client side programs.
- Class – IN – The class shows the type of record. IN points to Internet.
- Name Server - ns1.uni-koblenz.de - the primary name server for the domain.
- Email address – root.ns1.uni-koblenz.de. – This is the email of the domain name administrator. The email will be sent to root@ns1.uni-koblenz.de.
- Serial number – 2016102831 – This is a sort of a revision numbering system to show the changes made to the DNS Zone. This number has to increment whenever any change is made to the Zone file. The standard convention is to use the date of update YYYYMMDDnn, where nn is a revision number in case more than one updates are done in a day. For our example revision number is 31.
- Refresh – 14400 – After this is time when the slave DNS server will refresh from the master (primary server). It checks if the serial number for the zone has increased (so it knows to request a new copy of the data for the zone).
- Retry – 900 – If the attempt to contact the master server and failed to contact it because it was down. The Retry value will tell it when to get back.
- Expiry – 172800 – This is the time that a slave server will keep a cached zone file as valid, if it can't contact the primary server. It keeps domain information available for the specified time.
- Minimum – 28800 – This is the default time that the slave servers should cache the Zone file. The benefit of keeping this value high, is that your website speeds increase drastically as a result of reduced look-ups.

Answers:

From University Network:

1. IP Address from university network is: 141.26.200.8
2. Same description as from home network



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Omar K. Aly>dig www.uni-koblenz-landau.de

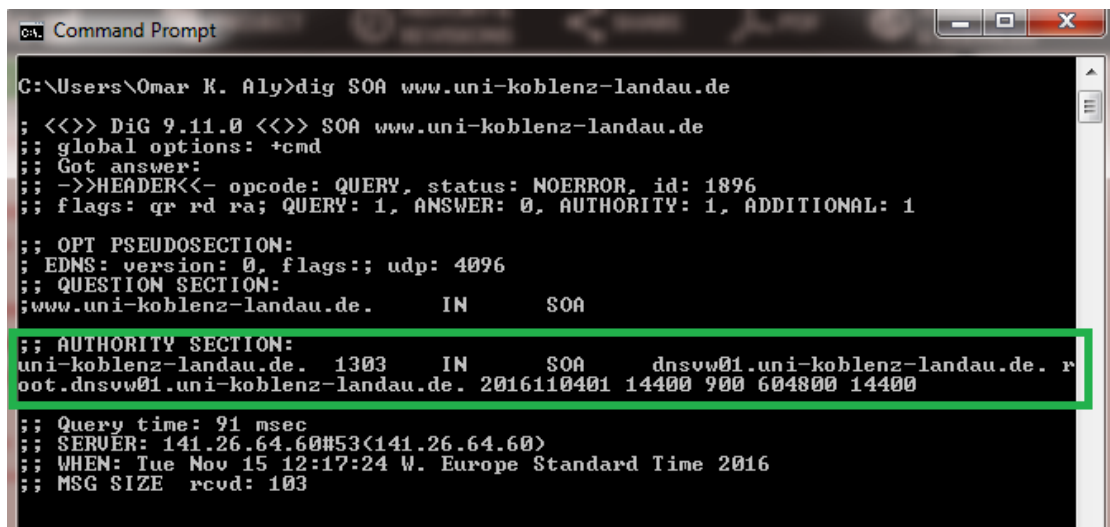
;; <<>> DiG 9.11.0 <<>> www.uni-koblenz-landau.de
;; global options: +cmd
;; Got answer:
;; ->>HEADER<- opcode: QUERY, status: NOERROR, id: 8100
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 6, ADDITIONAL: 11

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:;, udp: 4096
;; QUESTION SECTION:
;www.uni-koblenz-landau.de.      IN      A
;; ANSWER SECTION:
www.uni-koblenz-landau.de. 8470 IN    A      141.26.200.8
;; AUTHORITY SECTION:
de.      142679 IN      NS      f.nic.de.
de.      142679 IN      NS      s.de.net.
de.      142679 IN      NS      n.de.net.
de.      142679 IN      NS      a.nic.de.
de.      142679 IN      NS      l.de.net.
de.      142679 IN      NS      z.nic.de.

;; ADDITIONAL SECTION:
a.nic.de. 104469 IN      A      194.0.0.53
a.nic.de. 104469 IN      AAAA   2001:678:2::53
f.nic.de. 104469 IN      A      81.91.164.5
f.nic.de. 104469 IN      AAAA   2a02:568:0:2::53
l.de.net. 104469 IN      A      77.67.63.105
l.de.net. 104469 IN      AAAA   2001:668:1f:11::105
n.de.net. 104469 IN      A      194.146.107.6
n.de.net. 104469 IN      AAAA   2001:67c:1011:1::53
s.de.net. 104469 IN      A      195.243.137.26
z.nic.de. 104469 IN      A      194.246.96.1

;; Query time: 4 msec
;; SERVER: 141.26.64.60#53(141.26.64.60)
;; WHEN: Tue Nov 15 12:11:24 W. Europe Standard Time 2016
;; MSG SIZE rcvd: 384
```

Figure 4: IP Address from university network



```
C:\Users\Omar K. Aly>dig SOA www.uni-koblenz-landau.de

; <<>> DiG 9.11.0 <<>> SOA www.uni-koblenz-landau.de
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1896
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

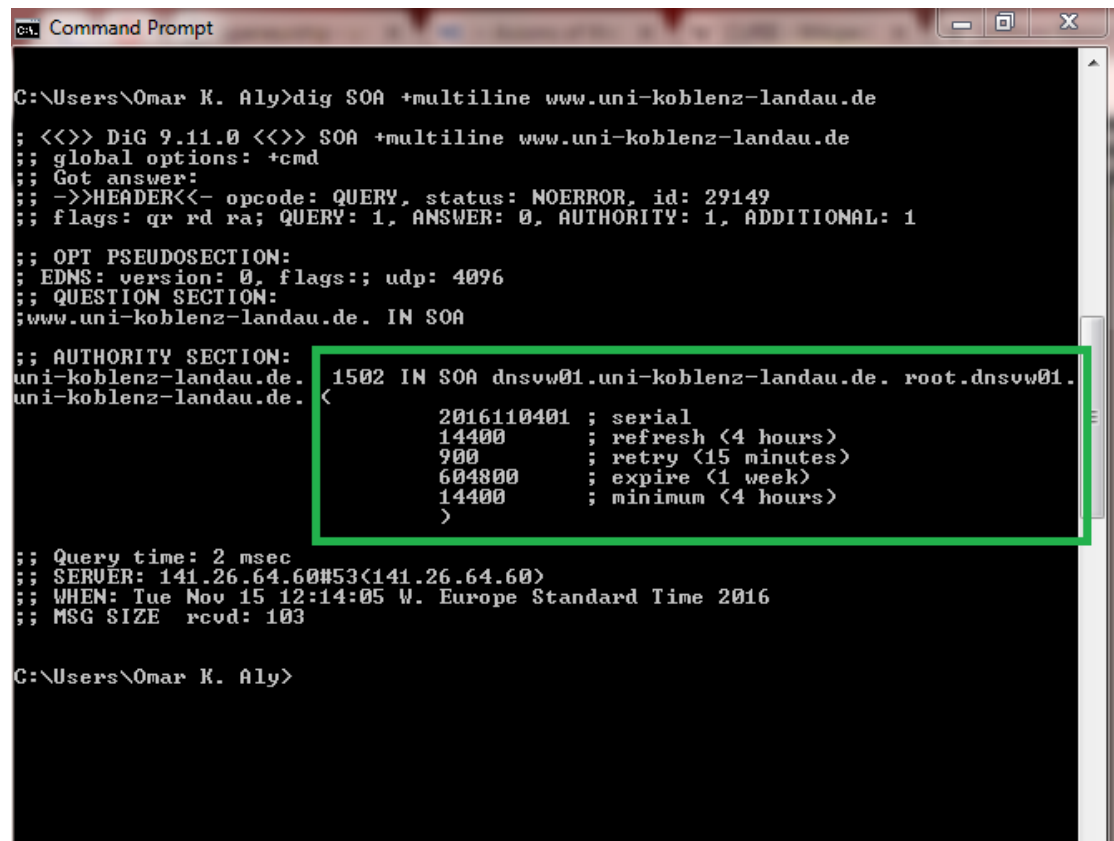
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:;, udp: 4096
;; QUESTION SECTION:
;www.uni-koblenz-landau.de.      IN      SOA

;; AUTHORITY SECTION:
uni-koblenz-landau.de. 1303     IN      SOA      dnsvw01.uni-koblenz-landau.de. root.dnsvw01.uni-koblenz-landau.de. 2016110401 14400 900 604800 14400

;; Query time: 91 msec
;; SERVER: 141.26.64.60#53(141.26.64.60)
;; WHEN: Tue Nov 15 12:17:24 W. Europe Standard Time 2016
;; MSG SIZE rcvd: 103
```

Figure 5: dig SOA - www.uni-koblenz-landau.de - University Network

3. Same definition as the Home network ones but as shown in the figure below there is some difference in some of the values.
 - TTL is 1502
 - Name Server: just different name server which is: "dnsvw01"
 - Serial number is sure different and is: 2016110401
 - Expiry: time is larger than that from home network and is: 604800 (1 week)
 - Minimum: is smaller than that from the home network and is: 14400



```
C:\Users\Omar K. Aly>dig SOA +multiline www.uni-koblenz-landau.de

;; <<>> DiG 9.11.0 <<>> SOA +multiline www.uni-koblenz-landau.de
;; global options: +cmd
;; Got answer:
;; ->>HEADER<- opcode: QUERY, status: NOERROR, id: 29149
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:;, udp: 4096
;; QUESTION SECTION:
;www.uni-koblenz-landau.de. IN SOA

;; AUTHORITY SECTION:
uni-koblenz-landau.de. 1502 IN SOA dnsvw01.uni-koblenz-landau.de. root.dnsvw01.
uni-koblenz-landau.de. <
                        2016110401 ; serial
                        14400      ; refresh <4 hours>
                        900       ; retry <15 minutes>
                        604800    ; expire <1 week>
                        14400    ; minimum <4 hours>
                        )

;; Query time: 2 msec
;; SERVER: 141.26.64.60#53(141.26.64.60)
;; WHEN: Tue Nov 15 12:14:05 W. Europe Standard Time 2016
;; MSG SIZE rcvd: 103

C:\Users\Omar K. Aly>
```

Figure 6: dig SOA +multiline - www.uni-koblenz-landau.de - University Network

2 Exploring DNS (10 Points)

In the first part of this assignment you were asked to develop a simple TCP Client Server. Now, using **that** client server setup. This time a url should be send to the server and the server will split the url into the following:

`http://www.example.com:80/path/to/myfile.html?key1=value1&key2=value2#InTheDocument`

1. Protocol
2. Domain
3. Sub-Domain
4. Port number
5. Path
6. Parameters
7. Fragment

The Protocol for sending the URL will be a string terminated with `\r \n`.

P.S.: You are **not** allowed to use libraries like `urlparse` for this question. You will also not use "Regular Expressions" for this.

Answer:

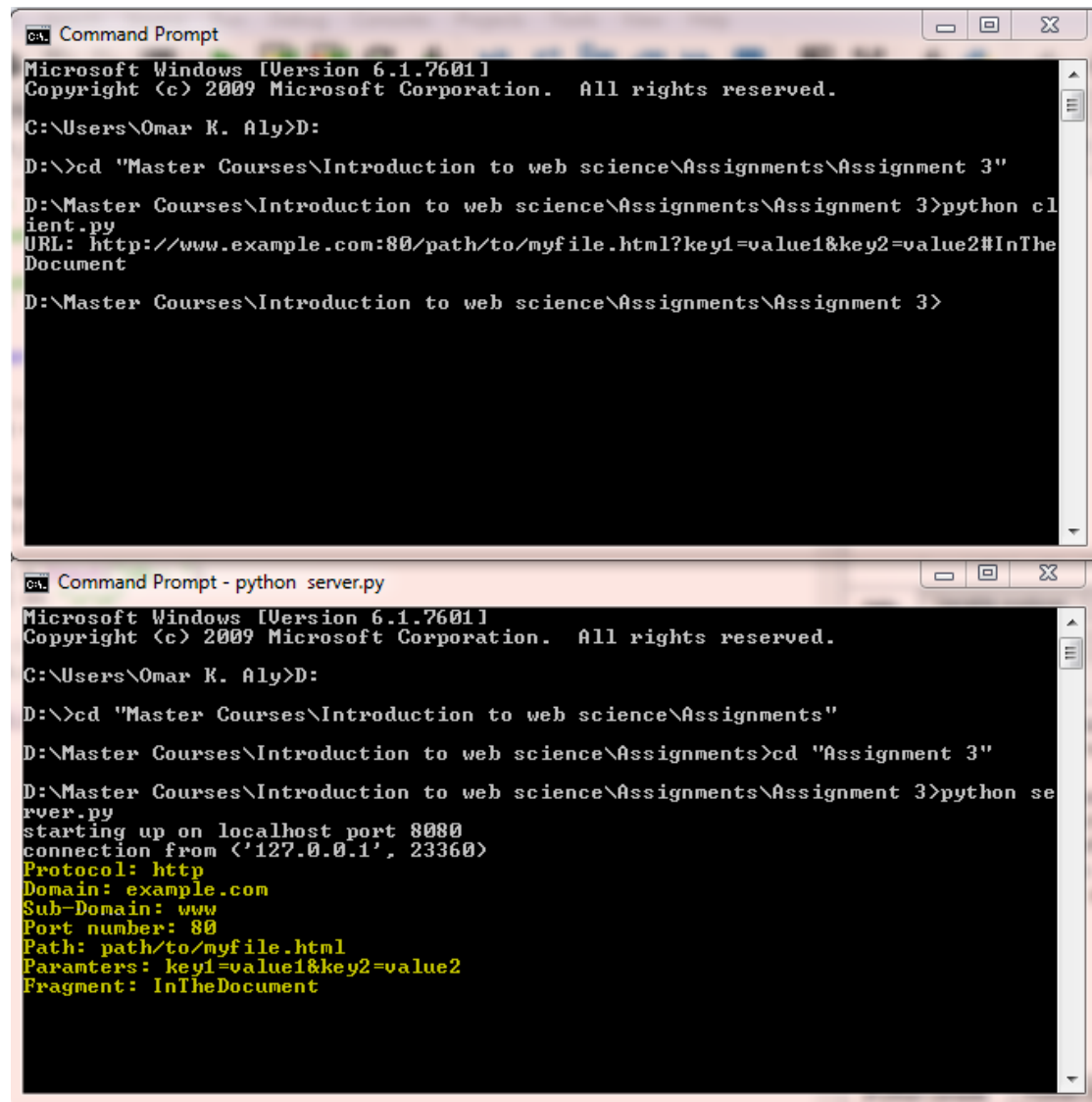
Client Code:

```
1: import socket
2:
3: # Create a TCP/IP socket
4: sck = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5:
6: # Connect the socket to the port where the server is listening
7: server_address = ('localhost', 8080)
8: sck.connect(server_address)
9:
10: url = input("URL: ")
11: url += "\r\n"
12:
13: #bytes and utf-8 is necessary for python 3.5
14: sck.send(bytes(url,'utf-8'))
15:
16: sck.close()
```

This code assume that any URL it will pars will be the same as the one given in this problem Server Code:


```
1: import socket
2:
3: # Create a TCP/IP socket
4: sck = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5:
6: # Bind the socket to the port
7: server_address = ('localhost', 8080)
8: print ('starting up on %s port %s' % server_address)
9: sck.bind(server_address)
10:
11: # Listen for incoming connections, the passed value indicates the maximum number
12: sck.listen(1)
13:
14: #empty string to concatenate result in
15: result = ""
16:
17: while True:
18:     # Wait for a connection
19:     connection, client_address = sck.accept()
20:     #try receiving data and with the finally option, its used for clearing action
21:     try:
22:         print ( 'connection from', client_address)
23:         while True:
24:             data = connection.recv(1024)
25:             #bytes and utf-8 is necessary for python 3.5
26:             url = data.decode('utf-8')
27:             # if we got data process it else break the loop of waiting
28:             if data:
29:
30:                 #splitting the url
31:                 splitted = url.split(":")
32:                 portPathParams = splitted[2].split("?")
33:                 paramsFragment = portPathParams[1].split("#")
34:                 domains = splitted[1].split(".")
35:                 portPath = portPathParams[0].split("/")
36:                 params = paramsFragment[0].split("&")
37:
38:                 #concatenating the final results in one string
39:                 result += "Protocol: " + splitted[0] + "\n"
40:                 result += "Domain: " + domains[1] + "." + domains[2] + "\n"
41:                 result += "Sub-Domain: " + domains[0].split("/") [1] + "\n"
42:                 result += "Port number: " + portPath[0] + "\n"
43:                 result += "Path: " + portPath[1]+"/"+portPath[2]+"/"+portPath[3] + "\n"
44:                 result += "Parameters: " + paramsFragment[0] + "\n"
45:                 result += "Fragment: " + paramsFragment[1]
46:                 print(result)
47:
48:             else:
49:                 break
```

```
50:
51:     finally:
52:         # Clean up the connection
53:         connection.close()
```



The figure consists of two screenshots of Windows Command Prompts. The top screenshot shows a client-side command prompt where the user navigates to a directory and runs a Python script. The bottom screenshot shows a server-side command prompt where the user navigates to a directory and runs a Python script, which then displays the details of an incoming HTTP connection.

```
Command Prompt
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Users\Omar K. Aly>D:
D:\>cd "Master Courses\Introduction to web science\Assignments\Assignment 3"
D:\Master Courses\Introduction to web science\Assignments\Assignment 3>python client.py
URL: http://www.example.com:80/path/to/myfile.html?key1=value1&key2=value2#InTheDocument
D:\Master Courses\Introduction to web science\Assignments\Assignment 3>
```

```
Command Prompt - python server.py
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Users\Omar K. Aly>D:
D:\>cd "Master Courses\Introduction to web science\Assignments"
D:\Master Courses\Introduction to web science\Assignments>cd "Assignment 3"
D:\Master Courses\Introduction to web science\Assignments\Assignment 3>python server.py
starting up on localhost port 8080
connection from ('127.0.0.1', 23360)
Protocol: http
Domain: example.com
Sub-Domain: www
Port number: 80
Path: path/to/myfile.html
Parameters: key1=value1&key2=value2
Fragment: InTheDocument
```

Figure 7: Sending URL @client processing it @server

3 DNS Recursive Query Resolving (5 Points)

You have solved the "Routing Table" question in Assignment 2. We updated the routing tables once more, resulting in the following tables creating the following topology

Table 1: Routing Table

Router1			Router2			Router3		
Destination	Next Hop	Interface	Destination	Next Hop	Interface	Destination	Next Hop	Interface
67.0.0.0	67.68.3.1	eth 0	205.30.7.0	205.30.7.1	eth 0	205.30.7.0	205.30.7.2	eth 0
62.0.0.0	62.4.31.7	eth 1	156.3.0.0	156.3.0.6	eth 1	88.0.0.0	88.6.32.1	eth 1
88.0.0.0	88.4.32.6	eth 2	26.0.0.0	26.3.2.1	eth 2	25.0.0.0	25.03.1.2	eth 2
141.71.0.0	141.71.20.1	eth 3	141.71.0.0	141.71.26.3	eth 3	121.0.0.0	121.0.3.1	eth 3
26.0.0.0	141.71.26.3	eth3	67.0.0.0	141.71.20.1	eth 3	156.3.0.0	205.30.7.1	eth 0
156.3.0.0	88.6.32.1	eth 2	62.0.0.0	141.71.20.1	eth 3	26.0.0.0	205.30.7.1	eth 0
205.30.7.0	141.71.26.3	eth 3	88.0.0.0	141.71.20.1	eth 3	141.71.0.0	205.30.7.1	eth 0
25.0.0.0	88.6.32.1	eth 2	25.0.0.0	205.30.7.2	eth 0	67.0.0.0	88.4.32.6	eth 1
121.0.0.0	88.6.32.1	eth 2	121.0.0.0	205.30.7.2	eth 0	62.0.0.0	88.4.32.6	eth 1

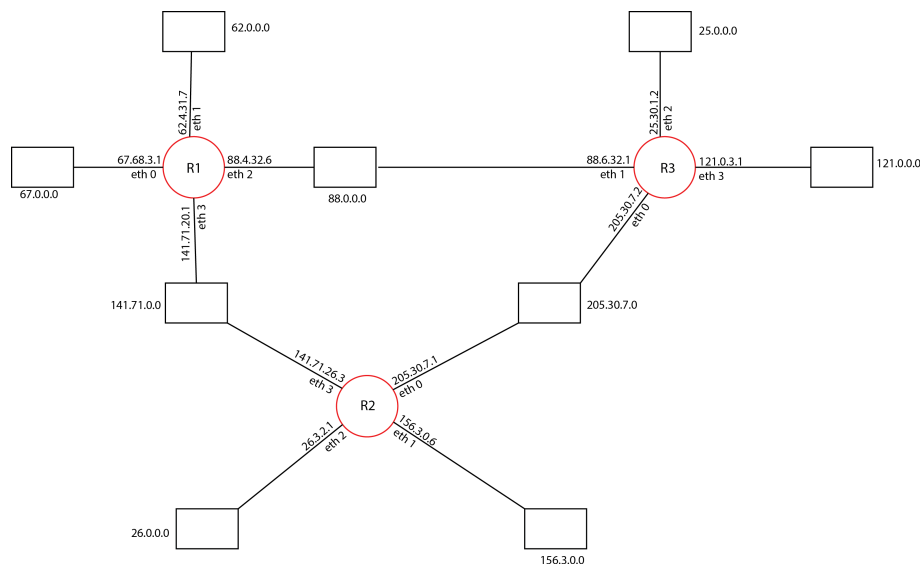


Figure 8: DNS Routing Network

Let us assume a client with the following ip address 67.4.5.2 wants to resolve the following domain `subdomain.webscienceexampledomain.com` using the DNS.

You can further assume the root name server has the IP address of 25.8.2.1 and the name-server for `webscienceexampledomain.com` has the IP address 156.3.20.2. Finally the sub-domain is handled by a name server with the IP of 26.155.36.7.

Please explain how the traffic flows through the network in order to resolve the recursive DNS query. You can assume ARP tables are cached so that no ARP-requests have to be made.

Hint: You can start like this:

67.4.5.2 creates an IP packet with the source address XXXXXX an destination address YYYYYY inside there is the DNS request. This IP packet is send as an ethernet frame to ZZZZZ. ZZZZZ receives the frame and forwards the encapsulated IP packet to

Also you can assume the DNS requests and responses will fit inside one IP packet. You also don't have to write down the specific DNS requests and responses in hex.

Answer:

Important Note: We assume, that 67.4.5.2 itself is a DNS Server that is responsible for resolving a query, by doing recursive DNS query.

67.4.5.2 sends an IP packet with the source address 67.4.5.2 and destination address 25.8.2.1 inside there is a DNS request. This IP packet is send as an ethernet frame to 67.68.3.1. 67.68.3.1 encapsulates the frame and forwards the encapsulated ip packet to 88.6.32.1. This router has an address 25.03.1.2 in 25.0.0.0 network, which means this router is in the same network as the root server and therefore forwards the packet to it's destination. When the packet arrives at the root server, he needs to send the address of the name server for webscienceexampledomain.com back to the requesting host. Which means an ip packet will be assembled with source address 25.8.2.1 and destination address 67.4.5.2. So the packet will take the same route back to 67.4.5.2.

Then 67.4.5.2 sends an IP packet with the source address 67.4.5.2 and destination address 156.3.20.2 inside there is a DNS request. This IP packet is send as an ethernet frame to 67.68.3.1. 67.68.3.1 encapsulates the frame and forwards the encapsulated ip packet to 88.6.32.1. The next router forwards the packet to 205.30.7.1 This router has an address 156.3.0.6 in 156.3.0.0 network, which means this router is in the same network as the responsible name server and therefore forwards the packet to it's destination. When the packet arrives at the name server, he needs to send the address of the name server for subdomain.webscienceexampledomain.com back to the requesting host. Which means an ip packet will be assembled with source address 156.3.20.2 and destination address 67.4.5.2. For the route back the packet will be send to 156.3.0.6 and via 141.71.26.3 back to Router 1. Which is in 67.0.0.0 network and is responsible for delivering the packet to the original host.

Then 67.4.5.2 sends an IP packet with the source address 67.4.5.2 and destination address 26.155.36.7 inside there is a DNS request. This IP packet is send as an ethernet frame to 67.68.3.1. 67.68.3.1 encapsulates the frame and forwards the encapsulated ip packet to 141.71.20.3. This router has an address 26.3.2.1

in 26.0.0.0 network, which means this router is in the same network as the root server and therefore forwards the packet to it's destination. When the packet arrives at the name server, he needs to send the IP address corresponding to subdomain.webscienceexampledomain.com back to the requesting host. Which means an ip packet will be assembled with source address 26.155.36.7 and destination address 67.4.5.2. So the packet will take the same route back to 67.4.5.2. and the original host can now communicate directly with subdomain.webscienceexampledomain.com.

Important Notes

Submission

- Solutions have to be checked into the github repository. Use the directory name `groupname/assignment3/` in your group's repository.
- The name of the group and the names of all participating students must be listed on each submission.
- Solution format: all solutions as *one* PDF document. Programming code has to be submitted as Python code to the github repository. Upload *all* `.py` files of your program! Use UTF-8 as the file encoding. *Other encodings will not be taken into account!*
- Check that your code compiles without errors.
- Make sure your code is formatted to be easy to read.
 - Make sure you code has consistent [indentation](#).
 - Make sure you comment and document your code adequately in English.
 - Choose consistent and intuitive names for your identifiers.
- Do *not* use any accents, spaces or special characters in your filenames.

Acknowledgment

This latex template was created by Lukas Schmelzeisen for the tutorials of "Web Information Retrieval".

\LaTeX

Currently the code can only be build using [LuaLaTeX](#), so make sure you have that installed. If on Overleaf, there's an error, go to settings and change the \LaTeX engine to LuaLaTeX.