Introduction to Web Science

Assignment 2

Mariya Chkalova
mchkalova@uni-koblenz.de

Arsenii Smyrnov
smyrnov@uni-koblenz.de

Simon Schauß
sschauss@uni-koblenz.de

Group Tango
Institute of Web Science and Technologies
Department of Computer Science
University of Koblenz-Landau

Submission until: November 9, 2016, 10:00 a.m.
Tutorial on: November 11th, 2016, 12:00 p.m.

The main objective of this assignment is for you to use different tools with which you can understand the network that you are connected to or you are connecting to in a better sense. 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.

1 IP Packet (5 Points)

Consider the IPv4 packet that is received as:

`4500 062A 42A1 8001 4210 XXXX C0A8 0001 C0A8 0003`

Consider `XXXX` to be the check sum field that needs to be sent with the packet.

Please provide a step-by-step process for calculating the "Check Sum".

Step **1**:

calculate the sum of each 16bit block

$$
\begin{aligned}
&+4500\\
&+062A\\
&+42A1\\
&+8001\\
&+4210\\
&+C0A8\\
&+0001\\
&+C0A8\\
&+0003\\
&=2D130
\end{aligned}
$$

Step **2**:

excerpt the carry bit and calculate the sum:

$$D130 + 2 = D132$$

Step **3**:

build the inverse of each 4bit block in the hexadecimal system

$$D132 \Rightarrow 2ECD$$

Therefore the checksum is 2ECD

2 Routing Algorithm (10 Points)

UPDATE. The bold fonted numbers have been updated on Monday Nov. 7th. (If you already have done so feel free to use the old numbers. But the solution with the old version will be more complex than the solution with the updated numbers.)

You have seen how routing tables can be used to see how the packets are transferred across different networks. Using the routing tables below of Router 1, 2 and 3:

1. Draw the network `[6 points]`
2. Find the shortest path of sending information from 67.68.2.10 network to 25.30.3.13 network `[4 points]`

Table 1: Router 1

Destination	Next Hop	Interface
67.0.0.0	67.68.3.1	eth 0
62.0.0.0	62.4.31.7	eth 1
88.0.0.0	88.4.32.6	eth 2
141.**71**.0.0	141.**71**.20.1	eth 3
26.0.0.0	141.71.26.3	eth 3
156.3.0.0	141.71.26.3	eth 3
205.**30.7**.0	141.71.26.3	eth 3
25.0.0.0	88.6.32.1	eth 2
121.0.0.0	88.6.32.1	eth 2

Table 2: Router 2

Destination	Next Hop	Interface
141.**71**.0.0	141.71.26.3	eth 3
205.**30.7**.0	205.**30.7**.1	eth 0
26.0.0.0	26.3.2.1	eth 2
156.**3**.0.0	156.3.0.6	eth 1
67.0.0.0	141.**71**.20.1	eth 3
62.0.0.0	141.**71**.20.1	eth 3
88.0.0.0	141.**71**.20.1	eth 3
25.0.0.0	205.30.7.2	eth 0
121.0.0.0	205.30.7.2	eth 0

1.

See: 1

Table 3: Router 3

Destination	Next Hop	Interface
205.**30.7**.0	205.30.7.2	eth 0
88.0.0.0	88.6.32.1	eth 1
25.0.0.0	25.30.1.2	eth 2
121.0.0.0	121.0.3.1	eth 3
156.**3**.0.0	205.**30**.7.1	eth 0
26.0.0.0	205.**30**.7.1	eth 0
141.0.0.0	205.**30**.7.1	eth 0
67.0.0.0	88.4.32.6	eth 1
62.0.0.0	88.4.32.6	eth 1

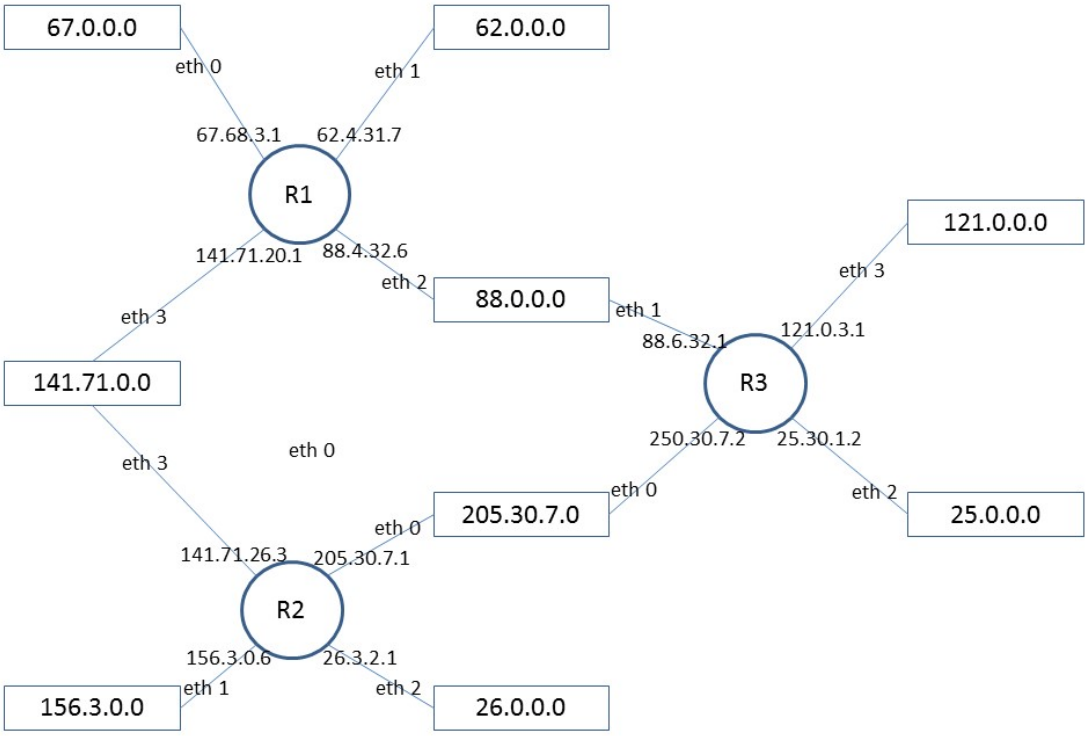

Figure 1: Network

2.

The shortest path of sending information from 67.68.2.10 network to 25.30.3.13 network:

send to Router 1 IP=67.68.3.1;

Router 1 will forward package to Router 2 IP=88.6.32.1

Router 2 will deliver package via 25.0.0.0 network to the destination computer.

3 Sliding Window Protocol (10 Points)

Sliding window algorithm, which allows a sender to have more than one unacknowledged packet "in flight" at a time, improves network throughput.

Let us consider you have 2 Wide Area Networks. One with a bandwidth of 10 Mbps (Delay of 20 ms) and the other with 1 Mbps (Delay of 30 ms) . If a packet is considered to be of size 10kb. Calculate the window size of number of packets necessary for Sliding Window Protocol. [`5 points`]
Answer:
WAN1 Window size: (2 * 10 Mb/s * 0.02s)/10 Kb = 40
WAN2 Window size: (2 * 1 Mb/s * 0.03s)/10 Kb = 6

Since you now understand the concept of Window Size for Sliding Window Protocol and how to calculate it, consider a window size of 3 packets and you have 7 packets to send. Draw the process of `Selective Repeat Sliding Window Protocol` where in the 3rd packet from the sender is lost while transmission. Show diagrammatically how the system reacts when a packet is not received and how it recuperates from that scenario. [`5 points`]
Answer:

In Selective-Repeat ARQ, the receiver while keeping track of sequence numbers, buffers the frames in memory and sends NACK for only frame which is missing or damaged. The sender in this case, sends only packet for which NACK is received.

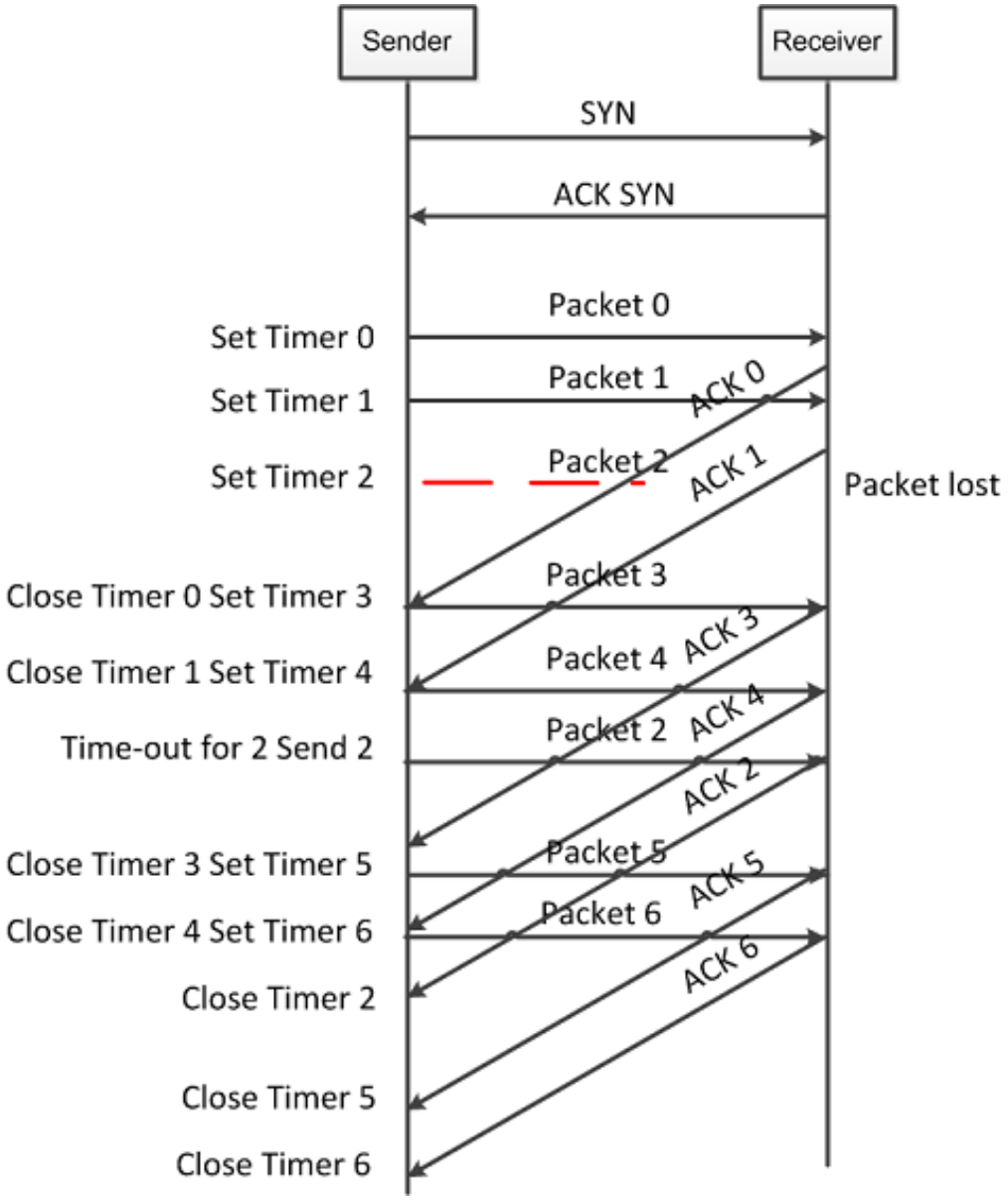

Figure 2: Packet loss

4 TCP Client Server (10 Points)

Use the information from the socket documentation and create: [`4 points`]

1. a simple TCP Server that listens to a
2. Client

Note: Please use port `8080` for communication on `localhost` for client server communication.

Given below are the following points that your client and server must perform: [`6 points`]

1. The *Client* side asks the user to input their name, age & *matrikelnummer* which is then sent to the server all together.
2. Develop a protocol for sending these three information and subsequently receiving each of the information in three different lines as mentioned in the below format. Provide reasons for the protocol you implemented.
3. Format the output in a readable format as:
   `Name: Korok Sengupta;`
   `Age: 29;`
   `Matrikelnummer: 21223ert56`

Provide a snapshot of the results along with the code.

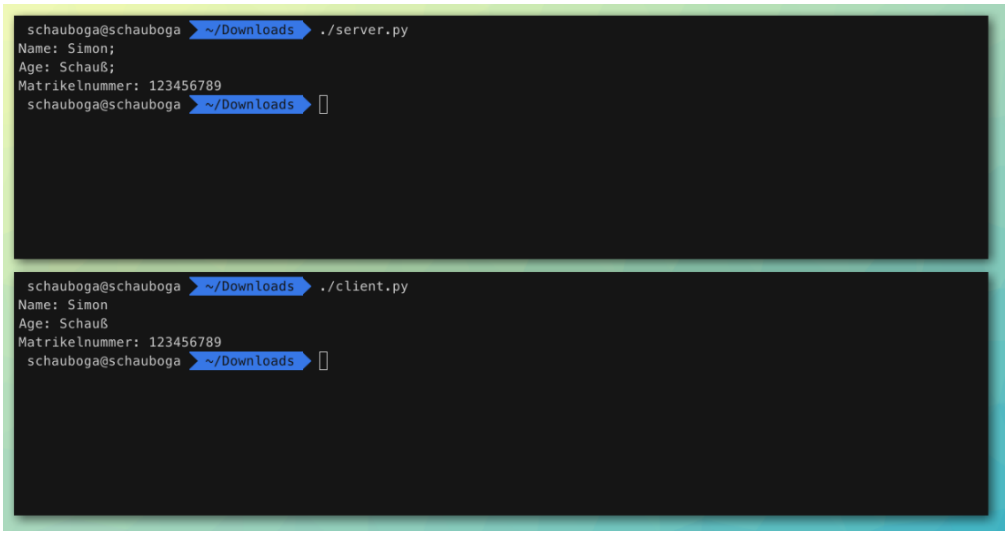

Figure 3: TCP client/server in action

Listing 1 TCP client

```
#!/usr/bin/python

import json
import socket

IP = '127.0.0.1'
PORT = 8080
BUFFER_SIZE = 1

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect((IP, PORT))
try:
    name = input('Name: ')
    age = input('Age: ')
    matrikelnummer = input('Matrikelnummer: ')
    data = json.dumps(
            {
                'name': name,
                'age': age,
                'matrikelnummer': matrikelnummer
            }).encode('UTF-8')
    client_socket.sendall(data)
finally:
    client_socket.close()
```

Listing 2 TCP server

```
#!/usr/bin/python
import socket
import json

IP = '127.0.0.1'
PORT = 8080
BUFFER_SIZE = 1

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind((IP, PORT))
server_socket.listen(1)
connection, address = server_socket.accept()
data = ''
try:
    while True:
        buffer = connection.recv(BUFFER_SIZE)
        data += buffer.decode('UTF-8')
        if not buffer:
            dict = json.loads(data)
            print('Name: %s;' % dict['name'])
            print('Age: %s;' % dict['age'])
            print('Matrikelnummer: %s' % dict['matrikelnummer'])
            break
finally:
    server_socket.close()
```

Important Notes

Submission

- Solutions have to be checked into the github repository. Use the directory name `groupname/assignment2/` 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, go to settings and change the LaTeXengine to `LuaLaTeX` in case you encounter any error