

Introduction to Web Science

Assignment 2

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 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".

Answer:

Step 1:

$$4500 + 062A = 4B2A$$

$$4B2A + 42A1 = 8DCB$$

$$8DCB + 8001 = 10DCC$$

$$10DCC + 4210 = 14FDC$$

$$14FDC + C0A8 = 21084$$

$$21084 + 0001 = 21085$$

$$21085 + C0A8 = 2D12D$$

$$2D12D + 0003 = 2D130$$

Step 2:

Now adding carry bit 2 from answer 2D130 :

$$D130 + 2 = D132$$

Step 3:

Now taking 1's complement

$$FFFF - D132 = 2ECD$$

So final checksum is 2ECD

To Verify checksum :

$$2D130 + 2ECD = 2FFFD$$

now add carry bit :

$$FFFD + 2 = FFFF$$

Reference : "https://en.wikipedia.org/wiki/IPv4_header_checksum"

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

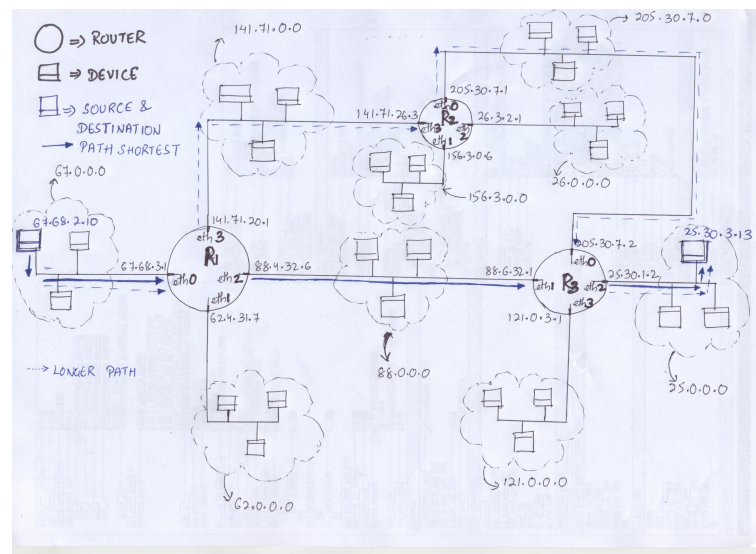
Destination	Next Hop	Interface
141. 71 .0.0	141.71.26.3	eth 3
205. 30.7 .0	205. 30.7 .1	eth 0
26.0.0.0	26.3.2.1	eth 2
156. 3 .0.0	156.3.0.6	eth 1
67.0.0.0	141. 71 .20.1	eth 3
62.0.0.0	141. 71 .20.1	eth 3
88.0.0.0	141. 71 .20.1	eth 3
25.0.0.0	205.30.7.2	eth 0
121.0.0.0	205.30.7.2	eth 0

Answer:

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

1.

**Figure 1: Network**

2. The required shortest path is as depicted in the network, **Path 1, i.e from Router 1 to Router 3 directly.**
 - Another(longer) path for the same source and destination is Path 2, that from Router 1, then to Router 2 and then Router 3 in the end. It has 3 hops from, one to Router1, then to Router2, and then to Router3.
 - Path 1 is shortest path as compared to path 2 because it contains less no of hops and less ARP requests in network to get MAC address to reach to the destination. It has 2 hops, one to Router 1 and then to Router 3.

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]

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:

1. Given Two Wide Area Networks One is of 10 Mbps and other is of 1Mbps
The Delay of the two Wide Area Networks Are 20ms and 30 ms respectively
The packet is considered to be of 10Kb
 $Window\ Size = Bandwidth * delay$

Window size of 10 Mbps WAN

10 Mbps = 10000000 bytes and 20 ms = 0.02 sec

$= 10000000 * 0.02$

$= 500000000$ bits

$= 500000000 / 8$

$= 625$ Bytes

the packet is of 10 Kb = 125 Bytes

Number of packets that can be slid are $625 / 125 = 5$

The window size of 10Mbps Network is 625 Bytes and Number of packets that can be slid in are 5 .

Window size of 1Mbps WAN

1 Mbps = 1000000 bytes and 30 ms = 0.03 sec

$= 1000000 * 0.03$

$= 30000$ bits

$= 30000 / 8$

$= 375$ Bytes

the packet is of 10 Kb = 125 Bytes

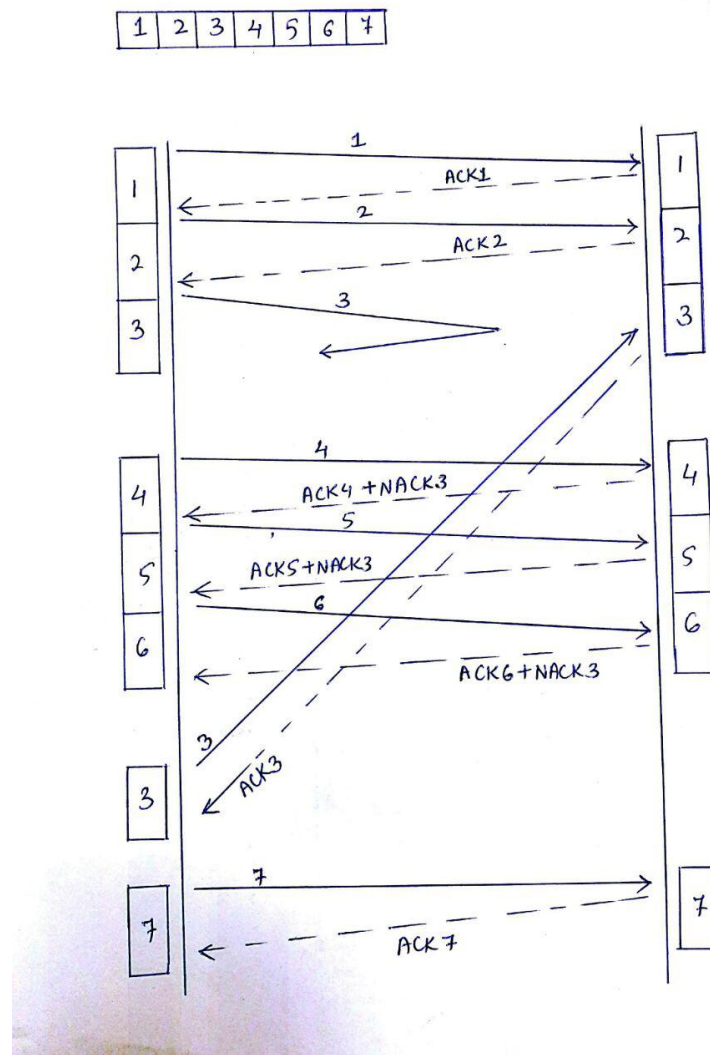
Number of packets that can be slid are $375/125=3$

The window size of 1Mbps Network is 375 Bytes and Number of packets that can be slid in are 3 .

Selective repeat sliding Window protocol:

- a) The data is sent with a buffer of 3 packets. Here, in our figure, the numbers denote the packet number which is sent.
- b) receiver then acknowledges the same packet and this is denoted, in our figure, as the ACK(N) where N denotes the packet number which is sent. If the packet is lost the receiver keeps track of the sequence number of the earliest frame it has not received, and sends that number (NACK i.e negative acknowledgement) with every acknowledgement (ACK) it sends. This continues till the window is slid, now it sender sends the packet for which the NACK was received, followed by the next packets in the following window.

2.

**Figure 2:** Selective repeat sliding window protocol

Reference : "https://en.wikipedia.org/wiki/Selective_Repeat_ARQ"

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]

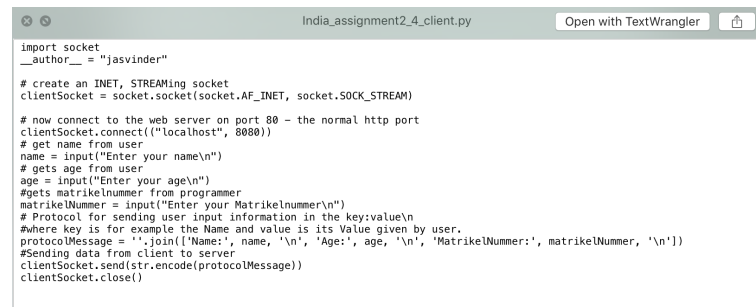
1. The *Client* side asks the user to input their name, age & *matrikelnummer* which is then sent to the server all together.
2. Develop a protocol for sending these three information and subsequently receiving each of the information in three different lines as mentioned in the below format. Provide 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.

Answer:

1. Client code is in file named `India_assignment2_4_client.py` and that of server is `India_assignment2_4_server.py`
2.
 - We have developed a protocol for sending user input information at once in the key:value pair from Client side where key is, e.g. the Name and value is its Value given by user and we have used `join()` and encoded the whole input.
 - In server side, we have made function to decode the data from client into key value pair of lists. E.g a data in format `"Name:jass\nAge:18\nMatrikelnummer:2168756\n"` is decoded into a list like this `[["Name", "jass"], ["Age", "18"], ["Matrikelnummer", "2168756"]]`
 - It splits the input in different lines when it encounters new line character and then finally splits the data in each line around the occurrence of colon.

3.



```

import socket
__author__ = "jasvinder"

# create an INET, STREAMing socket
clientSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

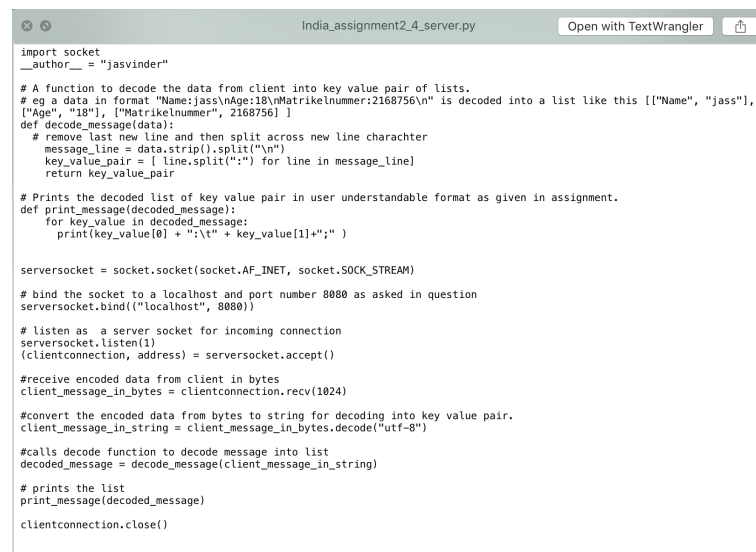
# now connect to the web server on port 80 - the normal http port
clientSocket.connect(("localhost", 8080))

# get name from user
name = input("Enter your name\n")
# gets age from user
age = input("Enter your age\n")
# gets matrikelnummer from programmer
matrikelnummer = input("Enter your Matrikelnummer\n")
# Protocol for sending user input information in the key:value\n
# where key is for example the Name and value is its Value given by user.
protocolMessage = ''.join(['Name:', name, '\n', 'Age:', age, '\n', 'Matrikelnummer:', matrikelnummer, '\n'])
# Sending data from client to server
clientSocket.send(str.encode(protocolMessage))
clientSocket.close()

```

Figure 3: Client code

4.



```

import socket
__author__ = "jasvinder"

# A function to decode the data from client into key value pair of lists.
# eg a data in format "Name:jass\nAge:18\nMatrikelnummer:2168756\n" is decoded into a list like this [{"Name", "jass"}, {"Age", "18"}, {"Matrikelnummer", "2168756"}]
def decode_message(data):
    # remove last new line and then split across new line character
    message_line = data.strip().split("\n")
    key_value_pair = [line.split(":") for line in message_line]
    return key_value_pair

# Prints the decoded list of key value pair in user understandable format as given in assignment.
def print_message(decoded_message):
    for key_value in decoded_message:
        print(key_value[0] + ":\t" + key_value[1] + ";")

serversocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# bind the socket to a localhost and port number 8080 as asked in question
serversocket.bind(("localhost", 8080))

# listen as a server socket for incoming connection
serversocket.listen(1)
(clientconnection, address) = serversocket.accept()

# receive encoded data from client in bytes
client_message_in_bytes = clientconnection.recv(1024)

# convert the encoded data from bytes to string for decoding into key value pair.
client_message_in_string = client_message_in_bytes.decode("utf-8")

# calls decode function to decode message into list
decoded_message = decode_message(client_message_in_string)

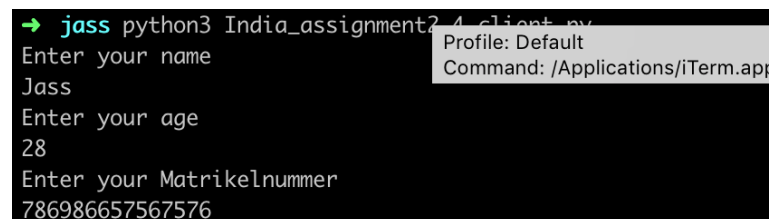
# prints the list
print_message(decoded_message)

clientconnection.close()

```

Figure 4: Server code

5.



```

→ jass python3 India_assignment2_4_client.py
Enter your name
Jass
Enter your age
28
Enter your Matrikelnummer
786986657567576

```

Figure 5: Client's code output

6.

A terminal window with two tabs. The first tab is titled '..../introws/jass (zsh) 1' and the second is '..../introws/jass (zsh) 2'. The first tab shows a green prompt '→ jass' followed by the command 'python3 India_assignment2_4_server.py'. The output of the command is displayed on the next three lines: 'Name: Jass;', 'Age: 28;', and 'MatrikelNummer: 786986657567576;'. The second tab shows a green prompt '→ jass' followed by a cursor.

```
→ jass python3 India_assignment2_4_server.py
Name: Jass;
Age: 28;
MatrikelNummer: 786986657567576;
→ jass
```

Figure 6: Server's code output

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".

L^AT_EX

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 L^AT_EX engine to LuaLaTeX in case you encounter any error