Names:

Sanket Kulkarni(A20537896)
Deep Pawar(A20545137)
Nitesha Paatil(A20544932)
Sayali Bhongade(A20524430)

Professor: Chlebus Edward
Institute: Illinois Institute of Technology

CS 542 – Computer Networks I: Fundamentals Fall 2023 HW2 (60 points)

Submission instructions

– ***Due date: Wednesday, Dec. 6, 11:59 pm Central Time***

– ***Late submissions and submissions violating these instructions will NOT be accepted.***

– ***No handwritten submissions. No credit will be given for handwritten submissions.***

– ***Teamwork is allowed (max. 4 students/team). Individual submissions are also OK.***

– ***Upload your assignment (pdf format only) to Blackboard. Submissions in formats other than pdf will be disregarded. The Beacon students: upload your submissions to Lumina.***

– ***One submission per team only. Write down names, A#, and section numbers of all the team members on the front page. Do not submit multiple copies of your HW (e.g. by each team member). It is very confusing and will be penalized. Clearly indicate how each team member contributed to your teamwork.***

– ***Show your work and explain every step of your solution for full credit. Only partial credit will be given for a correct final answer with missing calculations, no supporting explanations or unclear justifications.***

– ***My TAs Pranav Saji (psaji@hawk.iit.edu) and Aditya Sai Kolluru(akolluru@hawk.iit.edu) are responsible for grading this assignment. Feel free to ask questions is something is not clear but don't send me or my TAs:***

- ***Your partial solutions with inquiries "Is that what you expect?".***
- ***Questions the answers to, may give explicit hints on how to solve the HW problems.***

Student Name	Section	A#	Homework Contribution
Sanket Kulkarni	**01**	**A20537896**	**25%**
Deep Pawar	**01**	**A20545137**	**25%**
Nitesha Paatil	**01**	**A20544932**	**25%**
Sayali Bhongade	**01**	**A20524430**	**25%**

1. Given an IP datagram with the fragmentation offset of 0000001011010_2, HLEN of 5_{16} and the total length of $007A_{16}$, find the numbers of the first byte and the last byte of data in this datagram **(3 points)**

Ans:

Given,

Fragmentation Offset: 0000001011010_2, which is 90_{10}
Header Length (HLEN) = 5 × 4 = 20 bytes
Total Length: $007A_{16} = 007(10)_{16}$
Convert 007A to decimal = $007A_{16} = 122_{10}$
Total Length = 122

a. First bytes:

By using the following formula:

Calculate the first byte = Fragmentation Offset * 8
= 90 * 8
= 720
Therefore, First bytes = 720 bytes

b. Last bytes:

Data Length = Total Length - Header Length
= 122 - 20
= 102

Now, calculate the last byte:

Last Byte = First byte + Data Length - 1
= 720 +102 - 1
= 822 – 1
= 821
Therefore, last bytes = 821 bytes

So, the first byte of data is 720 and the last byte of data is 821 in the given IP datagram.

2. Consider fragmenting an original IP datagram whose total size is 6000_{10} bytes with a base header only. The offset of the second fragment is 98_{16}. Answer the following questions: **(6 points)**

a. How many fragments are there? Give the data range for each of them. (Assume that all the fragments except the last one is equal.) **(3 points)**

Ans:

Here,

Original IP datagram size: 6000 bytes
Base Header = 20 bytes
Offset of the second fragment: 98 in hexadecimal

To calculate the number of fragments:

Number of fragments = 5 fragments and Data range for each fragment:

a. **Fragment 1: 0 to 1215 bytes**
b. **Fragment 2: 1216 to 2431 bytes**
c. **Fragment 3: 2432 to 3647 bytes**
d. **Fragment 4: 3648 to 4863 bytes**
e. **Fragment 5: 4864 to 5979 bytes**

b. What is the total size of each fragment? **(2 points)**

Ans:

For all fragments except the last one, the size is the same (fragment size). The last fragment have a smaller size.

Fragment size for (1, 2, 3 and 4th fragment) = 1236 bytes
Fragment size of 5th fragment = 1136 bytes

c. What is the fragmentation offset of the last fragment? **(1 point)**

fragmentation offset of the last fragment = 4864 / 8
= 608

The fragmentation offset of last fragment = 608

3. The first few hexadecimal digits of an IP datagram are as follows: 4E00 00B4 0034 408F. Find the total length, the header length and the data size. Is there the next fragment? Can this datagram be fragmented? **(5 points)**

Ans:

a. Header Length:

Here,
IP datagram is 4E00 00B4 0034 408F
The second 4 bits (1 hexadecimal digits) are allocated to the header length = E
Convert E to decimal = $E_{16} = 14_{10}$ (14 * 4 = 56 bytes)

So, the header length in decimal format is 56 bytes

b. Total Length:

IP datagram is 4E00 00B4 0034 408F
The 16 bits (4 hexadecimal digits) are allocated to the total length = 00B4
Convert 00B4 to decimal = $00B4_{16} = 180_{10}$

So, the total length in decimal format is 180 bytes

c. Data Size:

Data Size = Total Length - Header Length
= 180 bytes - 56 bytes
= 124 bytes

So, the data size in decimal format is 124 bytes

d. Is there the next fragment?

IP datagram is 4E00 00B4 0034 408F
Converting 408F to binary = $408F_{16} = 0100000010001111_2$
From above conversion, the first 3 bits represent the flags.
Flag bits = 010
The relevant flags are: Reserved (1st bit), Do not Fragment (2nd bit), More Fragments (3rd bit)

The 3rd bit is 0 in this case, indicating that there is no next fragment.

Therefore, there are no next Fragment.

e. Can this datagram be fragmented?

Flag bits = 010
The relevant flags are: Reserved (1st bit), Do not Fragment (2nd bit), More Fragments (3rd bit)
The 2nd bit is 1, indicating that this datagram cannot be fragmented.

Therefore, this datagram cannot be fragmented.

4. An IP packet with HLEN = E_{16} carries 2540_{10} bytes of data. What is the size of the "*Options"*? What is the value (in the hexadecimal format) of the "*Total length"* field? **(2 points)**

Ans:

Here,

Header Length (HLEN): E_{16}
Convert E_{16} to decimal = $E_{16} = 14_{10}$

Header Length in bytes = 14 × 4
= 56 bytes

Therefore, size of header is 56 bytes

b. Size of the Options:

The size of the Options field can be determined by subtracting the standard header length (20 bytes) from the total header length.

Options size = Total header length − Standard header length
= 56 bytes – 20 bytes
= 36 bytes

Therefore, size of the options field is 36 bytes

c. Total Length:

Total Length = Header length + Data length
= 56 bytes + 2540 bytes
= 2596 bytes

Convert 2596_{10} to hexadecimal = $2596_{10} = A24_{16}$

Therefore, the size of options is 36 bytes and the value of the Total Length field in hexadecimal format is A24

5. An original IP datagram, that carries 3470_{10} bytes of data and has only the base header, was fragmented. The first fragment contains bytes from 0 to 399. All the fragments except the last one are equal. What is the total overhead (in bytes) needed to send all the data of the original datagram to the destination? **(2 points)**

Ans:

Here,

Original IP Datagram carries = 3470 [Data Length]

Given Base Header = 20 bytes
Given First Fragment = 0 – 399
Total bytes = 400

All the fragments except the last one is equal.

In order to calculate the overhead, we first need to calculate the total number of fragments we have.

Total number of fragments = Total data length / Total length of first fragment
= 3470 / 400
= 8.675 ~ 9

Total number of fragments = 9

Total number of overhead (in bytes) needed to send all the data of the original datagram to the destination is,

= Total number of fragments * Base Header
= 9 * 20
= 180

Therefore, Overhead (in bytes) = 180 bytes

6. An IP packet has arrived with a "*D"* bit value of 0, an "*M"* bit value of 0, and fragmentation offset value to zero. Is this packet the first fragment, the last fragment, the middle fragment, or the only fragment? **(2 points)**

Ans:

Here,

"D" bit value is 0: The packet can be fragmented.
"M" bit value is 0: This is the last fragment.
Fragmentation offset value is zero

Given that the "M" bit is 0, it means that this is the last fragment. Additionally, since the Fragment Offset is zero, it means that this is the beginning of the original datagram.

Therefore, the packet is the only fragment.

7. A header of a UDP datagram is (in the hexadecimal format): 0315 C43B 00B3 001C (give your answers in the decimal format) **(4 points)**

a. What is the source port number?

Ans:

Here,
UDP datagram is 0315 C43B 00B3 001C
The first 16 bits (4 hexadecimal digits) are allocated to the source port number = 0315

Convert 0315 to decimal = $0315_{16} = 789_{10}$

So, the source port number in decimal format is 789

b. What is the destination port number?

Ans:

Here,
UDP datagram is 0315 C43B 00B3 001C
The next 16 bits (4 hexadecimal digits) are allocated to the destination port number = C43B
Convert C43B to decimal = $C43B_{16} = 50235_{10}$

So, the destination port number in decimal format is 50235

c. What is the total length of this UDP datagram?

Ans:

Here,
UDP datagram is 0315 C43B 00B3 001C
The next 16 bits (4 hexadecimal digits) are allocated to the total length of the UDP datagram = 00B3
Convert 00B3 to decimal: $00B3_{16} = 179_{10}$

So, the total length of the UDP datagram in decimal is 179

d. Is this UDP datagram sent from client to server or vice versa?

Ans:

789 is the source port number. This falls between port numbers 0 and 1023. Furthermore, these ports are a part of well-known port numbers. Servers use well-known port numbers.

In this case, a server is the source.

50235 is the destination port number. This falls inside the range of private or dynamic port numbers (49161–65535). Additionally, clients use these ports.

Thus, the client is the destination here.

The UDP datagram is sent from Server to Client for this reason.

8. An IP packet has the base header and the total size of 1200_{10} bytes. A UDP datagram is encapsulated in this IP packet. How many bytes of data does this UDP datagram carry? **(3 points)**

Ans:

Here,

Total size of IP packet = 1200 bytes
The base IPv4 header = 20 bytes.
Size of UDP Datagram = Total Size of IP Packet −Size of IP Header

= 1200 bytes – 20 bytes
= 1180 bytes

Therefore, Total size of UDP Datagram is 1180 bytes

The UDP header is typically 8 bytes. Therefore, we can calculate size of UDP data as follows:

Size of UDP Data = Size of UDP Datagram − Size of UDP Header
= 1180 bytes – 8 bytes
= 1172 bytes

Therefore, the UDP datagram carries 1172 bytes of data.

9. The initial sequence number in the TCP client-server transmission was 4357. The highest ACK number that the server sent to the client was 10842. How many bytes of data were successfully transmitted from the client to the server? **(2 points)**

Ans:

The number of bytes successfully transmitted from the client to the server can be determined by calculating the difference between the highest ACK number sent by the server and the initial sequence number set by the client.

Bytes Successfully Transmitted = Highest ACK Number − Initial Sequence Number
= 10842 − 4357
= 6485

Therefore, 6485 bytes of data were successfully transmitted from the client to the server.

10. The following TCP header dump is given in the hexadecimal format: CB5A00D3 00B41234 00003021 50100EB4 00500000 (give your answers in the decimal format). **(6 points)**

a. What is the source and destination port numbers?

Ans:

Here,

TCP datagram is CB5A00D3 00B41234 00003021 50100EB4 00500000
The first 16 bits (4 hexadecimal digits) are allocated to the source port number = CB5A
Convert CB5A to decimal = $CB5A_{16} = 52058_{10}$

So, the source port number in decimal format is 52058

The next 16 bits (4 hexadecimal digits) are allocated to the destination port number = 00D3
Convert 00D3 to decimal = $00D3_{16} = 211_{10}$

So, the destination port number in decimal format is 211

b. What is the sequence number?

Ans:

Here,

TCP datagram is CB5A00D3 00B41234 00003021 50100EB4 00500000
The next 32 bits (8 hexadecimal digits) are allocated to the sequence number = 00B41234
Convert 00B41234 to decimal = $00B41234_{16} = 11801140_{10}$

So, the sequence number in decimal format is 11801140

c. What is the acknowledgment number?
Ans:

Here,

TCP datagram is CB5A00D3 00B41234 00003021 50100EB4 00500000
The next 32 bits (8 hexadecimal digits) are allocated to the acknowledgement number = 00003021
Convert 00003021 to decimal = $00003021_{16} = 12321_{10}$

So, the acknowledgement number in decimal format is 12321

d. What is the header length?
Ans:

Here,

TCP datagram is CB5A00D3 00B41234 00003021 50100EB4 00500000
The next 4 bits (1 hexadecimal digits) are allocated to the header length number = 5
Convert 5 to decimal = $5_{16} = 5_{10}$ (5 * 4 = 20 bytes)

So, the header length in decimal format is 20 bytes (base header)

e. Which flags are set?
Ans:

Here,

TCP datagram is CB5A00D3 00B41234 00003021 50100EB4 00500000
The next 4 bits (1 hexadecimal digits) are reserved bits.
After that the next 6 bits are allocated to the flags = 50100EB4
Convert 50100EB4 to binary = $01010000000100000000111010110100_2$
From above conversion, the first 4 bits are Header length, the next 6 bits are reserved and next 6 bits represents the flags.

Flag bits = 010000

The relevant flags are: URG (1st bit), ACK (2nd bit), PSH (3rd bit), RST (4th bit), SYN (5th bit), FIN (6th bit)

So, the flags set are: ACK

f. What is the window size?

Ans:

Here,

TCP datagram is CB5A00D3 00B41234 00003021 50100EB4 00500000
The next 16 bits (4 hexadecimal digits) are allocated to the window size = 0EB4
Convert 0EB4 to decimal = $0EB4_{16} = 3764_{10}$

So, the window size in decimal format is 3764

11. A TCP client-server connection was established with the initial sequence number of $1003_{10.}$ The client sent 500_{10} bytes of data in the first segment. What is the sequence number of this segment and the range of the transferred bytes? What is the sequence number of the second segment sent by this client? **(3 points)**

Ans:

Given,

Initial sequence number = 1003
First Segment Sent by Client (500 bytes of data)

Here, the sequence number of the first segment is the initial sequence number.

And the range of transferred bytes is from the sequence number to the sequence number plus the number of bytes sent.

Sequence number of the first segment = 1003 + 1 = 1004
Range of transferred bytes = 1004 to (1004 + 500 - 1)
= 1004 to 1503

Sequence Number of the Second Segment:

Sequence number of the second segment = Sequence number of the first segment + Number of bytes sent in the first segment
= 1004 + 500
= 1504

Sequence number of the second segment = 1504

Therefore,

Sequence number of the first segment: 1004
Range of transferred bytes in the first segment: 1004 to 1503 bytes
Sequence number of the second segment: 1504

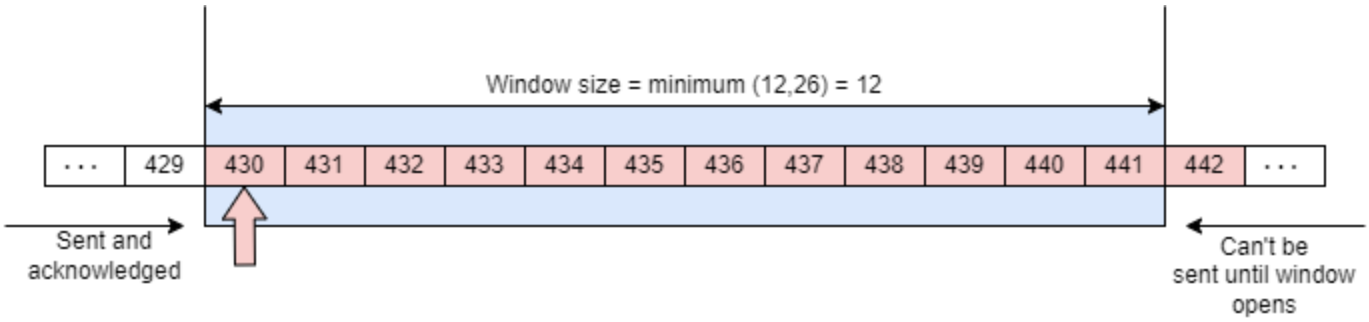
12. The current *cwnd*=12 and *rwnd*=26. The last acknowledgment received is 430. Draw the diagram showing this TCP window. A new TCP segment has just arrived with an acknowledgment number of 433 and *rwnd=x*. What is the minimum value of *x* to avoid shrinking the window? **(4 points)**

Ans:

Here,

Current cwnd=12 and rwnd=26
The last acknowledgement received is 430
Window size = minimum (cwnd, rwnd)
= min (12, 26)
= 12
Therefore, window size 12

- **Diagram showing this TCP window:**



Now,
New Acknowledgement is 433
To calculate the value of x the formula is as follows:

new rwnd > = lask ack + (min (cwnd, rwnd)) – new ack

Putting the given values in the above equation to get the value of x

x >= 430 + 12 - 433
x >= 430 - 421
x >= 9

The value of x is greater than 9

Therefore, X>=9

13. Is the size of the ARP packet fixed? Explain you answer. **(2 points)**
Ans:

The size of an ARP (Address Resolution Protocol) packet is not fixed; it can vary based on the specific parameters and information contained within the packet. ARP packets consist of several fields, and the size can change depending on the hardware and protocol addresses involved.

The lengths of the hardware and protocol addresses can vary, and the actual payload may include additional fields or options depending on the ARP type of packet (e.g., ARP Request, ARP Reply)

In summary, **the size of an ARP packet is not fixed** and depends on the specific network configuration and the types of addresses being used.

14. A host with an IP address 171.65.22.101 and a physical address A9:27:BB:F3:29:56 has a packet to send to a host in another network. The destination IP and physical addresses are 119.254.100.1 and AC:45:9D:E2:DD:67, respectively (this physical address is unknown to the sender). The next hop for this destination found in the sender's routing table is router R2 with an IP address 118.254.100.1 and a physical address AC:45:9C:52:66:B9 (this physical address is unknown to the sender). Show the ARP request and reply packets. Fill all the necessary fields. Ethernet and IPv4 protocols are implemented at the data link layer and the network layer, respectively. **(10 points)**

Ans:

ARP Request Packet:

<table>
<tr><td colspan="2">Hardware Type = 0x0001</td><td>Protocol Type = 0x0800</td></tr>
<tr><td>Hardware Length = 0x06</td><td>Protocol Length = 0x04</td><td>0x0001</td></tr>
<tr><td colspan="3">A9:27:BB:F3:29:56 = 0XA927BBF32956</td></tr>
<tr><td colspan="3">171.65.22.101 = 0XAB411665</td></tr>
<tr><td colspan="3">0x000000000000</td></tr>
<tr><td colspan="3">118.254.100.1 = 0x76FE6401</td></tr>
</table>

ARP Reply Packet:

<table>
<tr><td colspan="2">Hardware Type = 0x0001</td><td>Protocol Type = 0x0800</td></tr>
<tr><td>Hardware Length = 0x06</td><td>Protocol Length = 0x04</td><td>0x0002</td></tr>
<tr><td colspan="3">AC:45:9C:52:66:B9 = 0xAC459C5266B9</td></tr>
<tr><td colspan="3">118.254.100.1 = 0x76FE6401</td></tr>
<tr><td colspan="3">A9:27:BB:F3:29:56 = 0XA927BBF32956</td></tr>
<tr><td colspan="3">171:65:22:101 = 0xAB411665</td></tr>
</table>

15. What destination address is used in the Ethernet frame carrying an ARP request? Explain your answer. **(2 points)**

Ans:
The destination addresses of an Ethernet frame carrying an ARP (Address Resolution Protocol) request is usually a broadcast address. To guarantee that every device in the local network segment receives the ARP request we need to use this broadcast address. In hexadecimal, the broadcast address is FF:FF:FF:FF:FF:FF. It is a unique MAC address that is set to all binary ones.

The device sending the ARP request sends it to every device on the local network since it is unsure of the MAC address (Physical address) it needs to look for. Additionally, sending out a broadcast for the ARP request guarantees that the device with the matching IP address will receive it and reply with its MAC address (Physical Address).

Therefore, in an Ethernet frame carrying an ARP request, the destination MAC address (Physical Address) is set to the broadcast address FF:FF:FF:FF:FF:FF.

16. Router R1 has received an ARP request. Can this ARP packet be used to update a cache table of R1? Explain your answer. Note that R1 has received an ARP request not an ARP reply. **(4 points)**

Ans:

No. Because, as there is no physical mac address available as of now a router (such as R1) would not normally use an ARP request to update its cache table. Within a local network, ARP (Address Resolution Protocol) requests are exchanged to resolve the mapping of an IP address to a MAC address. On the local network, an ARP request is sent out to find the MAC address that goes with a specific IP address.

On the other hand, routers normally don't take part in the local network's ARP resolution directly. Devices on the same network use ARP to find each other's MAC addresses or the physical address. When routing packets between various subnets, routers use IP addresses rather than MAC addresses to determine which packets to forward. Routers function at the network layer.

When a router receives an ARP request, it may process the request by forwarding it to the appropriate subnet or dropping it if the request is not relevant to the router. The router itself does not typically update its cache table based on ARP requests; instead, it may update its ARP cache table based on ARP replies.

Upon receiving an ARP request, a router has the option to treat it by either forwarding it to the correct subnet or discarding it if it is deemed irrelevant to the router. The router itself may update its ARP cache table based on ARP answers rather than usually updating its cache table based on ARP queries.

In conclusion, routers typically do not use ARP requests to update their cache tables. In contrast, ARP replies give local network devices the information they need to update their ARP caches.