# FCS MIDSEM

**Question 1:**

**a.**

Anonymized Table for 2-Anonymity:

# Assumed for this table that none of the attributes are sensitive

| Customer ID | Name | Place | City | Country | No of Items Purchased (N) | Price |
|---|---|---|---|---|---|---|
| C000** | * | * | * | * | 1<=N<=5 | [4000.00, 12000.00] |
| C000** | * | New York | New York | USA | 2<=N<=5 | [2000.00, 6000.00] |
| C000** | * | New York | New York | USA | 2<=N<=5 | [2000.00, 6000.00] |
| C000** | * | * | * | * | 1<=N<=5 | [4000.00, 12000.00] |
| C000** | * | * | * | * | 1<=N<=5 | [4000.00, 12000.00] |
| C000** | * | New York | New York | USA | 2<=N<=5 | [2000.00, 6000.00] |
| C000** | * | Brisban | Brisban | Australia | 1<=N<=3 | 7000.00 |
| C000** | * | Brisban | Brisban | Australia | 1<=N<=3 | 7000.00 |
| C000** | * | Chennai | Chennai | India | 1 | [4000.00, 10000.00] |
| C000** | * | Mumbai | Mumbai | India | 1<=N<=3 | 7000.00 |
| C000** | * | Chennai | Chennai | India | 1 | [4000.00, 10000.00] |
| C000** | * | Mumbai | Mumbai | India | 1<=N<=3 | 7000.00 |

# Assuming that ‘Price’ attribute is a sensitive attribute

| Customer ID | Name | Place | City | Country | No of Items Purchased (N) | Price |
|---|---|---|---|---|---|---|
| C000** | * | * | * | * | 1<=N<=5 | 6000.00 |
| C000** | * | New York | New York | USA | 2<=N<=5 | 3000.00 |
| C000** | * | New York | New York | USA | 2<=N<=5 | 5000.00 |
| C000** | * | * | * | * | 1<=N<=5 | 5000.00 |
| C000** | * | * | * | * | 1<=N<=5 | 10000.00 |
| C000** | * | New York | New York | USA | 2<=N<=5 | 5000.00 |
| C000** | * | Brisban | Brisban | Australia | 1<=N<=3 | 7000.00 |
| C000** | * | Brisban | Brisban | Australia | 1<=N<=3 | 7000.00 |
| C000** | * | Chennai | Chennai | India | 1 | 8000.00 |
| C000** | * | Mumbai | Mumbai | India | 1<=N<=3 | 7000.00 |
| C000** | * | Chennai | Chennai | India | 1 | 7000.00 |
| C000** | * | Mumbai | Mumbai | India | 1<=N<=3 | 7000.00 |

Anonymized Table for 3-Anonymity:
# Assumed for this table that none of the attributes are sensitive

| Customer ID | Name | Place | City | Country | No of Items Purchased (N) | Price |
|---|---|---|---|---|---|---|
| C000** | * | * | * | * | 1<=N<=5 | [4000.00, 12000.00] |
| C000** | * | New York | New York | USA | 2<=N<=5 | [2000.00, 6000.00] |
| C000** | * | New York | New York | USA | 2<=N<=5 | [2000.00, 6000.00] |
| C000** | * | * | * | India | 1<=N<=3 | [4000.00, 10000.00] |
| C000** | * | * | * | * | 1<=N<=5 | [4000.00, 12000.00] |
| C000** | * | New York | New York | USA | 2<=N<=5 | [2000.00, 6000.00] |
| C000** | * | * | * | * | 1<=N<=5 | [4000.00, 12000.00] |
| C000** | * | * | * | * | 1<=N<=5 | [4000.00, 12000.00] |
| C000** | * | * | * | India | 1<=N<=3 | [4000.00, 10000.00] |
| C000** | * | * | * | India | 1<=N<=3 | [4000.00, 10000.00] |
| C000** | * | * | * | India | 1<=N<=3 | [4000.00, 10000.00] |
| C000** | * | * | * | India | 1<=N<=3 | [4000.00, 10000.00] |

# Assumed for this table that ‘Price’ attribute is sensitive

| Customer ID | Name | Place | City | Country | No of Items Purchased (N) | Price |
|---|---|---|---|---|---|---|
| C000** | * | * | * | * | 1<=N<=5 | 6000.00 |
| C000** | * | New York | New York | USA | 2<=N<=5 | 3000.00 |
| C000** | * | New York | New York | USA | 2<=N<=5 | 5000.00 |
| C000** | * | * | * | India | 1<=N<=3 | 5000.00 |
| C000** | * | * | * | * | 1<=N<=5 | 10000.00 |
| C000** | * | New York | New York | USA | 2<=N<=5 | 5000.00 |
| C000** | * | * | * | * | 1<=N<=5 | 7000.00 |
| C000** | * | * | * | * | 1<=N<=5 | 7000.00 |
| C000** | * | * | * | India | 1<=N<=3 | 8000.00 |
| C000** | * | * | * | India | 1<=N<=3 | 7000.00 |
| C000** | * | * | * | India | 1<=N<=3 | 7000.00 |
| C000** | * | * | * | India | 1<=N<=3 | 7000.00 |

(In all the tables made for this question, the Customer ID and Name do not reveal any sort of information at all as they are replaced with *. These are PII’s, i.e. Personally Identifiable Information. In the tables above it is as good as these columns being dropped)

**b.**

Techniques to increase utility of anonymized data:

- The value of ‘K’ in K-anonymization should be kept lower. This will ensure more specific information for analysis. For lower values of K, even K-anonymized data may demonstrate a lot of information which is of utility.

- In the K-anonymity table the organization shares the data with some other organization, they can group the data into sensitive and insensitive information. This is implemented in part a. of the question.

The sensitive columns are kept as it is, unchanged and K-anonymization is applied to the insensitive columns. The sensitive columns are chosen keeping in mind what the organization with whom the data is being shared would need to analyze for the data to be useful for them. Eg: A car insurance firm, buying data from let's say, Google will be interested in a person's average speed while using Google Maps while he is driving. So Google can make this column sensitive when sharing data with the car insurance firm. Other records such as number google searches or number of emails received in a day might not be useful for the car insurance firm. Such attributes should be made insensitive.
This non-suppression and non-generalization of sensitive data increases the utility of data for the firm who is procuring data.

- While anonymizing data, it can voluntarily be ensured that the statistical importance (such as mean, standard deviation, range) of certain attributes are not compromised while anonymizing data. It should also be noted that these values should be generalized such that relevant statistical calculations for certain columns can be done and their values do not drastically differ from statistical parameters of actual data.

- Anonymized data should be collected from multiple sources. It should be ensured that there is a significant overlap of users of these sources. Doing so may reveal a lot more information than data of sources with a completely non-overlapping user base. It is because data from these sources can be combined and then analyzed carefully. (Cross referencing of databases → Homogeneity Attack).

- For data analysis, anonymization can also be done by changing certain values such that some data becomes useful (particularly, for statistical analysis) but is not linked to any existing entity.

**c.**

Dummy Data:

| | | | | |
|---|---|---|---|---|
| Rahul Kumar | 2018 | Design | 7 | 10 |
| Arjun Singh | 2019 | Computer Science | 9 | 7 |
| Md Shanu | 2014 | Electronics | 5 | 8 |
| Radhika Jain | 2013 | Electronics | 6 | 8 |
| Arvind | 2013 | Computer | 8 | 5 |

| | | | | |
|---|---|---|---|---|
| Sharma | | Science | | |
| Neha Kumar | 2017 | Mathematics | 3 | 2 |
| Harsh Verma | 2016 | Computer Science | 7 | 7 |
| Aditya Joshi | 2019 | Computer Science | 10 | 8 |
| | | | | |
| | | **AVERAGE** | 7 | 7 |
| | | | | |
| | | **STD DEV** | 2.121320344 | 2.291287847 |

Methods Adopted:

GANs- Generative Adversarial Networks

The data of the participants is modified in such a way that average and standard deviation of the columns Academic Experience- Rating and Extracurricular Experience-Rating differ very minutely from the actual data. The other values such as batch, department, names are replaced with values that do not exist in real data. All the attributes are completely changed.

Reason for choosing this:

According to the GDPR, the ownership of data is not with the company collecting the data but with the users' whose data is collected.

Keeping this in mind, I will have to modify the data completely such that I as collector of data become the owner of the data. So I remove the users' details completely by synthesizing data such that the only similar thing between the users' data and the data I generated is the statistical parameters of a few important columns. This way, I am not really storing users' data, but rather the essence of data as synthetic data, without compromising data ownership and privacy rules of GDPR.

The synthetic data table (Table after applying above mentioned technique):

| Name | Batch | Department | Academic Experience- Rating (0-10) | Extracurricular Experience-Rating (0-10) |
|---|---|---|---|---|
| Shashwat Jain | 2014 | Biology | 8 | 7 |
| Riya Sharma | 2019 | Computer Science | 5.5 | 8 |
| Ananya | 2020 | Design | 7 | 8 |

| | | | | |
|---|---|---|---|---|
| Singhal | | | | |
| Vatsal Narula | 2012 | Mathematics | 4 | 10 |
| Akshay Kaushal | 2016 | Design | 9.5 | 8 |
| Robert Hales | 2015 | Electronics | 4 | 2 |
| C Henry | 2015 | Computer Science | 10 | 5.5 |
| Daniyal Khan | 2020 | Biology | 8 | 8.5 |
| Jasleen Kaur | 2014 | Design | 7 | 6 |
| | | | | |
| | | **AVERAGE** | 7 | 7 |
| | | | | |
| | | **STD DEV** | 2.165063509 | 2.304886114 |

**Question 2:**

**a.**

IPv6 overcomes the limitations of IPv4 in the following ways:

- Unique to devices: IPv6 is a 128 bit addressing format. Due to this large size, it can be used to identify the devices uniquely. IPv4 is 32 bit format and hence it is not possible to assign a unique IPv4 to all devices. With this NAT will not be required.
- The header format of IPv6 enables faster processing and forwarding at nodes. Hence data delivery is faster using IPv6 over IPv4.
- Due to its limited header size, IPv4 allowed only 64kB of data to be transferred via a single packet. IPv6 allows upto 4GB of data to be delivered by a single packet.
- Quality of Service: The header format of IPv6 facilitates QoS
    - It can identify priority among datagrams in a flow (ToS bit).
    - It can identify datagrams in the same flow.
- Next Header Field: Allows options for faster data transfer.
- Removal of Header Checksum: This ensures faster processing at each node.
- Security: Mandatory use of IPsec in IPv6.

IPv6 is not widely accepted and used for communication because many of the devices in network architecture are old and do not have support for IPv6. These are not only end systems but also several core network devices. Thus to provide backward compatibility

to each device we are restricted to shift to IPv6. If we do, the network core devices, not supporting IPv6 would become useless thus affecting performance of the internet as whole and other end systems, not supporting IPv6 won't be able to communicate over the internet.

Yes, IPv6 is more secure than IPv4 because when IPv6 was launched it was mandatory to use IPsec. However, it is no longer mandatory now, because of backward compatibility issues, but it is recommended and can easily be incorporated when using IPv6. This makes the vulnerable Network Layer (IP protocol) secured and hence the IPs cannot be tampered with, during transmission of packets.

**b.**

IPsec (Internet Protocol Security) allows the IP addresses in the packets to be validated and authenticated. It is implemented at the network layer. This ensures that the sender and receiver information (IP addresses) are not tampered with during data transfer. The primary goals of IPsec are:

- To verify sources of IP packets
- To prevent replaying of old packets
- To protect integrity and confidentiality of packets

IPsec sets up an encrypted channel of communication upto the network layer.

Yes, IPsec can be implemented (although optionally) with IPv4.

**c.**

Prevention of packet flooding attacks:

- Incorporate IPsec. It allows control packet replaying attacks as the IP addresses are authenticated and hence cannot be tampered with. This ensures that communication happens only between 2 legitimate parties whose identities are authenticated.

Some other ways could be as follows. All the below points can be implemented via 'Stateful Inspection Configuration of Firewall' in the network. This would also ensure lesser overhead.

- When there is a lot of traffic, much larger than usual, then the network should just slow down in creating further sessions for communication until it detects that the packet flow in the network is normalized. This is a very rudimentary idea and can be overcome by a proactive attacker.
- There can be implemented a system of checks to see whether there are too many requests from a particular IP address. In this case, after a certain threshold, their requests could be completely blocked.
- There could also be a check on whether a lot of junk packets (carrying repetitive information) are being sent into the network. We can prevent such a thing from continuing after a certain number of such packets are detected.

**d.**

A secure communication channel can be set up using IPsec. This would ensure that the IP addresses are validated and authenticated. Hence all the devices would be able to trust each other that they are actually the one's sending data and not someone from outside the network is impersonating them.

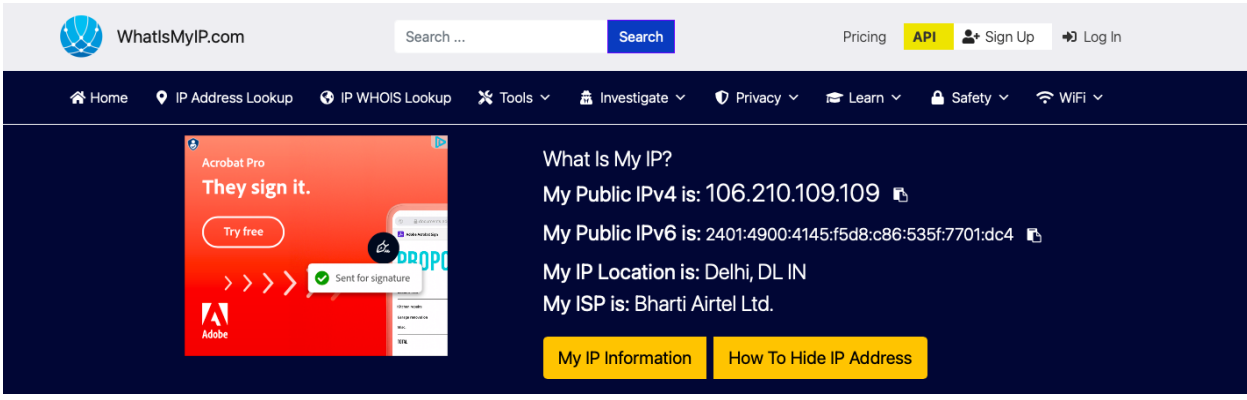
For key exchange, the idea of IKE (Internet Key Exchange) of IPsec can be implemented. Since there are a limited and countable number of devices (deduced from the image in question paper) within the network, devices can maintain a directory of Master Keys (Phase 1 of IKE) for communication with other devices inside the network. This Master Key can be set up securely using asymmetric cryptographic techniques such as RSA. Further for each session of communication, again a key for that particular session can be obtained using asymmetric cryptography. This key is obtained with the help of the Master Key obtained before. After this a secure communication channel will be set up and they can exchange data using Diffie Hellman Algorithm, which is a symmetric cryptography algorithm, hence requires much lesser computation.

There should also be an implementation of firewall for communication from outside the network over the internet.

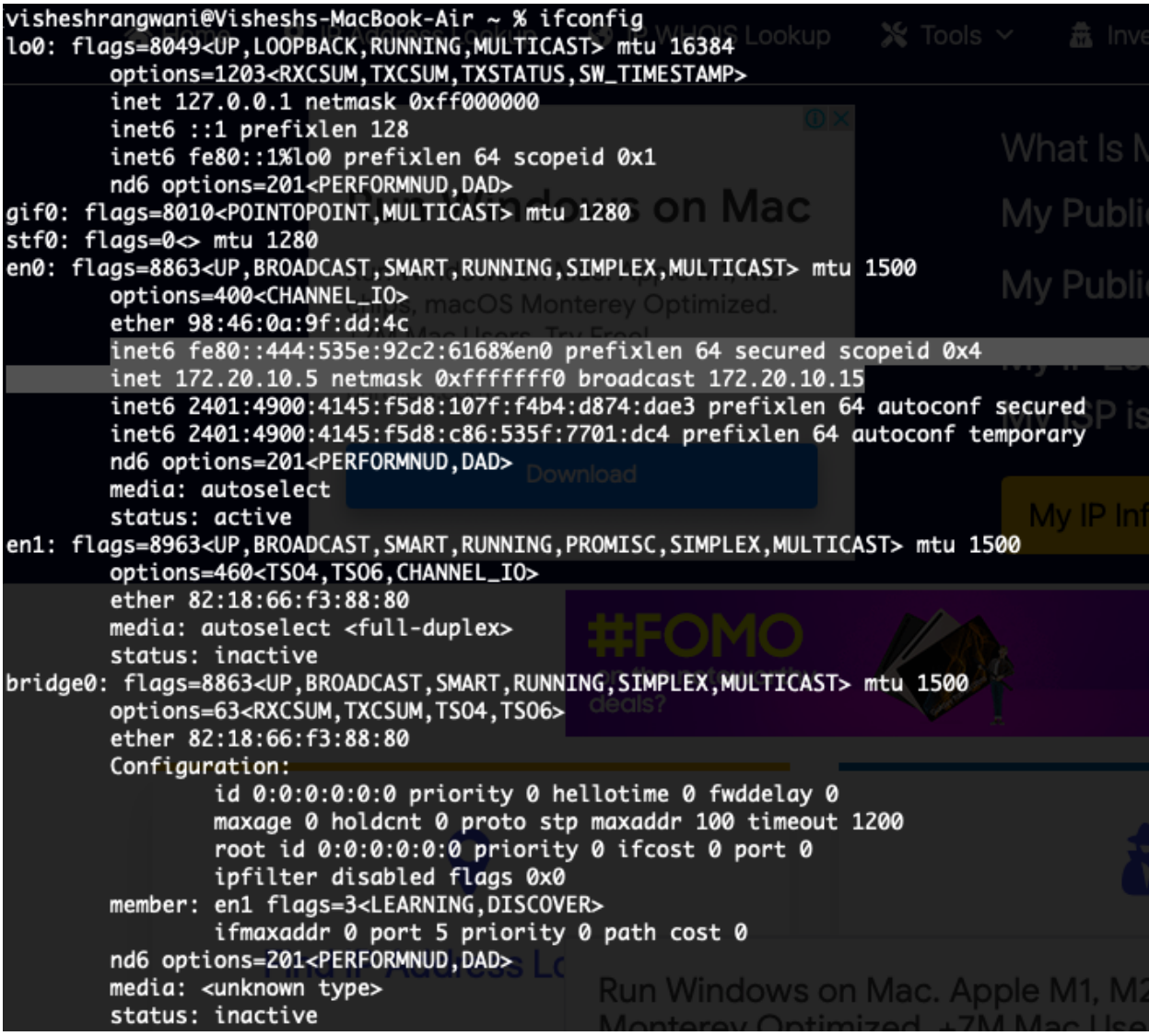
**e.**

After connecting my laptop to a mobile hotspot, I checked the IP on whatismyip.com for public IP and on the terminal using the ifconfig command for private IP. In both the interfaces, I observed a different IP than the one I see when connected on WiFI.

Below is the screenshot of whatismyip.com interface:



Here is a screenshot of terminal (IP is highlighted for interface en0)




I can observe both IPv4 and IPv6 addresses.

Advantages of IPv6: Can support a unique IP for each device, is faster in processing and forwarding and is more secure because of IPsec. (Mentioned in detail in part a.)

Disadvantages of IPv6: Many older devices do not support it (not backward compatible) hence difficult to implement.

Advantages of IPv4: Supported by all kinds of devices, even legacy systems hence no issue with regard to backward compatibility. We do not have to care before using IPv4 whether it will be supported or not.

Disadvantages of IPv4:

- Has become exhaustive because of the increasing number of devices. Hence not unique for each device.
- Can support just 64kB packets.
- Its header provides limited information.
- Slower processing and forwarding.

**Question 3:**

**a.**

There are quite a few security concerns in the shown three-message authentication shown in the question:

- 'Man In the Middle' attack. When Alice initiates communication that initiation response can be intercepted by an attacker in the middle. The attacker can impersonate Bob and send the challenge to Alice and so on. After authentication, Alice may share data thinking that the data is being sent to Bob but it would be sent to the attacker. There is no way to ensure that Alice is communicating with Bob only.
  This impersonation is not only valid for Bob but for Alice as well. An attacker may impersonate Alice and may communicate with Bob. This can be done only if the challenge is a weak security measure only such as a captcha and not something required for strong authentication (what entity has; what entity knows; what entity is; where entity is). The attacker would send back the challenge and its response. Another possible attack is possible if an attacker intercepts the third message. He would sit in between Alice and Bob and observe their communication.
  Concisely speaking, there is no way to authenticate Bob's and Alice's identity.

- The handshake is a challenge response type of authentication mechanism. It is an unsafe form of handshake, if used in isolation as the entire information of mutual authentication handshake, both the challenge as well as response is shared in the third message. Not only is the encryption breakable, it can even be falsely authenticated by a person spoofing Bob.

**b.**

Design of R, keeping in mind different considerations:

- Considering that the authentication system should strictly follow the protocol in question:
  R should be any of the following. For more strong authentication, there should be at least 2 of the following four:
  - ➢ What is the entity?
  - ➢ What does the entity know?
  - ➢ What entity has?
  - ➢ Where the entity is?
- For a system related to One Time Passwords:

Instead of using the Challenge Response mechanism OTP, if the Hash Chain mechanism of OTP is used, then Alice will have to share just one piece of information as response and not the challenge as well as the response. In such a case the change in protocol would be that Bob won't be able to remain stateless and will have to store for what value's hash, the message has been shared as OTP.
One way I could think of making Bob stateless is by introducing some sort of sequence numbers that are shared along with OTP to Alice and the level of chain at which hashing is to be checked. These 2 things along with hashed and encrypted OTP can be shared back to Bob.

- Certificate Authorities (CA):
  Introducing them will clearly modify the protocol quite a bit.
  Alice will have to authenticate herself by asymmetric key cryptography by a trusted CA. Similarly, Bob will also be authenticated by a CA. The handshake will be done as follows: Alice will get her packet digitally signed by her CA using CA's private key. Bob will decrypt using CAs public key as mentioned in Digital Certificate of Alice. Then he will get his handshaking packet signed by his CA's private key. Alice will decrypt that. Both will be authenticated and will decide upon a shared key for communication during that session.

**c.**

If Alice and Bob use SSL/TLS protocol, Man in the Middle attacks won't be possible as long as the private key of CA remains safe. Assuming that the private key of CA is safe, SSL/TLS authenticates both parties and secure the channel of communication using asymmetric cryptography. Thus launching a Man In the Middle attack is as good as impossible.
However, spoofing attacks are possible because SSL/TLS works at the Presentation Layer of the OSI stack. In case IPsec is not used, the TCP handshake can be impersonated by an attacker by predicting the sequence number and an attacker communicating with Bob as Alice or vice-versa. Also, at some gateway of a communication channel, the IPs of packets at lower layers (especially at an unsecured network layer) can be modified.

**d.**

If the server side of communication (here, Bob) has a verifiable certificate, then authentication can be accomplished. This happens as follows:
When Alice initiates the communication with Bob, he will encrypt the handshake data by his own private key, then with Alice's public key and then share it to the CA. CA will digitally sign this with his own private key and send it to Alice. Alice would decrypt first by public key of CA, ensuring that authentication of Bob by CA has not been tampered. Then it will decrypt with its own private key, ensuring that the receiver is correct and then decrypting with Bob's private key, thus confirming that sender is Bob. This ensures

authentication of Bob and Alice as they both use each other's public keys and their own private keys. All this is digitally signed by trusted CA.

This entire process is completely secured due to presence asymmetric cryptography all validated by a trusted CA.
A few points that are considered in above explanation;

- Data encrypted by X's private key can only be decrypted by X's public key and vice versa.
- During handshake a symmetric key is shared. After the handshake is done, this symmetric key is used to share further data.

**Question 4:**

**a.**

In the given scenario, Capabilities would be a better option to manage access privileges.
Reason: Capabilities specify a list for each user. The list contains the files the user can have access to along with the privileges granted to the user. In the question, it is mentioned that the users are transient whereas the files are persistent in a system. So whenever there has to be a change in privileges or deletion or insertion of a user, then using capabilities techniques, we can simply modify the privileges, delete the user or simply insert a new user without the hassle of looking into all users' lists and then finding a particular file. Stating simply, it requires O(1) operation for inserting and deleting a user whereas O(n) operations for changing the access privileges of a user. On the other hand, ACL specifies a list for each file. Each list has users along with their privileges for that file. For deletion of a user, we will have to traverse all the lists and see where all to delete the user. This will require O(m*n) operations. Similarly for modifying privileges, we'll have to search for the user in all the lists and then change its privileges; which is again O(m*n) operations. Insertion also would require traversing all lists and adding the user in them; which is at least O(m) operations.
Due to these reasons, in case of transient users and persistent files, it would be better to use the technique of capabilities.

**b.**

Yes, an unauthorized user can misuse the privileges of Alice for file X and read the contents or even write in the file. The term for such a malicious attack is 'Trojan Horse'. The description for the same is as follows:
Alice has write privileges on X. Suppose there is some attacker who while Alice is writing on file X is able to infect X with malware called Trojan Horse. Trojan Horse is a rogue software that does what the user is expected to do, but exploits the legitimate user's privilege to cause a security breach. In this case, the trojan horse could copy some data from file X to another file, say Y on which Alice has write access. This is obviously not desirable as anyone having read access on Y but not on X (In this case Bob), will be able to read the contents of file X. This could potentially be a major security flaw.

(# for the above part's explanation, it has been assumed that all permissions must be specified explicitly, i.e. even Owner of file must have explicit permission to read, write or execute)

**c.**

Here are a few options to prevent this Trojan Horse attack:

- A naive solution is to use some service such as Anti-virus softwares that detect Trojan Horses and warn the owner or remove them.

- A more technical oriented solution is to use the 'Bell-LaPadula' security mode for access control. In this model, there are 2 simple security propositions:
  - No read-up
  - No write-down

  The 4 classifications, in order of their hierarchy are:
  - Top secret
  - Secret
  - Confidential
  - Unclassified

  In our scenario, Alice can be considered to have a higher classification than Bob. Also file X will have higher classification than file Y. (X will be classified the same as Alice but as object and Y can be classified same as Bob) Hence using Alice's privileges, a malicious software won't be able to write contents of X to Y as writing down isn't permitted. Similarly, Bob won't be able to read contents of X as read up is not allowed.

- There could also be a provision for Discretionary Access Control (DAC) and Mandatory Access Control (MAC).
  DAC is based on the identity of users or groups of users and the stricter restrictions that can be imposed on them.
  MAC is more secure than DAC and an extension of DAC. Hence it is more recommended. It puts a restriction on access to an object based on the sensitivity of the information held by that particular object. It also requires a formal authorization for a user to access information of such sensitivity.

**Question 5:**

**a.**

"Perfect Anonymity" for me is being completely anonymous irrespective of the surroundings and how other people behave. To explain, I think, I would be completely anonymous on the Internet if there is no possible way, not even an impractical, very hard and condition-dependent way which can lead to my identity being revealed in any form. Since such a thing is not possible, as discussed in lecture, I don't think that "Perfect Anonymity" is possible.

- Clearly, when we are communicating normally on the web our IP addresses are available to all the routers and hosts through which our requests and responses are routed. This isn't anonymous at all and can be exploited by many agencies.
- When we are using VPN, now our IPs are not visible to the hosts in the route. However, the VPN provider would always know what requests are from which IPs are masked by the VPN. We are anonymous to the Internet by and large as long as our VPN provider lets us be. We are not anonymous to the VPN provider.
- Tor: It is one of the tools that guarantees our anonymity. But technically, there are a few conditions that need to be met for our anonymity to be maintained. This conditional anonymity is sufficient for almost all practical purposes but not a 100% certainty because of 2 things in Tor's architectural design:
    - Traffic in Large volume and many nodes: We are anonymous in Tor because the number of nodes in the Tor networks is very large. Due to this it is difficult to trace the data. Tracing of data becomes even more difficult because of voluminous traffic on Tor. Hence, anonymity on Tor is there as long as the number of nodes and users are large.
    - Importance of Entry Nodes: Entry nodes are the only nodes who have our IP addresses. Although there is a strict check and restrictions on who can be an entry node, by Tor architecture. Still, there can be a very minute chance that our anonymity is compromised because of entry node's carelessness or malicious behavior.

**b.**

Principles and Techniques used by tools like Tor to maintain anonymity:

- Tor is a circuit based overlay network.
- Each client and node in the Tor network have access to a directory of all Tor nodes with their public keys. The Initiator client, before sending out requests selects at least 3 nodes for the path of the data flow. These should preferably be in different judicial boundaries.
- After selecting the nodes, it encrypts the data with HTTPS. The IPs of each node are encrypted along with the data in an order which is opposite the flow of data, i.e. The IP of the entry node is encrypted at last.
- After this it sends a packet to the entry node (also called Guard node). Entry node is a restrictive node and a carefully chosen one by the Tor node as it is the only node having IP of the Initiator. Not every node can be an entry node. Entry node decrypts the packet using its private key and is able to access IP of the 2nd node. It forwards the packet to the 2nd node.
- All nodes except the entry and exit nodes are intermediate nodes. Intermediate nodes decrypt using their private keys to see the IP of the next router in sequence. This decryption is done layer by layer.
- When data reaches the exit node, it is again a sensitive node, but not as restrictive as the entry node. Exit node forwards data packet to the receiver.

- The exit node doesn't know the exact IP of the receiver (servers providing content on Tor network) as they are layered behind several Proxy server layers. This is helpful in hiding their identity.
- Even the public links and the IPs of websites are not static and keep changing at regular intervals.
- Monetary Transactions in Tor happen using cryptocurrencies.
- Each node has just 1 hop information, i.e. information about who sent data to them and to whom they have to send their packet.
- It is extremely difficult for a node to trace packets as the number of nodes and number of users in the Tor network are huge. This way, it becomes nearly impossible for a usual attacker on the internet to actually trace packets on Tor. The large numbers are Tor networks biggest strengths in maintaining anonymity.

**c.**

Yes anonymity on Tor can be compromised. A few scenarios could be:

- If exit and entry nodes somehow collide. This way the node will know who is sending data to whom, with the exact IP of the client.
- DNS leaks: If a client does not use Tor DNS but rather uses the general DNS of the web, then his anonymity is compromised to the DNS and can be traced back.
- Traffic Analysis: An attacker can analyze certain packets on Tor nodes, which are either setup by the attacker or have access over them. By analyzing he can try and trace the packets' hops from/to the end host.
- Malicious exit nodes: These nodes may be exploited to reveal the identity of the website they send data to in case the websites are not protecting themselves by proper use of Proxy servers.

<u>Exploiting Vulnerabilities:</u>

- A malicious person or an organization who has access over multiple tor nodes can observe the metadata of the packets. He can vigorously inject a pattern into the packets going through those nodes. During communication, he can observe his nodes and see which packets have that injected pattern. By a bit of analysis after studying the Tor network related to his nodes vigorously, he can actually trace the packets either to their source or destination or even both, depending on if the packet makes several hops on nodes under his control.

- Another way to exploit DNS vulnerabilities could be to observe UDP packets at a web cache. Since web cache stores the IP upto 'ttl' (time to live), we can check the packets whether any of them may contain DNS mapping of an onion site. If there is, the users' anonymity is compromised.

<u>Attempt for Bonus Part:</u>

To conduct a zero day exploit, I would combine the ideas of two vulnerabilities. First of all host a lucrative onion site on dark web, something that will be used often to provide information to users for what they search commonly on dark web. Maybe a clone and something similar to the extended wiki. When sending data packets back, I can add a Trojan Horse to the packets. The malicious job of this would be to modify the next UDP packet, which is going to the Tor DNS for resolution. It should modify in such a way that instead of going to Tor DNS, it goes to the DNS, whose IP I would want the Trojan to write. I will choose a DNS, whose incoming packet I can observe. This can be done by injecting some sort of pattern which I can observe as UDP packets are not secured. I would keep in mind that checksum is unable to detect that pattern. After this I would have exploited the DNS vulnerability using ideas from Trojan Horse concept taught and also the Traffic Analysis vulnerability.