FCS Mid-Semester Exam

Name: Meetakshi Setiya
Roll no: 2019253

Question 1.

a)
I have not anonymized ‘No. of items purchased’ and ‘Price’ because I believe they are sensitive information. The 3-anonymised table is:

Customer ID	Name	Place	City	Country	No items purchased	Price
'C000**'	*	*	*	*	'2'	'6000.00'
'C000**'	*	'New York'	'New York'	'USA'	'2'	'3000.00'
'C000**'	*	'New York'	'New York'	'USA'	'3'	'5000.00'
'C000**'	*	*	*	'India'	'2'	'5000.00'
'C000**'	*	*	*	*	'1'	'10000.00'
'C000**'	*	'New York'	'New York'	'USA'	'3'	'5000.00'
'C000**'	*	*	*	*'	'2'	'7000.00'
'C000**'	*	*	*	*	'1'	'7000.00'

'C000**'	*	*	*	'India'	'1'	'8000.00'
'C000**'	*	*	*	'India'	'1'	'7000.00'
'C000**'	*	*	*	'India'	'1'	'7000.00'
'C000**'	*	*	*	'India'	'2'	'7000.00'

b)
Dummy data:

Email Address	**Participant Smokes or Not**
irina18442@hotmail.com	Yes
raman@gmail.com	Yes
aadi18888@iiitd.ac.in	Yes
asna17777@iiitd.ac.in	No
vasilisa@gmail.com	No
raman@yahoo.com	Yes

- I would use k-anonymity to obscure the email addresses of the participants to protect their privacy.
- Using K-anonymity to obscure and anonymize the email addresses would reduce their specificity. Any effort to single out an individual from the database would narrow down the search to at least k individuals (courtesy of generalisation and suppression).

Here is an example of 2-Anonymity applied to the above table:

Email Address	**Participant Smokes or Not**
*	Yes

raman@*	Yes
*@iiitd.ac.in	Yes
*@iiitd.ac.in	No
*	No
raman@*	Yes

Question 2.

It indeed is absolutely necessary to follow the standard practices put in place by the governing authorities for the payment gateway of the Amazon-like e-commerce application. These practices per my understanding are:

1. It is absolutely mandatory that sensitive information like credit and debit card details must be securely encrypted. This has to be done to prevent leakage or unauthorized access to the data. The encryption keys must be strong and secret.
2. The website must have a Secure Sockets Layer(SSL) certificate to authenticate its identity and to protect every transaction that would take place on it. SSL creates an encrypted channel to send sensitive data by encrypting it in transit and preventing it from being hacked. This ensures a secure service for both the customer and the website (merchant).
3. Encryption must be point-to-point.
4. Sensitive customer information like PIN, bank card numbers etc must not be stored on the server (database).
5. If the payment method data has to be stored, it should be stored securely in a special server database. The data must be encrypted before reaching the database.
6. The payment gateway must ask for CVV to ensure that the bank card used in the transaction is actually with its owner. Similarly while displaying an already stored account number or card number, the digits must be masked and only the last 4-5 digits must be shown.
7. Input fields that ask for secret information like CVV must be hidden behind asterisks or some other characters so that it is non-discernible.
8. Use a firewall to secure the e-commerce web application.
9. The payment gateway must have the bank verify the transaction with the user by asking for confirmation using an OTP before processing the payment. This provides an additional layer of security and prevents fraudulent transactions.

10. The website server must look out for DDoS attacks that could prevent legitimate users from using the payment gateway. Firewall could be used to block the malicious traffic.
11. The payment gateway must be compliant of the guidelines and stated in the Payment Card Industry Data Security Standard (PCI DSS). Legitimacy can be achieved if the payment gateway offers PCI compliant services.
12. Conduct regular vulnerability scans to ensure that new vulnerabilities have not popped up. If there are vulnerabilities, third party services can be used to plug them.
13. If a third party service provider is used anywhere in the process, ensure that they comply with the website's security policies. The third party should be trustworthy and not malicious.
14. Maintain confidentiality of the customer's payment/billing information. Limit the user credentials so that even if an attacker obtains a legitimate user account, they can do only minimal damage.

Question 3.

a) encrypt(concat(m, hash(m)), public_alice)

Bob cannot decrypt the message.

1. Confidentiality: The message was encrypted with Alice's public key and can be decrypted only by Alice's private key. Even though Bob cannot decrypt this message, there is no unauthorized information gathering possible and confidentiality is maintained.

2. Non-repudiation: The communication carries no proof that Alice is the one sending the message to Bob. Also, there is a scope that Alice can successfully dispute her authorship of the message. Hence, non-repudiation is not possible here.

3. Steps to decrypt the message: cannot be decrypted.

b) encrypt(concat(m, hash(m)), public_bob)

Bob can decrypt the message.

1. Confidentiality: The message was encrypted using Bob's public key and can be decrypted only by Bob's private key. Even if someone intercepts the message, there is no way that anyone besides Bob can decrypt it. Hence, no unauthorized information gathering is possible and confidentiality is maintained.

2. Non-repudiation: The communication carries no proof that Alice is the one sending the message to Bob. Also, there is a scope that Alice can successfully dispute her authorship of the message. Hence, non-repudiation is not possible here.

3. Steps taken by Bob to decrypt the message:
    - ➢ Use his private key for decryption of the cipher text and obtain the string concat(m, hash(m)).
    - ➢ Use the function that splits a concatenated message perfectly, to unambiguously identify the message "m" and its hash.
    - ➢ (Optional) Hash the message "m" and compare that with the received hash to check if the message has not been tampered with.

c) encrypt(m, public_bob), sign(hash(m), private_alice)

Bob can decrypt the message.

1. Confidentiality: The message was encrypted using Bob's public key and can be decrypted only by Bob's private key. Even if someone intercepts the message, there is no way that anyone besides Bob can decrypt it. Hence, no unauthorized information gathering is possible and confidentiality is maintained

2. Non-repudiation: Alice has signed a hash of the message with her private key. When Bob obtains the signature, he can use Alice's public key to obtain the hashed message and cross check it with the decrypted message. Since signatures use private keys, it is extremely likely that Alice is the one who sent the message since the hashed message is signed with her private key and it can be established that the public key is legitimately hers. The message was signed using Alice's private key and anyone who has access to her public key can trace it back to her. This is how non-repudiation is possible here.

3. Steps taken by Bob to decrypt the message:
    - ➢ Use his private key for decryption. Message "m" is thus obtained.

- ➢ (Optional- Further verification to ensure the message has not been tampered). Use Alice's public key to get the hash(m) from the signature.
- ➢ Hash the message "m" and compare it with the obtained hash. If they are equal, the message is untampered. If not, then a few possibilities are that Alice's certificate or the message itself might have been tampered with.

d) encrypt(m, public_bob), sign(hash(m), private_bob)
Note: Alice cannot have access to Bob's private key. If Alice has access to Bob's private key, that means it has been compromised. I will assume that Bob's private key is known to everyone.

Bob can decrypt the message.

1. Confidentiality: The message was encrypted using Bob's public key and can be decrypted only by Bob's private key but since Bob's private key is compromised, anyone who has the key now can decrypt Alice's message to Bob. Unauthorized information gathering is highly possible now and confidentiality is not maintained.

2. Non-repudiation: Alice has signed the message using Bob's private key. This serves no purpose and does not make sense. There is no way to track the message back to Alice now that the communication carries no proof that Alice is the one sending the message to Bob. Also, there is a scope that Alice can successfully dispute her authorship of the message. Hence, non-repudiation does not hold here.

3. Steps taken by Bob to decrypt the message:
   - ➢ Use his private key for decryption. Message "m" is thus obtained.
   - ➢ (Optional) Although it is useless to, Bob can use his public key to get the hash(m) from the signature. He can then hash the message "m" and compare it with the obtained hash. This is futile as Bob's private key is known to everyone and anyone can tamper with both the signature, the hashed message and the original message now such that the tampered message on hashing gives the tampered hashed message.

e) encrypt(sym, public_alice) ; encrypt(sym, public_bob) ; encrypt(m,sym)
Note: Only Alice and Bob can know the shared symmetric key.

Bob can decrypt the message.

1. Confidentiality: The message was encrypted using a symmetric key by Alice that was in turn encrypted using Bob's public key and sent over to Bob. Only Bob can retrieve the symmetric key using his private key. After obtaining the symmetric key, Bob can now decrypt the message. Since no one except Bob knows his private key, no one except him can obtain the symmetric key used to encrypt Alice's message to him. Hence, no unauthorized information gathering is possible and confidentiality is maintained.

2. Non-repudiation: The communication carries no proof that Alice is the one sending the message to Bob. Also, there is a scope that Alice can successfully dispute her authorship of the message since anyone can pose as Alice and establish a symmetric key with Bob. Encrypting the symmetric key sym using Alice's public key doesn't help in any way. Hence, non-repudiation is not maintained here.

3. Steps taken by Bob to decrypt the message:
   - ➢ Use his private key to decrypt and obtain the shared symmetric key.
   - ➢ Use the retrieved symmetric key to decrypt the cipher text and obtain the message "m".

f) encrypt(sym_1, public_alice); encrypt(sym_1, public_bob); encrypt(sym_2, public_alice); encrypt(sym_2, public_bob); encrypt(encrypt(m, sym_2), sym_1); sign(sym_1, private_alice) sign(sym_2, private_alice)

Bob can decrypt the message.

1. Confidentiality: Alice is sending the two symmetric keys used to decrypt the message as digital signatures. She has signed it with her private key and now anyone can use her public key to obtain the two symmetric keys sym_1 and sym_2 she used to encrypt the message she sent to Bob. Once they have sym_1 and sym_2, anyone can decrypt the message meant for Bob. Unauthorized information gathering is highly possible now and confidentiality is not maintained.

2. Non-repudiation: Since Alice has signed the symmetric keys (sym_1 and sym_2) individually with her private key, when Bob obtains the signature, he can use Alice's public key to obtain the two keys sym_1 and sym_2. He can also obtain sym_1 and sym_2 after decrypting using his private key (since Alice encrypted them individually using Bob's public key). Bob can now cross check the four. Since signatures use private keys, it is extremely likely that Alice is the one who sent the message to Bob since the hashed message is signed with her private key and it can be established that the public key is legitimately hers. The message was signed using Alice's private key and anyone who has access to her public key can trace it back to her. This is how non-repudiation is possible here.

3. Steps taken by Bob to decrypt the message:
    - ➢ Bob uses his private key to decrypt the cipher text obtained from **encrypt(sym_1, public_bob)** and gets the symmetric key sym_1.
    - ➢ Bob uses his private key to decrypt the cipher text obtained from **encrypt(sym_2, public_bob)** and gets the symmetric key sym_2.
    - ➢ Bob then uses the key sym_1 to decrypt the cipher text from **encrypt(encrypt(m, sym_2), sym_1)** to obtain an intermediate cipher.
    - ➢ Bob decrypts the intermediate cipher (i.e. the one from **encrypt(m, sym_2)**) using the symmetric key sym_2 and finally obtains the message "m".
    - ➢ (Optional) Bob can use Alice's public key to decrypt her signatures **sign(sym_1, private_alice)** and **sign(sym_2, private_alice)** to obtain the symmetric keys sym_1 and sym_2 and cross verify with the symmetric keys he obtained by decryption.

Question 4.

The assessment report for the heist with some ways to carry it out in no particular order of preference is given below:

- ➔ Obtain the log in credentials of the instructor or any TA. Use the stolen credentials to take over the user's account. Some of the ways in which the credentials can be obtained:
    - ◆ Install a keylogger on any TA's computer or the instructor computers. The keylogger could be a software or even a hardware device. The keylogger tracks the keyboard activity of the computer's user and once it obtains the login credentials of the TA or instructor, it should be sent to The Professor.

- If a list of stolen credentials is in possession, use credential stuffing- i.e. test the list of stolen login credentials against multiple TA and instructor accounts to see if there's a match.
- Use a spoofing attack by masquerading as a TA or instructor and log into the submission portal.
- Use a parallel session attack when a TA or the instructor is logging in to get unauthorized access to their account (if the platform does not prevent it).
- If the submission platform does not use a TLS/SSL certificate, use wireless sniffing to capture the data sent between a TA's/instructor's computer and steal their login credentials from there or start a Man-in-the-Middle (MITM) attack for the same.
- Use phishing and create a fake login page for the website that has the submissions. If any TA/the instructor falls for this, their credentials would be easily obtained.
- Use brute-force methods to guess login and password.
- Blackmail someone into giving their credentials.

➔ Now that the login credentials are in possession or login to the submissions portal can be achieved successfully, compromise and take over the user's account. The user account will have access to the submission and they can be stolen with ease. For additional fun, change the permissions such that no one TA or instructor is able to access the submissions.

➔ If the system has loose security and provides the scope, inject a malicious script into the website that collects the assignments submitted by the students. Or use this script to add oneself as a legitimate user (TA or Instructor) and then steal the submissions.

➔ Launch botnet attack to exploit a vulnerability in the TAs' or the instructor's system.
- Try to deliver malware to the TAs' and instructor's system by sending mail, online messages, etc.
- Wait for the user(s) computer to get affected with the botnet malware.
- Once enough users get infected, organise all of the machines into a network of bots that can be managed remotely. Now, look for the students' submissions in this zombie network. Alternately, look for vulnerabilities in other devices too and have them join this zombie network.

- The zombie network would also allow viewing its users' stored passwords and other credentials. Find the login credentials of the submission website/portal for the users (TAs or instructor) and use it for authentication and logging into the website as that user. Remaining submissions which were not on the zombie network's local computers could be downloaded from the website itself.

➔ If the submission website/portal uses some cloud based storage to store the submission files, launch a Man-in-the-Cloud attack and grab the cloud based files. This exploits the vulnerability in the design of many synchronization offerings.
- Grab hold of the password token which sits on the users' (TAs or instructor) device by phishing or some other way (a few others mentioned in the first point).
- Use the token to fool a new machine into believing that the attacker is the account owner.
- Steal and access file from the submissions folder on the cloud of the victim user.

➔ Avoid getting detected.

Now, to prevent such attacks, the instructor and the TAs should use precautionary measures like:
- ➢ Using strong and secure passwords.
- ➢ Not clicking on dubious links or trusting unknown websites without a certificate.
- ➢ Using a submission website that has a TLS/SSL certificate.
- ➢ Monitoring network traffic and using firewalls.
- ➢ Using reliable anti-virus applications.
- ➢ Enable two factor authentication.
- ➢ Do not give access rights to all submission files to every TA. Keep them limited and circulate the rights.
- ➢ Leave no scope for vulnerabilities by keeping the system and apps up to date.

Question 5.

a) Five such attacks are:

1. **Brute Force Attack**: In a brute force attack, the attacker systematically tries a number of combinations of alphabets, numbers, common phrases, names, words (dictionary attack) until the correct one is found. Weak passwords, recycled credentials are highly susceptible to these attacks.
2. **Impersonation Attack:** In impersonation or forgery attacks, an attacker may eavesdrop on the network traffic between an unsuspecting user and the server and intercept the data. The attacker can then try to masquerade as a legitimate user and log into the system.
3. **Parallel Session Attack**: The attacker here may once again eavesdrop on the network traffic between an unsuspecting user and the server to concurrently create a valid login message. The attacker can then successfully log into the system
4. An attacker can falsely update verification information of a legitimate user without them knowing. When the user tries to log in the next time, they will not be able to do so anymore.
5. **Stolen Verifier Attack:** An attacker can compromise the server and steal the password verifier from its database. The stolen verifier can be used by the attacker to impersonate a legitimate user, launch a DoS attack against the server etc.

b)

Some of the goals an ideal password authentication scheme should achieve are:

1. Users should be free to choose or edit their passwords.
2. The passwords should NEVER be transmitted as plaintext over a network.
3. If a password is too short, it might be easily guessable. If it is too long, hashing may take a long time and there is a possibility of DoS attack. The password length should be just perfect to allow memorization and security.
4. An unauthorized login must be detected when a user enters the wrong password. Also, if a user enters the wrong password continuously, measures should be taken to ensure that it isn't a brute force attack.
5. Passwords or verification tables must never be stored in the system.
6. The server admin should not reveal the passwords.
7. A session key must be established during password authentication to ensure confidentiality of communication.
8. The password authentication scheme must not be inefficient or unserviceable.

c)

The passwords will expire once the fraction tested number of similar passwords out of the total possible passwords exceeds 0.1.

Let the passwords expire after n seconds => the password tester would have been running for n seconds too.
Since 10000 passwords are tested every second, the number of passwords tested in n seconds would be: 10000n.

i) Since we can use 127 ASCII characters (from 1 to 127), total possible combinations of 1 to 8 characters long passwords is:

$$\sum_{i=1}^{8} 127^i = 68212339274677760$$

Hence, we can now equate
no. of passwords tested in n seconds/no. of total possible passwords = 0.1

$$\Rightarrow \frac{10000n}{68212339274677760} = 0.1$$

=> *n = 682123392746.7776 seconds*
Or 21629.99 years.
Hence, the expected time to meet the probability is 21629.99 years.

ii) Now, we can use alphanumeric characters A-Z, a-z and 0-9 in the password. Total number of allowable characters = 26+26+10 = 62
Total possible combinations of 1 to 8 characters long passwords with 62 characters is:

$$\sum_{i=1}^{8} 62^i = 221919451578090$$

Now, we can equate-
no. of passwords tested in n seconds/no. of total possible passwords = 0.1

$$\Rightarrow \frac{10000n}{221919451578090} = 0.1$$

=> *n = 2219194515.7809 seconds*
Or 70.37 years
Hence, the expected time to meet the probability is *70.37* years.

iii) We can now use only digits (0-9) in the password. Total number of allowable characters is 10 now.
Total possible combinations of 1 to 8 characters long passwords with 10 characters is:

$$\sum_{i=1}^{8} 10^i = 111111110$$

Now, we can equate-
no. of passwords tested in n seconds/no. of total possible passwords = 0.1

=> $\frac{10000n}{111111110} = 0.1$
=> *n = 1111.1111 seconds*
Or 18.5185 minutes.
Hence, the expected time to meet the probability is 18.5185 minutes.

d)

i) Since the authentication module is directly connected to the system and the system has authentication software, there is no scope of middle-man interception (or MITM attacks). The only way to spoof the system in this case would be to use a spoofing attack towards the authenticator and replace it which is not very feasible.

ii) Since the biometric hardware is on a stand-alone computer which is connected to the authentication system (over a network, supposedly), it is comparatively very easy to spoof the system now.
One way could be that the attacker can make the stand-alone computer send a fake “Yes” and get authenticated by the system. Another way could be that the attacker can intercept the communication between the stand-alone computer and the system. The attacker can trick the system into believing that they are the stand-alone computer sending legitimate “Yes”, “No” information, and perform a fake authentication by sending a “Yes”.

Question 6.

a)
Access Control Lists (ACLs) represent Access Control Matrices by storing each individual column of the Access Control Matrix with its corresponding resource (object) name. In other words, the Access Control List lists each resource(object) name (column name) and the corresponding column entries (users/processes granted access to the resource) of the Access Control Matrix together as individually separate entities (rows).

Access Control Lists are widely used in environments where users would like to manage the security of their files. Some operating systems that use ACL are Microsoft Windows NT/2000, several UNIX based systems etc. They are also used in switches and routers for filtering traffic.

Advantages of ACL-

- In the networking scenario, using ACL provides control over network traffic which can increase network performance
- Provides granular control over user and user group permissions.

- ACL is generally easy to read and comprehend.
- Since ACL is a comprehensive list of resources and rights, it works well for distributed systems.
- ACL is easy to answer this question- “Given an object, which subjects can access it, and how?”

Disadvantages of ACL-

- The list may become too long too fast and it can become frustrating to configure or just search.
- In networking, the list must be carefully maintained so that nothing unexpected can pass the router.

b)

Assumption: Bell-LaPadula Security Model

simple security property (no read up): if acc is read, then level(subj) should dominate level(obj).

*Star-property/ *-property (no write down): if acc is write, then level(obj) should dominate level(subj).*

Source: http://www.cs.unc.edu/~dewan/242/f97/notes/prot/node13.html

i) Paul can neither read, nor write to the document (SECRET{B,C}). Top Secret dominates Secret but {B,C} and {A,C} do not compare. In other words, Paul’s clearance level does not dominate the document’s classification level.

ii) Anna cannot read or write to the document classified (CONFIDENTIAL{B}) since her clearance level is (CONFIDENTIAL{C}). {B} and {C} do not compare and Anna’s clearance level does not dominate the document’s classification level.

iii) Jesse can only READ the document since Jesse’s clearance SECRET level dominates over the document’s classification level CONFIDENTIAL and simple security property applies. Jesse cannot write to the document because *-property doesn’t hold.

iv) Sammi can only READ the document since Sammi’s clearance level TOP SECRET dominates over the document’s classification level CONFIDENTIAL. Also, Sammi’s categories {A, C} >= {A}. Hence, Sammi’s clearance level dominates over the document’s classification level and the simple security property holds. Sammi however, cannot write to the document because *-property doesn’t hold.

v) Robin can only WRITE to the document since the document's classification level CONFIDENTIAL dominates over Robin's clearance level UNCLASSIFIED and the *-property holds. Robin cannot however read the document because the simple security property does not hold.

c)

i) By the first relevant entry policy, Alice will fall into Group 1 and has to abide by the Access Control List for group 1. Group 1 has permission to only read File 1, hence Alice will not be able to write to File 1.

ii) By any permission in list policy, Alice is a member of both Group 1 and Group 2. Since Group 2 has write access to File1, Alice will be able to write to File 1.

Question 7.

a)

For n hosts and distribution of keys on a per-host-pair basis, the number of different key pairs required would be ${}^{n}C_{2}$ which is $n(n-1)/2$.

b)

The field specifying when the next such CRL will be issued is given to instruct the client that downloads the CRL that the CRL will be updated on the Certificate Distribution Point after the mentioned time. Since the CRL is cached on the client's side, the next update field would let the client know that the CRL has expired and a new CRL has been generated. If the client uses the same old CRL even past its expiration, it would never be updated with the latest revoked certificates and there will be no way for the client to know if any new or old URI they visit have had their certificate revoked at the present. This can pose a security threat.

d)

- CAs operate essentially on a trust model. It is in the best interest of the CAs to provide legitimate and non-malicious services since a single misstep would be a huge blow to their credibility and reputation which would drastically affect their market, if not driving them out of business. However, we can not always trust a CA.

- In 2007, it was shown that although difficult, CA certificates that use MD5 could be faked (http://www.phreedom.org/research/rogue-ca/). Incompetency such as this may exist even in well known CAs and are just waiting to be exploited.
- Additionally, there is always a possibility that a CA may have been compromised and enough harm might have been done before the information about the breach goes public. It could also be possible that the government or some other major authority uses specific CAs to issue certificates that are capable of hijacking or intercepting certain communication and this breach of trust never gets revealed to the public.

e)

I checked my laptop's public IP address using https://www.whatismyip.com after connecting it to my phone's hotspot. I obtained both IPv4 and IPv6 IP addresses. This is called Dual Stack.

Advantages of IPv4:

- Almost all of the internet traffic still uses IPv4 while not every host supports IPv6. Hence, network operators find IPv4 more familiar. Using IPv4 will always guarantee compatibility until the entire internet shifts to IPv6.
- Even though IPv6 has larger packets and does not require NAT, it is a relatively newer concept right now. IPv4 networks are more optimised and more mature since they have been in use for so long.
- IPv4 is smaller and simpler for humans to read.

Disadvantages of IPv4:

- IPv4 has a limited address space of 2^{32} (about 4.3 Billion) which is quickly being depleted and soon there will not be enough unique IPv4 addresses left for assigning to new devices.

- Since IPv4 was published a long time ago, it has no built in security measures that can effectively prevent present day security threats and attacks without the implementation of additional security protocols like IPSec by the ISP.
- Because we are running out of unique public IPv4 addresses, measures like NAT (Network Address Translation) have to be introduced. This requires additional configuration for forwarding, firewall etc.
- IPv4 networks have to be configured either manually, or using Dynamic Host Arrangement Convention (DHCP) which can be baffling.

Advantages of IPv6:

- IPv6 has an increased address space which overcomes the address depletion issue with IPv4 and allows direct addressing.
- Enables end to end connectivity by eliminating the need of using NAT which improves the functioning of peer to peer networks and services like VoIP.
- IPv6 reduces the size of routing tables which makes routing more efficient.
- IPv6 has additional features like multicast addressing, true Quality of Service, autoconfiguration etc. IPSec is built in IPv6 that provides integrated security.

Disadvantages of IPv6:

- IPv6 addresses are difficult for humans to read and remember.
- Many network devices do not support IPv6 yet. Also, there is no communication compatibility between IPv4 and IPv6 hosts. Practices like tunneling, dual stack networks have to be used to ensure compatibility in this transition state from IPv4 to IPv6.
- It will be a tough and slow transition- going from IPv4 and IPv6. Additional costs would have to be borne by device owners to replace old devices that do not support IPv6.

Question 8.

I attended all classes except the one on 3rd September, 2021. Also, I had to leave the September 27th class at 5PM since I had to be in a meeting. I attended 11 out of 12 classes.