Vishesh Rangwani
2020155

FCS ENDSEM

Question 1:

a. Attacker 1 can perform DNS poisoning as it is placed at the switch that also connects to the local DNS server. Since all the requests for the DNS name resolution go via the local DNS server (even the cache is installed on the local DNS server), it can perform an attack to modify the DNS resolution of a website to corresponding IP to some other malicious website's IP. The attacker's website may be exactly the same as the legitimate website and may be used for phishing attacks. This can happen only if DNSsec or IPsec is not used.
Another reason for these kinds of attacks is that DNS uses UDP for data exchange. This means that there is no handshake between the client and the server for data exchange and also the source IP and port numbers are not considered by a process using UDP.
Assuming that the DNSsec is not used, the DNS poisoning attack can be performed.
There can be 2 ways to do DNS poisoning attack:
    1. Manipulating DNS server entries:
    The steps for this attack are as follows:
        - The attacker makes a query to the local DNS server for the DNS resolution of a particular website.
        - Now, if the DNS server does not have the resolution stored as cache, it will query the nameserver of the website.
        - In this meantime, when the Local DNS queries the name server, the attacker should forge a UDP response and send the IP of a malicious site (used for phishing) to the DNS server. This IP response would correspond to the request made by the local DNS server for resolution.
        - Since there is no way to verify whether the UDP packets came from a legitimate source, the local DNS server stores the mapping of the website to the IP address provided by the attacker.
        - In case the website's IP is cached, the attacker can use certain tools like nslookup to find the TTL(TIme To Live) for a mapping and then time his response well to store a malicious mapping in the local DNS server.
        - This can be done for several websites

    2. Man In The Middle
    Following are the steps:

- In this case, the attacker will have to monitor the network traffic. This can be done using tools like SolarWinds Network Performance Monitor, BetterCap, WireShark, etc.
- Next, the attacker would use one of the internet's not yet properly solved vulnerabilities of ARP spoofing. ARP maps an IP to the corresponding MAC address. The attacker can exploit this using a tool like arpspoof.
- Using this , the attacker can change the structure of the network by manipulating the MAC address of the switch in between to be replaced by his own MAC address. This way, all ARP tables of all the devices in the local network would change and instead of the switch, the data would be routed through the attacker's machine.
- This way, the attacker can intercept all the requests in the network. The attacker should enable ip forwarding in his machine so that his machine can act like an intermediate router.
- Then the attacker can use DNS spoofing modules to forward the malicious IP instead of legitimate DNS responses.

b. In a TCP SYN flood attack, the attacker sends a huge number of SYN packets to the server. The TCP SYN packets are used to establish a TCP connection with the server. It is a part of the TCP 3-way handshake. For each SYN, the server responds with a SYN ACK. The attacker does not respond to this to complete the connection and forms several of such half open connections. This blocks all of the ports of the server and it cannot form further connections with legitimate users. This can be prevented as follows:
    - Several tools such as SNORT (an IDS/IPS tool) can be deployed on the server along with FIrewall. These tools can analyze the packet headers of the incoming packets. On analyzing the headers, we can keep a count of the number of connection requests from a particular IP address. If this number exceeds a particular threshold, the IP address should be blacklisted and should not be allowed to even make connection requests for some specified amount of time (say a few days). A 'fail close' mechanism should be adopted for such requests, i.e. the packets of such IP addresses should be dropped, irrespective. This is known as Firewall filtering.
      This would be effective in blocking a particular IP from sending continuous SYN requests.
    - Another method could be that the server itself sends an incorrect SYNACK response to all the requests. In such a case, the legitimate client would respond with a TCP packet with RST flag set. This flag is set according to the TCP protocol when a segment does not meet a satisfied criteria. In case of a TCP SYN flood attack, the attacker does not wish to form connections as it wants to maintain half open connections, so it would not respond with the RST flag set. This way the server can distinguish between a legitimate and an illegitimate SYN request and then prevent itself from forming half open connections.
    - Another method could be using cookies. The SYNACK returned by the server can be constructed to have an encoded hash as the cookie identifier. This hash

may be formed from parameters such as source IP, source port, and other details that will be available even in the response for SYNACK. In this technique, the SYNACK is sent and the connection is dropped. This makes the port free for other requests. In case there is a legitimate user, it will send the response to SYNACK and using the hashing, we can use the cookie information generated before and thus create a connection for further communication. In case of SYN flood, the attacker does not respond to SYNACK as he wishes to have half-open connections. This also ensures that ports do not remain busy in half-open state as the connection requests are dropped by storing information in cookies.

- The servers can be designed in such a way that the new incoming connections are allocated 'micro-blocks' which have very low memory requirements. These micro-blocks are allocated instead of new connection objects. This prevents the over utilization of resources for SYN requests. When the 3-way handshake is completed, the object for a connection can be allocated. This would mean that the server is not overwhelmed in terms of the number of resources being used to accept new requests.
- Statistical(ML based) techniques, checking for the nature of requests can be employed. They would observe that there is suddenly an increase in the number of SYN requests and would either shut down the system for new requests or drop certain half-open connections in smaller and smaller intervals to free up ports for new incoming connections.

c. Attacker 2 can DDOS the web server as follows:
    - This step may be required for some tools. The attacker can find the authoritative IP address of the web server from the DNS using nslookup command. It should use the type parameter as 'soa' if it gets the non-authoritative response from nslookup.
    - After getting the IP, the attacker can DDoS the server using various tools such as LOIC (Low Orbit Ion Cannon), HOIC (High Orbit Ion Cannon), slowloris, etc. The attacker can use these tools to send a large number of HTTP requests to the server by mentioning its IP address. This would make the server unavailable.
      Eg: 'slowloris' makes HTTP requests from the terminal. It can be simply installed if you have python in your machine using the command 'pip install slowloris'.
      After installing the port 80 (in case of HTTP) or 443 (HTTPS) can be mentioned in the '-p' option along with the number of sockets in '-s' option.
      For instance: the command: slowloris -p 80 -s 20000 192.168.2.240 tries to make HTTP requests on port 80 from 20000 ports of the attackers machine. The target is the IP 192.168.2.240
    - The DDoS is done when there are multiple machines and sockets. So these tools can be run from multiple devices. Apart from that these tools try to utilize several of the sockets of each machine to DDoS the system and make it unavailable for legitimate users. Since multiple distributed machines are used, the IP filtering cannot be performed. If there are a large number of machines (this forms the

botnet for attack), then each machine makes requests from a large number of sockets that too repetitively, the server goes out of service.
- DDoS can also be done at layer 4 of network stack by sending a flood of such UDP packets to the server. This will also clog the network of the server and make it unavailable. UDP packets can be sent from the LOIC tool.

The attack can be mitigated as follows:

- One effective strategy to limit the number of requests to the server. This rate limiting of traffic towards the server ensures that the server at no point gets overwhelmed by a huge number of requests and doesn't go down.
- Another method could be that there could be firewall rules in such a way that if there are a large number of continuous requests, the system should shut down itself. This will ensure that the system isn't damaged during the DDoS. It should switch itself on after a specified amount of time.
- There could be a few other lightweight alternative websites for the main website in case it goes down. After shutting down, it can tell the users to visit those sites. It shouldn't redirect as it will be down itself but users themselves can go there. (Eg: This happens in case of codeforces, when it goes down.)
- One should also ensure that the attack surface (the open ports, supported protocols) should be minimized to bare minimum as necessary.
- One alternative could be to host the website on a cloud platform. The cloud service itself implements a lot of precautions against DDoS. This also ensures that this risk is transferred to the cloud service provider. Also cloud has more computing resources to handle the requests that can be configured.

References:

https://abnormalsecurity.com/glossary/dns-spoofing
https://www.cloudflare.com/en-gb/learning/dns/dns-cache-poisoning/
https://www.varonis.com/blog/dns-cache-poisoning#process
https://www.imperva.com/learn/ddos/syn-flood/
https://www.janbasktraining.com/blog/perform-ddos-attacks/
https://www.cloudflare.com/en-gb/learning/ddos/ddos-attack-tools/how-to-ddos/
Lecture Slides

Question 2:

a. The main idea behind the SSLstrip attack is that it makes the communication between the victim machine and a web server unsecure via HTTP and not HTTPS. The server thinks that it is communicating with the victim via HTTPS but actually, it is communicating with an attacker who then forwards the server packets using HTTP to the victim. It basically downgrades the security for the victim machine and the data shared by it is using plain text and susceptible to MITM attacks.

   SSLscript attack can be done by exploiting the fact that when the client wants to set up SSL communication with the server, it first sends an unsecured HTTP request to the server, who then responds via HTTP and then shifts the connection to the secured HTTPS. Basically, this is the key exchange part before a secured connection can be established. The attacker can exploit the communication channel in this small window while the communication is done on unsecure HTTP. The attacker can do this only if it is in the same network as the client.

   In the question, since we have to do SSLstripping on one of the projects of the class, I, as an attacker, am sure that the client and I would be on same network (i.e. IIITD network) as the project sites are accessible only from IIITD network. To perform the SSLstripping attack the following steps need to be performed:

   - The main idea for the attacker machine is to come in between the communication and act as a router between the client and server. So to act as a router, it must be put in port forwarding mode. This can be done using the command: echo 1 > /proc/sys/net/ipv4/ip_forward. This way, the machine will forward the IPv4 packets that it will receive.
   - Now, the next step is to redirect the packets that come on the attacker's machine. For this the attacker will have to set up iptable rules to route the packets coming from port 80 (as the initial communication by client would be via HTTP) to route out through any other port. This can be done using this command: iptables -t nat -A PREROUTING -p tcp –destination-port 80 -j REDIRECT –to-port X
     X is the port through which redirection has to be done.
     This will ensure that when the MITM will be done and the machine will act as router, it will redirect the packets incoming at port 80 to port X.
   - Now, we wish to do MITM. For this, we will have to replace the router. This can be done using ARPspoof exploit. To do this we can use the arpspoof tool using this command: arpspoof -i (Interface) -t (IP of Client) (IP of Gateway Router)**.** This will modify the ARP table (IP to MAC conversion) and restructure the network. Now our machine acts as a router. To find the IP of Gateway Router, use 'route' command and to find the IP of the client use nmap.
   - Now start the SSLstrip attack using command sslstrip -l port X. sslstrip can be installed from here if not present.
   - Now when the attacker visits the webpage, our machine will act as a router. It will intercept the packets coming in at port 80 (unsecured HTTP). This will be

forwarded to the server. Now the HTTPS will be formed between our machine (attacker machine) and the server. Whereas the communication between the client and router (again our machine → attacker) would be via HTTP.

- This can be used to exploit and view even sensitive information such as entered login details among other things. The reason being that communication over HTTP happens via plain text. On the other hand, the server will communicate with us using HTTPS. This data can be seen in sslstrip.log.

b. SSLstrip can be countered as follows:
    - Do not use Free public WiFi to do sensitive tasks such as login,banking, etc on the web. The WiFi hotspot may be created by an attacker or an attacker may be present in the network who might do an MITM attack for your communication.
    - Do not use outdated browsers like Internet Explorer (highly vulnerable to SSLstrip). Always use extensions like 'HTTPS Everywhere' (available for chrome and firefox) to ensure that all communications happen via HTTPS. On doing this, no communication over HTTP can be done by the browser. Also enable site-wide SSL.
    - While surfing the web, use VPN. The advantage with VPN is that MITM won't be possible even if you are accessing an HTTP site. The reason for this is that the data exchange would be secured due to the exchange over the VPN tunnel. This is effective because of the encryption provided by the VPN tunnel, thus preventing SSLstrip and MITM.
    - The frequently visited sites requiring credentials should be bookmarked. Whenever you open a bookmarked website, it doesn't have to do the initial communication over HTTP. It can just begin exchanging information using HTTPS. This doesn't even open the surface for SSLstrip.
    - Use HSTS (HTTP Strict Transport Security). When enabled, browsers can only allow communication using HTTPS. Even requests using HTTP will be automatically re-routed using HTTPS, thus providing no chance for an SSLstrip attack.
    - Enable secure cookies in the browser so that cookie data also gets exchanged every time using HTTPS and thus no MITM in the cookie data exchange channel is possible.

References:

https://www.computerweekly.com/tip/Sslstrip-tutorial-for-penetration-testers
https://www.hackeracademy.org/how-to-perform-a-man-in-the-middle-attack-using-ssl-strip/
https://www.encryptionconsulting.com/detailed-guide-to-preventing-ssl-stripping/
https://www.https.in/ssl-security/how-ssl-strip-work/
https://crashtest-security.com/ssl-stripping-attack/
https://www.venafi.com/blog/what-are-ssl-stripping-attacks

Question 3:

a. A thorough risk assessment needs to be done as it is critical for financial organizations to ensure the highest level of security. It might be possible that while merging the two IT systems of the 2 companies, a few security concerns may seep in. First of all, I as a CISO will have to do a thorough analysis of the various threats and vulnerabilities by doing 'System Characterization'. I will have to understand all the inputs that are possible in the system and the outputs that all the IT systems might show. I will also have to analyze the entire software and hardware of the IT system. A thorough analysis of the data storage, its criticality and sensitivity must be done along with the system interfaces and the functions the system must perform.

   Second step would be 'Threat Identification'. Under this, I will have to have a look at the attack histories of the 2 systems. Data must be properly analyzed at this step.
   One of the major threats in a financial organization is the database. Some of the important things in the database are, customer's records, their PIIs including copies of their ID proofs. The databases have a huge amount of transactions for each account. It also has data about the customer's assets and liabilities. All these details of the customer are of very high importance as many attackers may try to steal this information to get to know the financial status of certain people. Preserving this data and following guidelines to do so must be done by all organizations as per law. The attackers might even try to modify some records or indulge in cyber theft in case they get the chance to do so. So, I as a CISO need to ensure that any sort of attack on the database is not possible. Database for a financial organization is very essential and there should be zero tolerance for any threats on it. All these risks must be mitigated through best possible measures. For this aspect, risk avoidance or risk transfer cannot be an option. There must be proper detective and preventive controls for the money related data in the system.
   Threats to other things such as the network, availability of hardware at all times, physical availability of a few systems must also be considered.

   The third step would be 'Vulnerability Identification'. For this, the previous risk assessment reports, security requirements, and security test results must be considered. I would prefer to form a team whose job will be to produce the list of vulnerabilities for the entire IT system. The tasks to be performed for this would be:

   - Performing the Security Assessment: This will include analyzing and understanding the entire application structure and based on different independent and interdependent modules, security requirements must be laid out.
   - Identification of Unit Interfaces: Under this, target Vulnerability identification must be done by finding out the hosts in the network and the open interfaces or ports in different hosts and their use cases. At this step, the code must also be reviewed to look for any hidden interfaces. There must also be an overall review of the software requirements to see the utility of all the interfaces present.

- Finding users' details to understand acceptable levels of risk: Under this, utmost caution should be there as the users could even be some experienced attackers and may try to access the application in a malicious way. All these things must be identified and should be blacklisted. This includes, access from fishy networks, Repetitive requests from a particular IP. These things must not be allowed.
- In this step, the various units of software as segregated earlier must be critically examined for vulnerabilities using certain tools such as SARA, Nessus, etc. as well as human inspection. The severity of certain vulnerabilities must be determined.
- A proper testing infrastructure must be set up for different types of tests.
- The vulnerabilities that were identified must be validated using Vulnerability Validation Techniques. The following testings must be done:
    - Password Cracking
    - Unit Testing
    - Integration Testing
    - Regression Testing
    - Fault Injection Testing
    - Penetration Testing
    - Social Engineering
    - Recovery Testing
- Each validated vulnerability must be validated on the basis of ease of exploit, resources and knowledge required and impact.
- All these things such as system information, network details, results of testing and vulnerabilities must be properly documented.

In the fourth step, 'Control Analysis' is done. This determines the extent upto which the vulnerabilities must be protected and what approach should be taken for handling each vulnerability or threat. The CISO should check the current controls of the 2 organizations, and make plans based upon the vulnerabilities identified in the previous step and based on current practices being followed.

- It might be possible to deploy deterrent controls for a few systems such as maybe the system for finding some insensitive information.
- On the other hand, any interface involving transactions will have to be planned keeping in mind preventive, detective and corrective control mechanisms.
- Recovery controls for various aspects of the system must also be planned out in case the entire system or a certain component is attacked.

The fifth step would involve the 'determination of likelihood of attack' on various interfaces and different systems and subsystems of the entire software application. We will have to consider the motivation, great capacity, nature of vulnerability and the controls being planned to determine likelihood rating of various modules and assets of the entire application.

The sixth step would involve the consideration of ‘impact’ of attacks on various systems. A few systems would lead to a critical impact, such as anything related to transactions. Other things such as promotional content on the application would not have a critical impact, if that module of the application is targeted and needs to be shut down or modified. In this step, there must be criticality analysis, data sensitivity analysis and asset analysis to find out the impact rating of various systems.

The seventh step is ‘determination of risk level’. In this we will have to do a cost/benefit analysis by determining what are the modules that must be protected with all vigor. We will have to even consider not so important modules and analyze whether there is even a risk which is high enough to spend a considerable amount in protection of that system.

The eight step is ‘Controls Recommendation’. Under this the various control mechanism options for different modules such as risk transfer, risk reduction and risk transfer plan are drawn out and provided to higher authorities. They must be provided with all the analysis of costs, impact and other things being considered for controls to be deployed on each module and system with proper justification. This would also involve chalking out a plan when an attack is observed. Various options ranging from ignoring the attack to shutting down the entire system could be adopted based on the impact and threat posed by the attack.

Step nine involves ‘Documentation’. All the things we have done up to this point, risk identification, vulnerability identification and validation, various assessments, cost-efficiency evaluation and suggested controls must be properly documented. This would ensure that all the work done till now can appropriately be utilized for the next phase of progress. This whole documentation with proper explanations of various things done should be presented to the management and executives of the organization with an adequate proposal for budget and human resources required. We must properly explain the impact, process, logic, and level of risk for each vulnerability.

The final step would be ‘monitoring’ the entire system.

b. Consider the following calculations:
Risk Rating = V x L x (1 - P + U)

Where

V: The value of the information asset
L: The likelihood of the occurrence of a vulnerability
P: The percentage of the risk mitigated by current controls
U: The uncertainty of current knowledge of theVulnerability

For Information Asset A:

Given for Vulnerability #1:

V = 50
L = 0.4
P = 0 (since there is no control in risk mitigation)
U = 0.3 (since there is 70% accuracy in assumption of vulnerability, so uncertainty will be 30%)

Therefore,
Risk Rating = 50 x 0.4 x (1 - 0 +0.3)
= 26

Similarly for Information Asset B:

Given for Vulnerability #2:
V = 40
L = 0.3
P = 0.50
U = 0.2 (Since assumptions are 80% accurate, uncertainty = 20%)

Therefore,
Risk Rating = 40 x 0.3 x (1 - 0.5 + 0.2)
= 8.4

Given for Vulnerability #3:
V = 40
L = 0.5
P = 0.2
U = 0.2 (Since assumptions are 80% accurate, uncertainty = 20%)
Risk Rating = 40 x 0.5 x (1 - 0.2 +0.2)
= 20
Risk Rating for Asset B = 20+8.4 = 28.4

Since Asset B has a higher risk rating, it is more important for the protection of organizations' information.

References:

Lecture Slides

Question 4:

a. There are several steps that must be taken to prevent a CSRF attack. The following measures must be taken to prevent it:
    - As a developer of the social media platform, one of the most important things to prevent CSRF is to ensure that no one is able to embed images or any other web objects that may contain a link or script to be redirected to another website. There must be a check on the things being uploaded such as images, text, etc. that they don't not contain any links or scripts
    - The above measure can be implemented by using the technique of input validation in all the fields that may accept any form of data from a user. In input validation, any links, HTML tags, Javascript code or any scripting code is not allowed even in embedded form in documents or images. Input validation must be both at the frontend as well as the backend so that the user does not try to embed a link using just a POST request from tools such as BurpSuite or Postman.
    - An addition to the above point is that there must not be any XSS vulnerabilities in the website. In case there are, the attacker would be able to add links and redirections using Javascript. Prevention of XSS is explained in detail in part b of this question.
    - For each request, use a session based or a request based random token. A new token for each request may hamper the usability, so session-based token would be better suited. The token is generated by the server and the browser should not be able to remember it. If this is the case, any HTTP request that may emerge from the site would have the added random token. If this is the case, the token value in the website being redirected to will not match and hence CSRF won't be possible. It is important that the tokens are being implemented on the website being redirected to. If not, at least there should be a check on any additional information that may come through.
      It is good practice to have CSRF tokens on each site so that there aren't any redirections from any other site in the form of CSRF attack as the server won't be able to verify the token value.
    - The developer must also set that only Site-based cookies are shared. This ensures that the browsers do not use the cookies of any other site while using the website. This helps in preventing CSRF as cookies are needed to remember the state of Login in an HTTP connection. If the website doesn't allow cookies of their sites to be sent, the attacker won't be able to use them to do any malicious stuff on the site where the user is logged in.

    For the site preventing CSRF attack on itself, apart from the above these must also be considered:
    - Do not use GET requests for any state change requests. State change must always be done via the POST request. This is a basic prevention of CSRF attack as in case of a GET request, the details such as the details of a monetary transaction (details about the amount, payee and payer) are sent in plain text in

the URL. This is unsafe as an attacker might maliciously utilize the authentication done on the transaction site by embedding a URL in any other site. The URL will contain the parameters that are desirable to the attacker and harmful for a naive user.

- Use of session based tokens as explained above.
- All transactions must involve at least one or more manual user interaction such as solving a captcha, or entering an OTP, or even asking again for password. This will ensure that all transactions must involve users to actively participate in them and the transactions are not automated via links or scripts. It is also good to inform users about transactions using email or message.

b. XSS attacks enable an attacker to write any desired code (that may be persistent, i.e. remains available for all users that use the website or non-persistent, i.e. for the current user only to extract information using JavaScript) or manipulate the DOM of the webpage. These attacks can be prevented as:
    - For XSS attacks, the attacker has to enter a malicious input in any of the fields. To prevent this from happening, we must perform input sanitization and input validation both at the frontend as well as backend. Under this, it must be ensured that the user does not put in characters such as <,>,(),;, etc. All these may be used to execute HTML or Javascript code. One of the better ways for input sanitization is to whitelist the kind of characters that a user can input in a field. For example, the user should only be allowed to enter alphanumeric characters and underscore as username and no other special characters.
    - The content can be sent through a canonicalize() function before storing in the database to ensure it is sanitized.
    - Another way to prevent XSS attacks is output encoding. Any user data that must be rendered on the web page must be encoded to prevent it being treated as an active content. This prevents it being rendered as HTML, JavaScript or XML content. As a part of this, it must be ensured that the data being output should be printed as plain string text so that it doesn't get executed.
      In the given scenario, output encoding of all the data on the website must be done so that any malicious code that has been entered in the website is not rendered and executes undesirably.
    - We should also not allow execution of scripts and prevent DOM or any data from being modified by scripts. This can be done by implementing Content-Security Policy offered by various browsers. This allows control over where JavaScript, HTML, XML, etc code can be loaded from and where it is executed. Content security policy can be set up in response header, indicating the browser to never execute inline Javascript code. We can also specify the URIs from where JS can be loaded so that no other Javascript code is executed.
      This is also useful in our case as we can specify that maliciously entered JavaScript or any other code can be prevented from execution.
    - Various language specific encodings must also be taken into account based on the backend framework.

To remove any malicious code that has been input in our website,

- Do output encoding for everything that is printed on screen so that code does not get executed as explained before.
- Specify content security police as explained before.
- Scan the entire database and look for content that contains unusual data such as undesired characters like ;,<,>,!, etc. and remove all that content from the database.
- Similarly, also check through all the backend code to and remove any additional code that may have been undesirably added.

c. Prevention for SQL Injection:
   - Input Validation: It has been common in preventing the above 2 attacks as well. Do check for fields that require text input for any undesirable characters such as ‘, =, |, etc. Apart from these also validate the input from list fields such as radio button or dropdown as these may be modified to contain unallowable content using intercept tools such as BurpSuit. So check them before storing them in the database.
     There should also be a limit on the length of input a field can take as input.
   - In the case of SQL, it is essential to use parameterized queries and well prepared statements on the backend to search through the database. Direct input of the user should not be put in the SQL query, rather it should be stored in the variable, then content in the variable should be encoded (such as null encoding in the end of the string). One necessary check would be to not allow characters such as ;,--,!,{,},’ etc. in the variable. This well formed statement using the variable should be used to query the SQL database. This also involves avoiding use of dynamic SQL queries (which may be faster and easier to write but unsafe).
     Apart from proper statements, it is also beneficial to use Stored Procedures in the database that require variable binding to process data. This also prevents SQL injection attacks.
   - One of the important cues on running the SQL injection is from the error thrown on on entering an invalid input. The error message may print the name of the tables, the columns and the structure of the entire query. This allows the attacker to creatively form the input so that the SQL query is executed in his desired way, revealing or modifying critical content stored in the database. Throwing of such errors should be prevented.
   - We must set the least privileges in SQL databases in such a way that the user should have the least access when he is performing login or other non-modification activity. Whenever a query must be run, the user should have the least possible privilege to access or modify the database. Different levels of privileges such as read only, edit, delete should be specified to prevent blind SQL attacks.
   - Another option, which totally prevents SQL injection is to shift to a safer database option if possible such as MongoDB. It is not an SQL based database. Rather it

is a schema-less database and stores in the form of documents of varying sizes. Due to this injection attack is not possible

References:

https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html#user-interaction-based-csrf-defense
https://www.acunetix.com/websitesecurity/csrf-attacks/
https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html
https://www.hacksplaining.com/prevention/xss-stored
https://portswigger.net/web-security/cross-site-scripting
https://security.berkeley.edu/education-awareness/how-protect-against-sql-injection-attacks
https://www.indusface.com/blog/how-to-stop-sql-injection/
https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
https://www.esecurityplanet.com/threats/how-to-prevent-sql-injection-attacks/
https://www.tutorialspoint.com/mongodb/mongodb_advantages.htm
Lecture Slides

Question 5:
Yes, it is possible for a highly skilled attacker to exploit the buffer overflow vulnerability in the OpenSSL library.

a. The following steps can be performed to exploit the vulnerability:

- We know that to exploit the buffer overflow vulnerability, the attacker will either have to exhaust the memory in stack or make the stack pointer point to a desired location so that where the function returns can be controlled.
- There could be a possibility wherein the long strings of data in certain fields are entered so that becomes a DoS attack as the memory space would be exhausted and it would not be possible to communicate via SSL as it will not work properly.
- This all can be performed to exploit the OpenSSL overflow vulnerability during the initial process of communication involving verification of certificate. During the process of verification or signing of the certificate, the attacker will have to provide an email address whose length will overflow 4 bytes.
- The above would work because in the email address field, only 4 bytes of data is allowed and controlled and checked internally, but that field can accept a longer data. When this overflows, the pointer on stack does not do the desired actions and can lead to a DoS as the pointer goes out of bound for the local variables on the stack.
- Since OpenSSL is an open source library, the attacker can imagine what the stack of the C/C++ program will look like before the variable holding the email is worked upon in the stack. Based on that, the attacker can input the field such that on decoding into assembly language and further to machine code, the Program Counter should point to some other undesired function that would be executed on reading that.
- The attacker will also have to have knowledge of assembly language and its translation to machine code to ensure that the execution of arbitrary code is done properly. He will also need to add string format for conversion into several ‘nop’ statements in assembly language to gain more control over the attack.
- This can be exploited to do an Off-by-one buffer overflow attack or even Overflow function pointers & longjmp buffers.

- Another vulnerability that was based on Buffer Overflow in OpenSSL was the Heartbleed vulnerability.
- SSL provides a ‘Heartbeat’ option to confirm whether there still is someone active on the other end of the connection. This feature is provided so that the one party can ensure whether the other is still active as it may be the case that some intermediate routers may drop the SSL connection if the connection remains idle for some time.
- In this heartbeat message, there are 3 components, a request for acknowledgement, a field for a message and number of characters in the message. For response, the server responded with the same message.

- The exploit could be that if someone sends a message whose length is less than say, ‘x’ but the person has entered in a number of characters a number ‘x’, then the server did not check if the message really had ‘x’ number of characters and sent ‘x’ characters reply. Since the attacker’s message had less than ‘x’ characters, the server sends some other information that should not have been revealed to the other party.
- The maximum number of characters could be upto 64000 and they could be requested multiple times. Using this the attacker could gain a lot of undesirable information.

b. Defense against such attacks:

- The ‘Heartbleed vulnerability’ was present in OpenSSL version 1.0.1 to 1.0.1f. It is fixed in subsequent versions by checking on the number of characters that are actually sent in the heartbeat message. So to ensure that you are not targeted with an exploit of this vulnerability, ensure the certificate of the server is signed using the latest OpenSSL versions.
- For the other exploit as well the vulnerability has been overcome in latest versions of OpenSSL starting from version 3.0.7. Since the vulnerability is overcome in 2022 only, it is recommended to update OpenSSL as soon as possible.

Other methods to overcome buffer overflow could be:

- Avoid using C/C++ languages that do not do a strict check of the number of characters being entered and the memory allocated to a variable. Use typesafe languages such as Java, etc.
- In case you have to use C/C++, avoid using functions such as strcpy(), gets(), strcat(), scanf(), etc.
- For each input a strict check on the amount of memory allocated and the data being attempted to store in that memory location.
- It is important to find these overflow vulnerabilities by doing manual code inspection.
- In a system, the address space should be randomized so that the attacker cannot predict the memory location to direct the stack pointer to.
- If possible, one should also make the stack non-executable so that the fields stored in the stack as variables are not interpreted as executable pieces of code.

References:

https://nsfocusglobal.com/openssl-multiple-buffer-overflow-vulnerability-notice/#:~:text=OpenSSL%20Buffer%20Overflow%20Vulnerability%20(CVE%2D2022%2D3602)%3A&text=In%20a%20TLS%20client%2C%20connecting,and%20a%20malicious%20client%20connects.
Lecture Slides
https://www.openssl.org/blog/blog/2022/11/01/email-address-overflows/
https://en.wikipedia.org/wiki/Heartbleed#Affected_OpenSSL_installations
https://www.redhat.com/en/blog/openssl-email-address-buffer-overflow-security-flaws
https://www.vox.com/2014/6/19/18076318/heartbleed

Bonus Question:

- Zombinder is a third party service using which an attacker can add a module of code in the apk of an android application.
- An attacker can create this code module in such a way that it does not affect the proper working of the application but it may send some data of that user about the app to the attacker.
- The code can even be made in such a way that apart from normal functioning of the app, it does something extra, something malicious i.e. using the services enabled for the app such as location, camera access to harm the privacy of the user.
- After adding this to the main android app, the zombinder may bind this and form an APK. If the user is forced to somehow download the APK, or update the current app using the link provided by the attacker, the zombinder may bring in malicious code into the system, causing the damage to the user.
- The data manipulated by the malicious code can be made to be shared with the attacker when the Android device is connected to the Internet using WiFi.

References:

https://www.threatfabric.com/blogs/zombinder-ermac-and-desktop-stealers.html
https://securityaffairs.co/wordpress/139431/malware/zombinder-apk-binding-service.html