


National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Information Security	Course Code:	CS3002
	Program:	BS (CS, SE, DS)	Semester:	Fall 2023
	Duration	60 minutes	Total Marks:	35
	Date:	14-11-23	Weight	12.5
	Exam Type:	Midterm-II	Pages	6

Student : Name: _____ Roll No. _____ Section: _____

Instruction: If you think some information is missing then make an assumption and write it clearly.

Question 1: [CLO:1] [5 marks]

- _____ malware type does not modify the total size of infected file.
A. prepending
B. appending
C. overwriting
D. cavity
- In Kerberos protocol, the Ticket-Granting Server (TGS) issues _____ to clients.
A. ticket-granting tickets
B. verification tickets
C. service tickets
D. correct-user tickets
- Which one of the following characters is most important to restrict when performing input validation to protect against XSS attacks?
A. &
B. !
C. <
D. \$
- What is the best design for input validation?
A. Detecting attacks and rejecting them
B. Setting a policy for good input and rejecting everything else
C. Setting a policy for bad input and logging them
D. None of the above
- Random Sample Query is a _____ perturbation technique to protect from _____ attack.
A. Data, In-band
B. Output, In-band
C. Data, Inferential
D. Output, Inferential

Question 2: Short Questions

[CLO: 3] [4+2+2+2 marks]

- a. Write SQL statements (DB access control) for each of the following tasks. (4 marks)
- i. Allow a user 'peter' to create new rows or delete some rows in 'Repository' table. He should not be able to see table data or modify anything within the table rows.
 - ii. User 'sam' has full access for all tables in databases. Take away his writing privileges, but keep the reading access.

i.

GRANT INSERT, DELETE ON Repository TO peter;

ii.

REVOKE INSERT, UPDATE, DELETE ON ANY TBLE FROM sam;

- b. “In salted password storage, the salt should not be leaked to attackers in any case, otherwise user’s password will be compromised.” Discuss whether you agree or disagree with the statement.

Incorrect statement. Salt itself is not a secret information. Server stores hashcode = hash(password| salt).

If salt is leaked, attacker still needs to "crack" the hashcode to get the password.

Main benefit of salt is to slow down the dictionary attacks.

- c. Discuss how checking the Origin header can help in mitigating CSRF attacks.

The Origin header is added automatically by browser to indicate the source of requests.

In CSRF attack, forged requests are initiated from attacker’s website, hence the Origin header will disclose this information to server, which can then reject such requests.

- d. What kind of obfuscation techniques is used by retro viruses to avoid detection and analysis?

Retro virus tries to bypass or hinder the operation of an antivirus, personal firewall, or other security programs.

They generally have a database of identification mechanisms for different security controls like process names, registry keys. Once identified, the security controls can be terminated or corrupted.

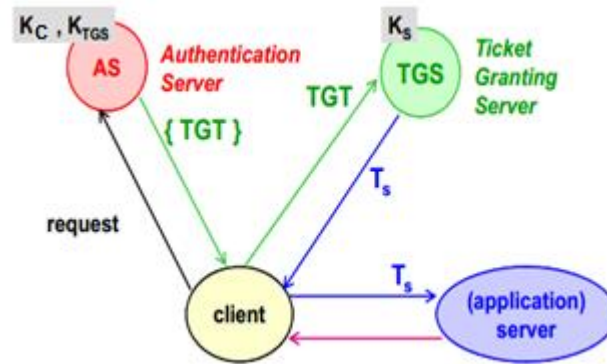
These could also block users from updating their antivirus software or opening of system utilities or antivirus vendor websites.

Question 3:

[CLO: 1]

[6 marks]

The following figure depicts the Kerberos Authentication process with keys involved.

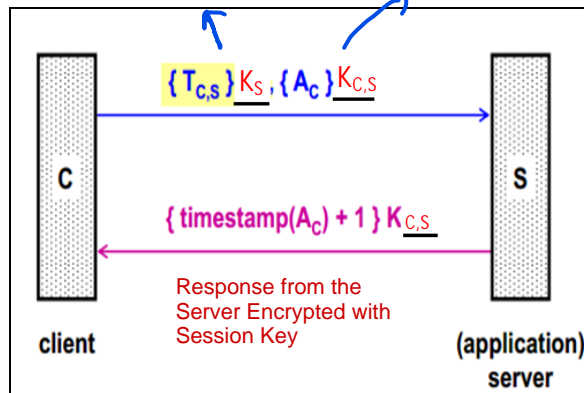


If a client wants to access a specific service from application server then how the user will be authenticated? Partial information is provided in the following figures. Complete the missing information and explain each process in the right column.

<p>Diagram showing the first step of Kerberos authentication:</p> <ul style="list-style-type: none"> Client (C) sends "C, TGS" to the Authentication Server (AS). AS responds with a message: $\{ K_{C,TGS}, \{ T_{C,TGS} \} K_C \} K_C$. The response is broken down into: <ul style="list-style-type: none"> session key: $K_{C,TGS}$ Ticket granting ticket: $\{ T_{C,TGS} \} K_C$ secret key of TGS: K_C 	<p>Client sends an authentication request to AS, along with its identity C.</p> <p>AS replies with a ticket granting ticket, that is encrypted with secret key of TGS. Additionally, it also sends a session key for encrypting client-TGS communication. Whole message is encrypted by client's secret key (derived from their password).</p>
<p>Diagram showing the second step of Kerberos authentication:</p> <ul style="list-style-type: none"> Client (C) sends "s, $\{ T_{C,TGS} \} K_{C,TGS}$, $\{ A_C \} K_{C,TGS}$" to the Ticket Granting Server (TGS). TGS responds with a message: $\{ \{ T_{C,s} \} K_s, K_{C,s} \} K_{C,TGS}$. The response is broken down into: <ul style="list-style-type: none"> service ticket: $\{ T_{C,s} \} K_s$ Session Key for client-service communication: $K_{C,s}$ Session Key for client-tgs communication: $K_{C,TGS}$ 	<p>To request authorization of required service, client sends s (service id) and encrypted TGT to TGS. Client also sends an authenticator message, encrypted by their session key.</p> <p>TGS successfully decrypts both messages, which proves that client is authenticated. It then checks if the client is authorized to use the service. If yes, it sends a service ticket, encrypted by secret key of application-server. It also sends a session key for client and application-server communication. Whole message is protected by client-TGS session key.</p>

Service Ticket Encrypted with Application Server's Secret Key

Authenticator Encrypted with Client-Service Session Key



Client requests access to the service by sending the encrypted ticket to server. Client also sends an authenticator message containing a timestamp. This message is encrypted by their session key.

Application server decrypts the ticket to ensure client has been authorized. It then extracts the timestamp from client's message and returns an incremented timestamp, encrypted by this session's key. This provides confirmation to client that server is ready to serve the user.

Question 4:

[CLO: 4]

[6 marks]

In the given login screen, what input would be required to exploit the SQL injection vulnerability. Moreover, also write down the complete SQL query with your input which results in successful login without valid username and password.

Please sign-in

Name

Password

Login

Dont have an account? [Please register here](#)

Input:

' or 1=1; --

SQL Query:

SELECT id FROM users WHERE name = '' OR 1=1; -- ' AND password = '';

Question 5:**[CLO: 4]****[3+2+3 marks]**

Answer the given questions according to the following code. Note that `scanf()` is used in C to get string/integer input from user. It takes two parameters: `input_format` and `dest_address`. Data in memory is stored in ASCII encoding.

```
int getMarks (int rollNumber);
Boolean checkName (int rollNumber, char* firstname);

int main (int argc, char **argv)
{
    Boolean valid = False;
    int rollNumber = 0;
    int marks = 0;
    char firstname[15] = "";

    printf("Enter your roll number: ");
    scanf("%d", &rollNumber);

    // function to get marks from the database
    marks = getMarks(rollNumber);

    printf("Enter your first name: ");
    scanf("%s", firstname);

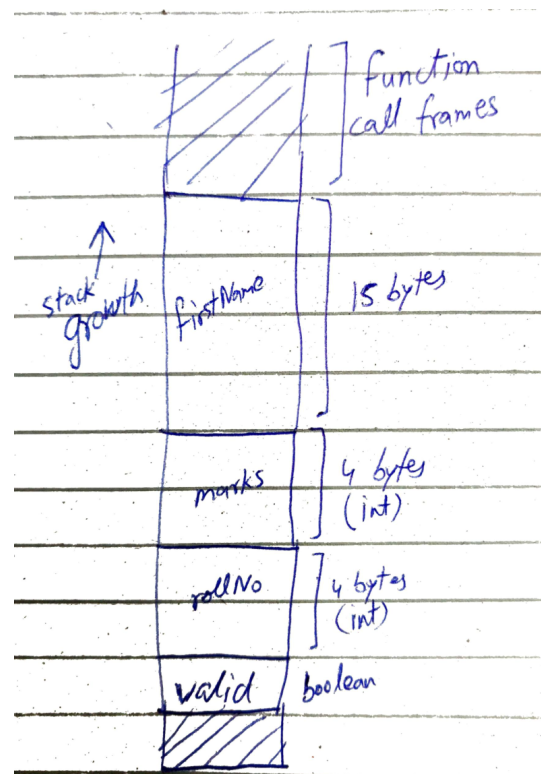
    // function 'checkName' validates the name of the user against
    // the rollNumber and returns a Boolean value.
    valid = checkName(rollNumber, firstname);

    if (valid == False)
        return 0;

    printf("%s ", firstname);
    printf("your marks are %d\n", marks);

    return 0;
}
```

1. Depict how stack will grow?



2. Identify the problem in the above code?

For first name, user input is directly saved in memory without bounds-checking. It will result in buffer overflow. Adjacent memory cells will be overwritten (other local variables, then the main parameters, then return address).

For roll number input, user can provide a really large number, but only lower 4 bytes will be stored, causing an integer overflow.

3. How can you exploit firstname input to display 98 as your marks?

Provide as input a string of exactly 16 characters.

First 15 characters can be anything, but the last character should be letter "b" (ascii code 98).

Last character will overflow into marks variable, which will be treated as an integer with value 98.