NANYANG TECHNOLOGICAL UNIVERSITY

SEMESTER 2 EXAMINATION 2017-2018

CE4067/CZ4067 – SOFTWARE SECURITY

Apr/May 2018 Time Allowed: 2 hours

INSTRUCTIONS

1. This paper contains 5 questions and comprises 7 pages.

2. Answer **ALL** questions.

3. This is a closed-book examination.

4. All questions carry equal marks.

---

1. "Be liberal in what you accept" has been a design principle for the internet and the web. "Don't trust your inputs" is a design principle in software security. These two principles hardly become compatible.

   (a) What are meta-characters? Give examples for meta-characters in SQL. What is the usage of the meta-characters you have given? Describe an attack that makes use of meta-characters in user input.

   (10 marks)

   (b) What is an escape character? What is their purpose? Which escape character is used in SQL?

   (5 marks)

   (c) A liberal implementation of UTF-8 encoding of Unicode characters leads to a situation where characters have more than one representations. Why is this a potential security problem? How would you deal with this problem, other than insisting on a strict implementation of UTF-8 encoding?

   (5 marks)

2. Return-to-libc, return-oriented programming, and jump-oriented programming all make use of existing executables.

(a) Give a brief description of Data Execution Prevention (DEP) and Address Space Layout Randomization (ASLR). Can these measures prevent return-to-libc attacks? Justify your answer.

(10 marks)

(b) Give a brief description of return-oriented programming (ROP).

(4 marks)

(c) Give a brief description of jump-oriented programming (JOP). Why is it more difficult to defend against JOP than against ROP?

(6 marks)

3. "Type safety is cheating. It does not solve the problem but throws it back to the programmer."

(a) What does it mean for a programming language to be type-safe? What is the explanation for the remark above?

(5 marks)

(b) What is a race condition? Does type-safety always detect race conditions? Can type-safety guarantee that all race conditions are flagged? Justify your answer.

(10 marks)

(c) Type confusion attacks might exploit bugs in the type system or directly modify objects that are managed by the type system. Explain how such a direct modification might be performed, despite protection by the type system.

(5 marks)

4. (a) Figure Q4a shows a data flow diagram of a simple web-based grade entry/view system. The *Enter Grade* component accepts "Grade entry" requests from *Lecturers* via HTTP. It will then check whether the lecturer is a valid user for entering the grades and update the *Grades* data store if the request is valid. The *View Grade* component accepts "Grade view" requests from *Students*. The students should ONLY be able to view their grade.

One of the serious security threats for this system is *Tampering with Data* threat, which may allow malicious users to tamper with the grades.

Draw a threat tree in an outline view, not more than 4 levels, to analyze this threat.

(6 marks)

(b) Figure Q4b shows the web page for sending the "Grade view" requests. The web page contains four input parameters – user id, password, course code, and output format. Upon clicking the "View Grade" button, an HTTP GET request is generated and sent to the *view_grade* server program. The following shows a sample URL of a GET request generated:

```
http://www.abc.com/view_grade?uid=wxyz12&pwd=123456&
                cid=CS4063&format=html
```

Assume that you need to generate security test cases to test the security of the *view_grade* server program. In the test cases, the input parameters must be mutated by applying the following mutation operators:

Operator 1. Valid+Invalid: well-formed data with malformed data attached

Operator 2. Wrong type: incorrect data type

Operator 3. XPath meta-characters: characters that have special semantics to XPath parser

Generate nine security test cases. Use pairwise test generation method to observe the behaviors of different combinations of mutated inputs.

(14 marks)

Note: Question No. 4 continues on Page 4

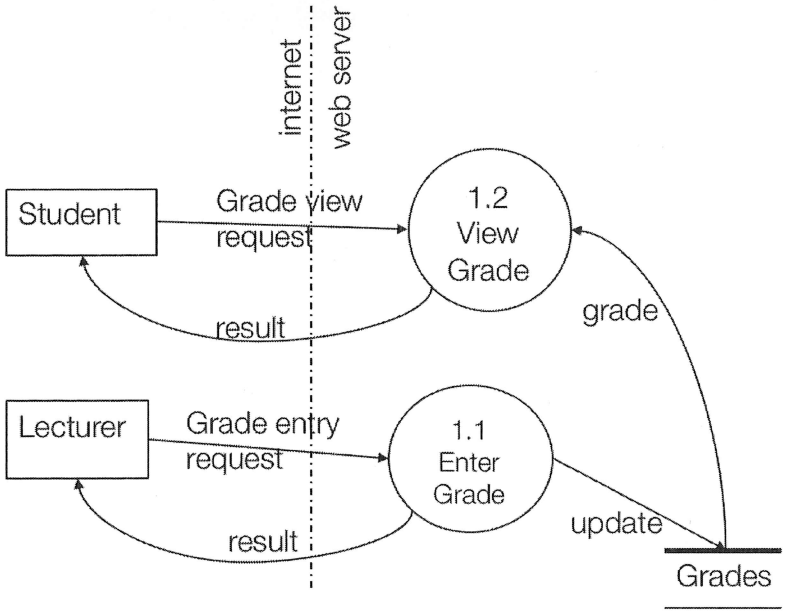

Figure Q4a The data flow diagram of a web-based grade entry/view system

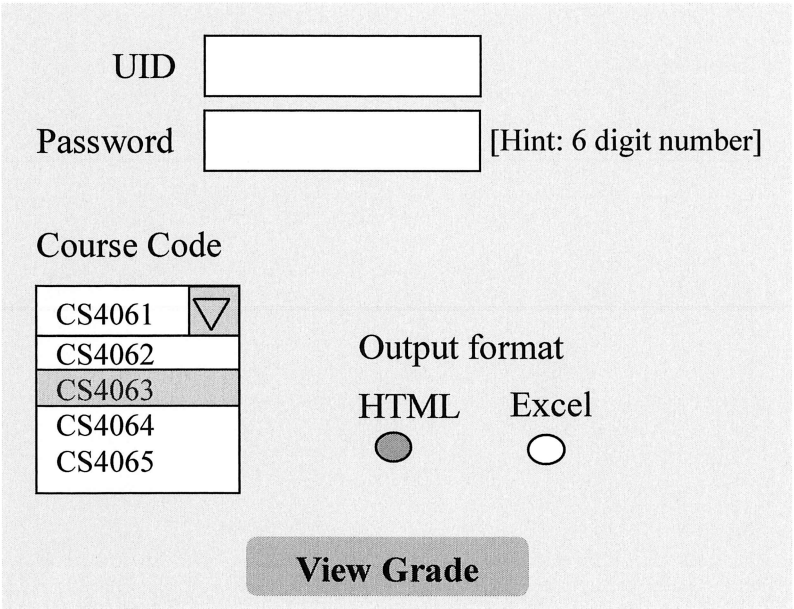

Figure Q4b The web page for sending the grade view requests

5. Figure Q5a shows the code snippet of the *enter_grade* server program. Figure Q5b shows a sample XML database that stores the user id, password, and grade information of each student regarding each course. The *enter_grade* program accepts HTTP GET requests for entering the grades. It checks whether the password provided is the correct lecturer's password and updates the grade information in the XML database. It then returns an HTTP response message stating whether the grade entry is successful or not.

(a) State the types of vulnerabilities contained in the code shown in Figure Q5a. Locate the line numbers of the vulnerable code.

(9 marks)

(b) Generate a security test case that demonstrates how a student can tamper with her/his grade stored in the XML database in Figure Q5b. (Assume that the student does not know the lecturer's password)

(4 marks)

(c) Suggest at least 3 defense techniques that can prevent the vulnerabilities in the code shown in Figure Q5a. Demonstrate how a vulnerability can be fixed using your suggested technique. (Use a code snippet if necessary)

(7 marks)

Note: Question No. 5 continues on Page 6

```
Protected void doGet(…) {
  // HTTP response output
1 PrintWriter out=response.getWriter();

2 try{
3   out.println("<!DOCTYPE html><html><body>");

    // access XML document
4   File inputFile = newFile("grades.xml");
5   Document doc = DocumentBuilderFactory.newInstance().
                              newDocumentBuilder().parse(inputFile);
6   XPath xPath = XPathFactory.newInstance().newXPath();

   // get inputs from HTTP GET Parameters
7  String cid = request.getParameter("cid");
8  String pwd = request.getParameter("pwd");
9  String suid = request.getParameter("uid");
10 String grade = request.getParameter("grade");

    // construct & execute a query to grades.xml
11  String query = "/grades/course[@cid='" + cid +
                          "' and @lecturer_pwd=" + pwd + "]/student";

12  NodeList nodeList = (NodeList) xPath.compile(query).
                               evaluate(doc, XPathConstants.NODESET);

    // access each student returned by the query
13  boolean gradeEntered = false;
14  for(int i=0; i<nodeList.getLength(); i++) {
15     Element student = (Element) nodeList.item(i);
16     String uid = student.getElementsByTagName("uid")
                                          .item(0).getTextContent();

       // enter the grade if uid matches with the input
17     if(uid.equals(suid)) {
18         student.getElementsByTagName("grade")
                                        .item(0).setTextContent(grade);
19         gradeEntered = true;
20         out.println("Grade Entered Successfully!");
        }
     }

21   if(!gradeEntered)
22      out.println("Student UID: " + suid + "Not Found!");
     else
23      … // code for dumping the update to grades.xml document; not shown

24   out.println("</body></html>");

25  } catch(Exception e) {
26     e.printStackTrace();
27  } finally {
28    out.close();
    }
  }
```

Figure Q5a enter_grade Java Servlet

Note: Question No. 5 continues on Page 7

```
<?xmlversion="1.0" encoding="UTF-8"?>

<grades>

<course cid='CS4061'
              lecturer_pwd='812813'>
<student>
<uid>John11</uid>
<pwd>300300</pwd>
<grade>B</grade>
</student>

<student>
<uid>Jack12</uid>
<pwd>004004</pwd>
<grade>F</grade>
</student>
…
</course>

<course cid='CS4062'
              lecturer_pwd='812813'>
<student>
<uid>John11</uid>
<pwd>300300</pwd>
<grade>B</grade>
</student>

<student>
<uid>Jack12</uid>
<pwd>004004</pwd>
<grade>C</grade>
</student>
…
</course>
…
</grades>
```

Figure Q5b grades.xml document which stores the grade information of the students

END OF PAPER

CE4067 SOFTWARE SECURITY
CZ4067 SOFTWARE SECURITY

Please read the following instructions carefully:

1. **Please do not turn over the question paper until you are told to do so. Disciplinary action may be taken against you if you do so.**

2. You are not allowed to leave the examination hall unless accompanied by an invigilator. You may raise your hand if you need to communicate with the invigilator.

3. Please write your Matriculation Number on the front of the answer book.

4. Please indicate clearly in the answer book (at the appropriate place) if you are continuing the answer to a question elsewhere in the book.