Com6501 - Exam

Computer Security and Forensics (University of Sheffield)

The University Of Sheffield.

Data Provided:
None

DEPARTMENT OF COMPUTER SCIENCE Spring Semester 2018-2019

COMPUTER SECURITY AND FORENSICS 2 hours

Answer ALL FOUR QUESTIONS. Write clearly in the sense of logic, language, and readability. The clarity of your arguments and explanations affects your grade.

Questions 1 and 2 carry a weight of 20%, questions 3 and 4 carry a weight of 30%. Figures in square brackets indicate the percentage of available marks allocated to each part of a question.

THIS PAGE IS BLANK.

1. **Security Fundamentals & Access Control**

a) Many banks issue hardware tokens for authenticating their e-banking customers. These hardware tokens

1. require the user to enter a code into the hardware token and
2. generate a one-time password that the user enters as password into the e-banking application (website).

Briefly explain why this is considered a two factor authentication and explain if this is a good or bad two factor authentication system. [15%]

b) Briefly explain the concepts *access control* and *usage control* and name one example for each. [20%]

c) Convert the following access control matrix into an equivalent hierarchical role-based access control model (RBAC):

	File1	File2	File3
Alice	read	read	read,write
Bob	read	read, execute	read,write
Charlie	read, write	read, write, execute	read
Dave	read	read	read
Eve	read	execute	
Fiona	read	execute	
Gail	read	read	read

(i) Define formally the users of the equivalent hierarchical RBAC model. [5%]

(ii) Introduce roles and define formally the mapping from users to roles. [15%]

(iii) Define formally the role hierarchy. [15%]

(iv) Introduce permissions and define formally the mapping of roles to permissions. [15%]

(v) Compute the permissions of Bob using your hierarchical RBAC model and check that they are equivalent to the permission of Bob defined in the access control matrix. [15%]

2. PKIs & Cryptography

a) Explain briefly one problem of certificate revocation lists (as part of a Public Key Infrastructure).

[10%]

b) Explain briefly the concept of *integrity* of a message and name a cryptographic primitive that can be used for providing integrity. [15%]

c) Recall the *one-time pad* (also called Vernam cipher) as defined in the lecture. Explain briefly, if using the same key for encrypting several message is secure or not.

- If it is secure, give an informal justification why it is secure.
- If an attack is possible, explain this attack.

[15%]

d) Use the Extended Euclidean Algorithm to compute t, such that

$$23 \cdot t \mod 36 = 1$$

Describe briefly the intermediate steps of your calculation. [30%]

e) Alice wants to compute an RSA key-pair. She chooses the large prime numbers $p = 13$ and $q = 7$. Complete the RSA key generation for Alice, i.e., compute an RSA key pair for her. Describe briefly the intermediate steps of your calculation.
[**Hint:** You might want to make use of the following fact: $7^{-1} \mod 72 = 31$.] [20%]

f) Assume you need to implement RSA in a programming language (e.g., Java). Explain, by giving one example, why a direct translation (i.e., an implementation that naïvely follows the textbook description) of RSA might be insecure. [10%]

3. **Security Protocols**

a) Explain briefly the concept of a *replay attack* and name one technique that can be used for preventing replay attacks. [10%]

b) (i) Explain briefly the *Dolev-Yao* attacker model. [10%]

(ii) Name one attack that *can be found* using the Dolev-Yao attacker model. [5%]

(iii) Name one attack that *cannot be found* using the Dolev-Yao attacker model. [5%]

c) Consider the following protocol with the two roles C and KAS and the two public functions pk(_) and sigk(_):

1. $C \longrightarrow KAS: \ \{C, \mathsf{pk}(C), \mathsf{sigk}(C)\}_{\mathsf{inv}(\mathsf{sigk}(s))}, \{T_C, N_2\}_{\mathsf{inv}(\mathsf{sigk}(C))}$
2. $KAS \longrightarrow C: \ \Big\{\{KAS, \mathsf{pk}(KAS), \mathsf{sigk}(KAS)\}_{\mathsf{inv}(\mathsf{sigk}(s))}, \{K, N_2\}_{\mathsf{inv}(\mathsf{sigk}(KAS))}\Big\}_{\mathsf{pk}(C)}$

State whether or not the protocol transmits the key K securely (i.e., secretly and authentically) from KAS to C. If it is secure, explain why. If it is not secure, explain the possible attack and briefly explain a possible fix. [15%]

d) In the Dolev-Yao attacker model, consider the following intruder knowledge:

$$M = \Big\{ \{\!|\, \mathsf{inv}(\mathsf{pk}(b))|\!\}_{g(n_2)}, \ \ \{\{n_1\}_{\mathsf{pk}(i)}, \{n_2\}_{\mathsf{inv}(\mathsf{pk}(a))}\}_{\mathsf{inv}(\mathsf{pk}(b))}, \\ \mathsf{pk}(b), \ \ \mathsf{pk}(a), \ \ \mathsf{pk}(i), \ \ \mathsf{inv}(\mathsf{pk}(i)), \ \ \{secret\}_{\mathsf{pk}(a)}, \ \ \{secret\}_{\mathsf{pk}(b)} \Big\}$$

where g is a public function (i.e., $g \in \Sigma_P$).
Prove formally that the intruder can learn the message "*secret*".
[**Hint:** if you cannot recall the formal proof notation, give a detailed informal argument.] [25%]

e) Consider the following skeleton of a key exchange protocol using a trusted server S.

Step 1: $B \longrightarrow A: \ B, A, \{\!|B, A, \exp(g, Y)|\!\}_{\mathsf{sk}(B,S)}$
Step 2: $A \longrightarrow S: \ \{\!|A, B, \exp(g, X)|\!\}_{\mathsf{sk}(A,S)}, \{\!|B, A, \exp(g, Y)|\!\}_{\mathsf{sk}(B,S)}$
Step 3: $S \longrightarrow A: \ \ldots$
Step 4: $A \longrightarrow B: \ \ldots$

(i) Complete the protocol, i.e., what messages need to be sent in *Step 3* and *Step 4* to ensure that A and B can use the Diffie-Hellman half-keys to compute a shared full-key. [15%]

(ii) In this protocol, the protocol designers decided to use Diffie-Hellman half-keys. Name the property that is established by this and explain briefly against which attack the Diffie-Hellman half-key exchange protects the users of this protocol. [15%]

4. Application Security

a) Consider the following vulnerability in SearchBlox, an enterprise search and data analytics service (CVE-2015-0970):

> A cross-site request forgery (CSRF) vulnerability in SearchBlox Server before version 8.2 allows remote attackers to perform actions with the permissions of a victim user, provided the victim user has an active session and is induced to trigger the malicious request.

What CVSS v3 Base Vector
(AV:[N,A,L,P]/AC:[L,H]/PR:[N,L,H]/UI:[N,R]/S:[U,C]/C:[H,L,N]/I:[H,L,N]/A:[H,L,N])
would you assign to this vulnerability? Provide a brief justification of your assessment. [30%]

b) Assume that registering as a *new user* at a web site, you use

`jii6'Aeje`

as your password. This results in a database error.

Name two vulnerabilities or bad practices that this website probably has and briefly justify your assumption. [15%]

c) A web application is using the following function for protecting itself against Cross-site Scripting (XSS):

```
function sanitiseXSS(str){
    var re = new RegExp("^([a-z0-9]*)$");
    var retval = ""
    if (re.test(str)) {
      retval = str;
    } else {
      console.log("Invalid␣Input");
    }
    return retval;
}
```

(i) Explain briefly if this sanitisation function successfully protects an application against client-side XSS or not. [5%]

(ii) Explain briefly if this sanitisation function implements black listing or white listing. [5%]

QUESTION CONTINUED ON THE NEXT PAGE

d) Consider the following Java program

```
String mname = request.getParameter("month");
if (!mname.matches("[a-zA-Z ;&']+")) {
  throw new IllegalArgumentException();
}
PreparedStatement pstmt = conn.prepareStatement
       ("SELECT username, startdate, transaction,amount
         FROM transaction_history WHERE month = ?");
pstmt.setString(1,mname);
Runtime.getRuntime().exec("cal -m " + mname);
pstmt.execute();
pstmt.close();
```

(i) What vulnerability does this program have? Briefly explain the vulnerability in general and, in particular, why this program is vulnerable.
[**Hint:** You can ignore missing error/exception-handling.] [15%]

(ii) Rewrite this program to fix the vulnerability. [10%]

e) Consider the well-known example for SQL injections where SQL expressions are constructed by concatenating strings, e.g.:

```
String sql="select * from where name = '" + name + "'";
```

where `name` is a variable that can be influenced by an attacker.

(i) Explain briefly *mutation-based* fuzzing. [5%]

(ii) Using the strings
- `"OR␣1=1"`,
- `"--"`,
- `"'"`,
- `"␣"`,

as seed, write (as pseudo-code) a mutation strategy that will detect the SQL injection vulnerability in the example code above. [15%]

END OF QUESTION PAPER