NATIONAL UNIVERSITY OF SINGAPORE

CS2107 — INTRODUCTION TO INFORMATION SECURITY

(Semester 1: AY2022/23)

Time Allowed: 2 Hours

INSTRUCTIONS TO STUDENTS

1. Please write your **Student Number** only. Do not write your name.
2. This assessment paper contains **TWENTY FIVE (25) questions** in **TWO (2) parts**; and comprises **FIFTEEN (15) printed pages**, including this page.
3. Answer **ALL** questions.
4. Write your answer on this question paper.
5. This is an **OPEN BOOK** assessment.
6. You may use **NUS APPROVED CALCULATORS**.

Student Number: __ __ __ __ __ __ __ __ __

This portion is for examiner's use only:

Question	Full Marks	Marks	Remarks
Part A	20		
B-1	7		
B-2	6		
B-3	6		
B-4	5		
B-5	6		
Total	50		

Part B [30 marks] (Scenario-based Questions):

(1) **[7 marks] Cryptographic Techniques & Attacks**

(a) **Padding Oracle Attack (3 marks):** Consider the Padding Oracle attack as described in the lecture notes. Suppose the attacker already knows that the 16-byte plaintext is the following byte sequence:

$$\langle b_1, b_2, b_3, b_4, b_5, b_6, b_7, \texttt{02}, \texttt{01}, \texttt{00}, \texttt{06}, \texttt{06}, \texttt{06}, \texttt{06}, \texttt{06}, \texttt{06} \rangle$$

where the bytes are shown in hexadecimal representation, and the attacker does not know the value of the b_is yet.

Now, write the algorithm to determine the value of $\mathbf{b_7}$, including by correctly setting the IV' values to be sent to the padding oracle.

```
For x = 0 to FF
      Let IV' = IV XOR <0,0,0,0,0,0,x,8,B,A,C,C,C,C,C,C>
      Send the two block query (IV' || c) to Padding Oracle
      If Oracle gives YES, then output: A XOR x
End for loop
```

(b) **Hash Construction using Block Cipher (4 marks):** Bob wants to develop a hash function using AES in CBC mode. Let K be a fixed AES key and IV be a fixed IV. Unlike in a standard encryption setting, both K and IV are now made public (i.e. known to everyone) so that the generated hash digests can be verified by everyone using public K and IV. If a padding is needed, the message can just be padded with 0s.

Bob's hash function works as follows:

- Encrypt the (padded) message with AES in CBC using K and IV.
- Output the ciphertext's *final block*, c_n, as the hash digest of the message.

Note that the digest length is 128 bits since the block size of AES is 128 bits. Can you tell Bob that his hash function is *not* collision resistant?

(***Hint***: You can see if you can select two different messages having the same digest. That is, suppose the hash digest of a message m as outputted by Bob's hash is c_n. Can you find another message m' that hashes into c_n as well, i.e. $h(m') = c_n$? This m' does *not* need to be of the same length as m.)

Let m be 1 block, m' be 2 blocks. First block of m' is same as m, second block of m' we call m2.

Ek(m XOR IV) = c1
We want: m XOR IV = m2 XOR c1
Thus, m2 = m XOR IV XOR c
Then, m' = m || m2, and m and m' have the same digest.

(2) **[6 marks] Authentication System**

Our lecture has described a **strong unilateral authentication protocol** using secret-key based challenge and response. The protocol between a user A and the server S can be written as follows (assuming A and S have shared a secret key k, and agreed on an encryption scheme such as AES):

1. $A \rightarrow S$: "Hi, I'm A".
2. $S \rightarrow A$: $y = E_k(m_S)$, with m_S as a message randomly picked by S.
3. $A \rightarrow S$: $m_A = D_k(y)$.
4. S: If $m_A = m_S$, accept the authentication;
   otherwise allow A to redo **Step 3 for x (the number of allowed attempts) times**.

Suppose a server employs this protocol to authenticate its users. A user's password is used as the secret key k. The server, however, fails to ensure that a user should not select a weak password contained in a weak password list, which is also known to Mallory. Alice, a user of the server, happens to use a weak password listed on the list.

(a) (2 marks) Mallory wants to impersonate Alice. Mallory runs the authentication protocol by pretending to be Alice. He doesn't know the key k, but intends to launch a dictionary attack by testing each entry on the weak password list. If the server doesn't limit the number of failed login attempts, and doesn't add a delay after a failed login attempt, explain succinctly whether or not Mallory should be able to find Alice's password.

Yes, Mallory will eventually find Alice's password when authenticated by S.

(b) (2 marks) Suppose the server limits the number of failed login attempts to 3 times. That is, after 3 failed login attempts on every randomly picked m_S, the server rejects the authentication attempt. Explain succinctly whether or not Mallory should be able to find Alice's password.

Mallory can simply restart the authentication protocol from Step 1 again and continue trying different entries in the weak password list. Since k does not change, eventually Mallory will find Alice's password.

(c) (2 marks) Mallory manages to listen to one Alice's successful authentication with the server. Explain succinctly how Mallory can find Alice's password even if the server limits the overall number of failed login attempts to a small number. Your answer can just tell what Mallory should do by referring to the protocol step(s) above.

Mallory can sniff step 2,3,4. If step 4 shows a successful authentication, take y from step 2 and mA from step 3 and perform a known plaintext attack by testing all the passwords in the entry list and checking which one satisfies mA = Dk(y)

(3) **[6 marks] (Network Security Protocol)**

One day, Bob visited Alice's office building. Alice's company was generous enough to allow Bob to plug his laptop into the company's wired local area network (LAN). When Bob was left alone, he run a packet sniffer tool on the company's broadcast-based LAN and managed to capture Alice's network traffic.

(a) (2 marks) Suppose Alice visited a forum site over HTTP. Could Bob know the username and password that Alice entered to this forum site? Explain briefly why or why not.

Yes, HTTP is unencrypted.

(b) (2 marks) Suppose Alice also accessed an Internet banking site over HTTPS. Could Bob know the username and password that Alice entered to the banking site? Explain briefly why or why not.

No. HTTPS is encrypted and Bob as a MITM in a layer below HTTPS, thus he is not able to see the username and password as it is encrypted before it reaches his layer.

(c) (2 marks) On that day, Bob was also given the password to access the company's WPA2-protected WiFi. Bob run a packet sniffer on the company's WiFi as well. An employee named Charlie accessed the WiFi to visit a HTTP site. Could Bob know the username and password that Charlie entered to this site? Explain briefly why or why not.

Since Bob is able to know the WPA2 key, he is able to decrypt communications at the WPA2 (Data link) layer. Thus, even though Charlie's communication is encrypted at the Data link layer, Bob is able to decrypt it. As HTTP is not encrypted, Bob is able to see Charlie's username and password after decrypting at the Data link layer.

(4) **[5 marks] (Secure Programming)**

Consider the following C code snippet. Some lines denoted by "..." are not shown as they are irrelevant to this question.

```
void process_arguments(char *inputstr, char *inputlen) {
   char buffer[20] = "";
   int length;  /* integer value of inputlen */

   ...
   ⟨a string operation⟩
   ...

}

/* Take 2 string arguments from user: a string, and the length of the string */
int main(int argc, char *argv[]) {

   ...
   process_arguments(argv[1], argv[2]);
   ...

   return 0;
}
```

Pay attention to the line marked with "⟨*a string operation*⟩" in the above code snippet. The following are some possible string operations. Tell whether each operation is safe or unsafe, and briefly cite the reason (including what software vulnerability that can exist if unsafe).

(a) (1 mark) `strcpy (buffer, inputstr);`

Unsafe. strcpy can lead to buffer overflow.

(b) (1 mark) `strncpy (buffer, inputstr, strlen(inputstr));`
(Note that in C, `strlen()` takes a string as an argument, and returns its length.)

Unsafe. strlen(inputstr) may be >20 char

Buffer overflow.

(c) (1 mark) `strncpy (buffer, inputstr, length);`

Unsafe. length may be >20.

Buffer overflow

(d) (1 mark) `strcat (buffer, inputstr);`
(Note that in C, `strcat()` contcatenates/joins two strings. The function is defined as follows: `char *strcat(char *destination, const char *source)`.)

Unsafe. inputstr may be >20 char.

Buffer overflow

(e) (1 mark) `strncpy (buffer, inputstr, 19); buffer[19] = '\0';`

Safe. Maximum 19 char is copied from inputstr to buffer.

(5) **[6 marks] Web Security**

(a) **Same-Origin Policy (3 marks):** Suppose you are writing JavaScript inside the HTML file of `https://cs2107.comp.nus.edu.sg/content/index.html`. Under the Same-Origin Policy (SOP), identify the URL(s) from below whose objects/resources can be accessed by your JavaScript. (By default, HTTP runs over TCP port 80, while HTTPS runs over TCP port 443)

URL 1: `https://cs2107.comp.nus.edu.sg/content/cryptography.html`

URL 2: `http://cs2107.comp.nus.edu.sg/content/cryptography.html`

URL 3: `https://cs2107.comp.nus.edu.sg/others.html`

URL 4: `https://cs2107.comp.nus.edu.sg/content/others.html`

URL 5: `cs2107.comp.nus.edu.sg:80/content/index.html`

Mention all the applicable URL numbers in the box below.

1, 3, 4

(b) **SQL Injection Attack (3 marks):** In a web application, the session-ID of each user is stored in a database table named `usersessions`. The table has three fields: `username` (string), `sessionid` (string), and `timestamp` (integer). Below is the PHP code snippet that sets a SQL query to find the `username` associated with the value of `sid` cookie in a user's request (note that "." is the concatenation operator in PHP):

```
$query = "SELECT username FROM usersessions WHERE sessionid = '" .
         $_COOKIE["sid"] .  "'";
```

There's no validation done by the web application on the value of `sid` cookie. Suppose you know that the `username` of the admin user is `admin`. Now, you want to update the `sessionid` of the admin user in the table to `1234567890` and the `timestamp` to 281122. Explain succinctly how you can accomplish your task.

Notes: You can assume "`--`" as a comment starter in SQL. The syntax of an UPDATE statement in SQL is: `UPDATE <table> SET <field1>=<value1>, <field2>=<value2>, ... WHERE <condition>`. If still necessary, you can state your own other assumptions regarding the target web application.

Set the sid cookie to the value:

'; UPDATE usersessions SET sessionid = '1234567890',
timestamp = 281122 WHERE username = 'admin'; --

BLANK PAGE

(You can use this page if you need more space to write down your answers)

— END OF PAPER —