NATIONAL UNIVERSITY OF SINGAPORE

CS2107 — INTRODUCTION TO INFORMATION SECURITY

(Semester 1: AY2020/21)

Time Allowed: 2 Hours

INSTRUCTIONS TO STUDENTS

1. This assessment paper contains **THREE** questions and comprises **SIXTEEN** printed pages.
2. Answer **ALL** questions.
3. Write your answer within the given box in each question on this question paper.
4. This is an **OPEN BOOK** assessment.
5. You may use **NUS APPROVED CALCULATORS**.
   Nonetheless, you should be able to work out the answers without using a calculator.

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

This portion is for examiner's use only:

Question	Full Marks	Marks	Remarks
Q1	10		
Q2	10		
Q3	25		
Total	45		

1. **[10 marks] (Terminology):** The following ten security-related descriptions are obtained from the Web. Fill in the blanks ***on this question paper*** below with the **most** appropriate term from the given list, which is divided into several groups just for your easier reference. Note that some choices *may appear more than once* in this part. You may ignore any grammatical rules on plural forms.

   [***Cryptographic Notions***]: Confidentiality; Integrity; Availability; Authenticity

   [***Cryptographic Objects***]: Public key; Private key; Digital signature; MAC

   [***Security Protocols***]: Key exchange; WPA2; IPSec; Station-to-Station

   [***Vulnerabilities/Attacks***]: Buffer overflow; Integer overflow; XSS; CSRF; SQL injection; Clickjacking; Privilege escalation; Side channel; Covert channel; Zero-day; Typo squatting; Click fraud; Phishing; Pharming

   [***Miscellaneous***]: Fuzzing; Mandatory Access Control; Discretionary Access Control; Intermediate access control; Role-based access control; Protection rings; Reference monitor; Address randomization

   (i) In 2016, a Domain Name System (DNS) provider Dyn was attacked by a series of DDoS attacks using numerous DNS-lookup requests from tens of millions of IP addresses. As a result, the [Availability] of major Internet platforms and services catering to many users in Europe and North America was affected.

   (ii) [Buffer overflow] vulnerability allows for a possible writing outside the bounds of a block of allocated memory. When exploited, this vulnerability can corrupt data, crash the program, and even cause the execution of malicious code.

   (iii) To correctly enforce a system's access control policy, a/an [Reference Monitor] ideally must be invoked to mediate all security-sensitive operations, must not be tampered, and has undergone complete analysis and testing to verify its correctness.

   (iv) Although it is rather difficult to manage, [Mandatory Access Control] is usually justified when used to protect highly sensitive information, such as certain classified government/military information. When it is enforced, an access to a resource is allowed if and only if system-wide rules exist that allow the access to proceed.

   (v) To deal with [SQL injection] attacks, a filter can be deployed to block or sanitize any presence of, among others, "OR" and "UNION" strings in user-inputted texts.

(vi) IPSec authenticates and/or encrypts IP packets in order to provide secure communication between two computers on the Internet.

(vii) In 2017, an Italian citizen was charged due to his creation and perpetration of a global botnet, which he used to mimic clicks on website advertisements for his advertising revenue. Because of his committed Click fraud attack, he was extradited to the US.

(viii) CSRF attacks trick end users to execute unwanted actions on a web application in which they're currently authenticated.

(ix) Privilege Escalation exploits a bug, design flaw, or configuration oversight in an OS or application to perform unauthorized actions on resources that are normally inaccessible by an application or user. The actions thus go beyond what are actually intended by the application developer or system administrator.

(x) Fuzzing is useful to software testing by providing random inputs and then checking the behavior of a tested application or system, or seeing whether it will crash.

2. **[10 marks] (Multiple Choice Questions):** Choose the **best** answer, and circle/cross the corresponding *letter choice* on this **paper**. No mark is deducted for wrong answers.

(i) Which of the following cryptographic primitives *cannot* provide integrity?

(a) Unkeyed hash
(b) MAC
(c) Digital signature
(d) Keyed hash
(e) Stream cipher

e

(ii) Suppose Alice and Bob perform a Diffie-Hellman Key Exchange with pre-agreed $g = 2$ and p is a prime number larger than 2^{12}. Alice, however, sends 8 to Bob; and Bob sends 16 to Alice. What is the key agreed by Alice and Bob?

(a) 8
(b) 24
(c) 128
(d) 1024
(e) 2^{12}

$8 = 2^a \bmod p$
$16 = 2^b \bmod p$
$a = 3$
$b = 4$
$k = 8^4 \bmod p$
$k = 16^3 \bmod p$
$k = 4096$

e

(iii) Suppose an attacker manages to obtain the `/etc/shadow` file of a Linux server, which stores the hashed and salted passwords of all users in that server. Assuming that a strong standard hash function and random salts are used by the server, what *can't* the attacker do using the password shadow file still?

(a) The attacker can't know the salt of the superuser
(b) The attacker can't know the salt of normal users
(c) The attacker can't know the hash function employed by the server
(d) The attacker can't attempt to find out the superuser's password by trying all entries in a weak-password list
(e) The attacker can't use his pre-computed lookup table containing the digests of all entries in a weak-password list

e

(iv) The following are some network related tools, which are shown together with their corresponding intended usage. Which tool below is shown with an *incorrect* intended usage?

(a) John the Ripper for offline password cracking

(b) Wireshark for network packet capturing
(c) Wireshark for network packet analysis
(d) Nslookup for port scanning
(e) Traceroute for network packet tracing

d

(v) Suppose the TCP/IP software module of Bob's Windows machine is already compromised by a malware. Bob, however, *always* uses HTTPS and also WPA2 to communicate with his Internet banking server. Further assume that Bob machine's TCP/IP software module is *independent* from the modules deploying HTTPS and WPA2, which still function securely. What can we conclude about the security (i.e. confidentiality and authenticity) of Bob's sent messages to the server, or its communication with the server?

(a) The malware can know Bob's communicated banking data as it resides higher than WPA2 in the protocol stack
(b) The malware can know Bob's communicated banking data as it resides higher than HTTPS in the protocol stack
(c) The malware can know Bob's communicated banking data as it resides lower than HTTPS in the protocol stack
(d) The malware cannot know Bob's communicated banking data becasue of WPA2 encryption
(e) The malware can know the IP address of the banking server

e

(vi) What is *incorrect* about the superuser in Linux/UNIX environment?

(a) It may have a UID other than 0
(b) Its username can be other than `root`
(c) It is the owner of the `/etc/shadow` file
(d) It can set an executable program to be a set-UID-root program
(e) It can access all objects in the system

a

(vii) The following countermeasures can help deal with buffer-overflow attacks on the call stack at *run time*, *except*:

(a) Using stack canaries
(b) Deploying (memory-segment) address randomization
(c) Making the stack non-executable
(d) Copying the function's return address in the stack into a safe memory area upon a function entry; and then checking that the return address in the stack still matches after the corresponding function exit

e

(e) Replacing `strcpy()` with `strncpy()` in a correct manner

(viii) **The last 3 (three) questions refer to the following given scenario.**
Alice (UID=2000) wants to set user access to an executable file named `myprogram`, which she owns, with the following permission requirements:

- The file should be readable, writable, and executable by herself.
- It should be readable, *un*writable, and executable by other users in her group.
- Then, it should be *un*readable, *un*writable, and *in*executable by the other remaining users.
- Lastly, the file's set-UID is on/set.

Which command should Alice run?

(a) `chmod 755 myprogram`
(b) `chmod 750 myprogram`
(c) `chmod 4750 myprogram`
(d) `chmod 2750 myprogram`
(e) `chmod 7504 myprogram`

c

(ix) Now, suppose Bob (UID=3000) is *not* in Alice's group. Bob can see Alice's `myprogram` executable file, and then invokes the executable in order to execute it. What will Bob observe after his invocation action?

(a) The program will run with real UID = 3000 and effective UID = 3000
(b) The program will run with real UID = 3000 and effective UID = 2000
(c) The program will run with real UID = 2000 and effective UID = 3000
(d) The program will run with real UID = 3000 and effective UID = 0
(e) The Linux shell will reject Bob's executable invocation

e

(x) Lastly, Charlie (UID=4000) is in Alice's group. Charlie can see Alice's `myprogram` executable file, and then invokes the executable in order to execute it. What will Charlie observe after his invocation action?

(a) The program will run with real UID = 4000 and effective UID = 4000
(b) The program will run with real UID = 4000 and effective UID = 2000
(c) The program will run with real UID = 2000 and effective UID = 4000
(d) The program will run with real UID = 4000 and effective UID = 0
(e) The Linux shell will reject Charlie's executable invocation

b

3. **[25 marks] (Scenario-based Questions):**

(i) **[5 marks] (Encryption Scheme's Requirements)**

Bob wants to devise a cipher that encrypts each letter in the plaintext into another letter. For Bob's need, the alphabet consists of only the 26 lowercase letters, i.e. $\{a, b, ..., z\}$.

Bob knows that the Shift cipher is insecure. Yet, he still likes the idea of mapping a number in $\{0, 1, ..., 25\}$, which represents a letter in the alphabet, into another number in $\{0, 1, ..., 25\}$ using a mathematical operation. For his cipher's encryption, Bob now uses a multiplication operation as opposed to the Shift Cipher's addition operation. That is, Bob defines his encryption of a number x by key k (as the selected factor) as follows: $E_k(x) = (x \cdot k) \mod 26$.

(a) (2 marks) Bob wants to verify that his cipher does work, and also meet the requirements of a working cipher. Suppose he considers his key $k = 2$ for the encryption, and checks all possible encryption outputs. Tell something about Bob's cipher with this particular selected key. Is it a *working cipher* w.r.t. its encryption process, and why is that so? If not, tell what cipher requirement/property is unmet, for instance, by giving a counterexample.
(**Note:** You can omit the security-strength analysis of the cipher.)

Not a working cipher. 2 letters in plaintext can be mapped to the same letter in the ciphertext, thus decryption is not possible.

Example: (0*2) mod 26 = 0, (13*2) mod 26 = 0

(b) (2 marks) Bob now chooses his key $k = 3$ for the encryption, and again checks all possible encryption outputs. Tell something about Bob's cipher when this key is selected. Is it a *working cipher* w.r.t. its encryption process, and why is that so? If not, tell what cipher requirement/property is unmet, for instance, by giving a counterexample.

It is a working cipher as every plaintext letter maps to a unique ciphertext letter

Thus there exists some D() such that Dk(Ek(m)) = m

The security requirement is met, as given the ciphertext, with a random string of numbers, it is "difficult" to derive useful information of the key k and the plaintext x.

The performance requirement is met as the encryption & decryption processes can be efficiently computed

(c) (1 mark) If any of the two keys above seems to work, specify what the corresponding decryption operation is.
(**Hint:** You can reuse the multiplication and modulo operations if suitable/needed.)

If (Ek(x)+26) mod k = 0, x = (Ek(x)+26) / 3
If (Ek(x)+26) mod k = 1, x = (Ek(x)+2*26) / 3
If (Ek(x)+26) mod k = 2, x = (Ek(x)) / 3

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

Consider the following C program.

```
1   #include <stdio.h>
2   #include <string.h>

3   void copy_input(char *input) {
4      unsigned char buffer[31];
5      unsigned char length = strlen(input);

6      if (length <= 30) {
7         ... /* securely copy input to buffer */
8         printf("Your supplied argument is %s.\n", buffer);
9         printf("Its length is %d.\n", length);
10     }
11     else {
```

```
12        printf("Your supplied argument is longer than requested!\n");
13     }
14  }

15  int main(int argc, char *argv[]){
16     if (argc == 2)
17        copy_input(argv[1]);
18     else
19        printf("Please supply one argument with at most 30 characters.\n");
20     return 0;
21  }
```

The program takes a string argument as input. For its objectives, the program is supposed to perform the following:

i. Safely/securely copy the inputted argument to `buffer`;

ii. Print out the *whole* inputted argument; and

iii. Print out the *actual* length of the inputted argument.

Notice that line 7 has a missing instruction, which is indicated by "...".
(**Hint:** The `strlen()` function takes a string as an argument, and returns its length. The length is of the unsigned integer type, which can take 32 bits on a 32-bit system, and 64 bits on a 64-bit one. Hence, the type's possible values can be greater than 255.)

(a) (2 marks) Examine whether the program will meet its 3 objectives if the missing instruction is: `strcpy(buffer, input);`. Tell which stated objectives are met, and which ones are not. Also succinctly explain why and why not, perhaps by using some input examples.

> i. Not met. strcpy does not check the length and so will copy the entire long input into buffer even though it is longer than buffer, resulting in buffer overflow.
> ii. Not met. The program will crash as the return address is overwritten due to buffer overflow as minimum 256 bytes are needed to bypass the length check, and copying 256 bytes will overwrite the return address in the stack frame
> iii. Not met as strlen() can return values above 255

(b) (2 marks) Examine whether the program will meet its 3 objectives if the missing instruction is: `strncpy(buffer, input, 30);`. Tell which stated objectives are met, and which ones are not. Also succinctly explain why and why not, perhaps by using some input examples.

i. Met. strncpy will only copy max 30 char
ii. If input argument is <31 char then met, else not met as strncpy will only copy 30 char to buffer
iii. Not met if input argument is >255 char

(c) (1 mark) Examine whether the program will meet its 3 objectives if the missing instruction is: `strncpy(buffer, input, strlen(input));`. Tell which stated objectives are met, and which ones are not. Also succinctly explain why and why not, perhaps by using some input examples.

i. Not met. strlen() can return values above 30 if the input is longer than 30 bytes
and so strncpy can copy more than 30 bytes
ii. Not met. The program will crash as the return address is overwritten due to buffer overflow as minimum 256 bytes are needed to bypass the length check, and copying 256 bytes will overwrite the return address in the stack frame
iii. Not met as strlen() can return values above 255.

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

(a) (1.5 marks) A web server can set its cookies with `HttpOnly` attribute. The cookies thus become inaccessible to the JavaScript `Document.cookie` API, and they are sent only to the server. Which web attack(s) can be prevented by this measure? Briefly explains why.

XSS. The malicious javascript is unable to access the cookie.

(b) (1.5 marks) A web server wants to prevent XSS attacks by sequentially performing the following string operations on a user-inputted string:

i. Convert all uppercase characters in the string into their corresponding low-

ercase characters

ii. Remove *an* existence of `<script>`

iii. Remove *an* existence of `</script>`

An attacker knows the 3 steps performed by the target web server. For his proof-of-concept XSS attack, the attacker wants to use the string:

"`<script>alert("bypassed")</script>`".

In order to bypass the filtering mechanism, tell how the attacker should modify/extend the string just by *employing the characters already exist* in his original attack string.

Note that <script> and </script> are only removed once. Thus, just enter it twice

<script><script>alert("bypassed")</script></script>

(c) (2 marks) Suppose Mallory is a user of his company's HR website that has a SQL Injection vulnerability. Further, suppose Mallory has already authenticated himself with the site, and is currently accessing a webpage that allows him to self-update his address and phone no. The self-update page with two editable textboxes looks like the one below, with the corresponding `address` and `phoneno` fields are of *string* type in the `employee` table to be updated:

```
    Edit Employee Profile Information
User Name    : Mallory
Address      : _________________________
Phone No     : _________________________
```

Given the conditions below, how can Mallory possibly use the page above to set a higher salary of 20,000 for himself? Answer this by showing what *input string* that Mallory can set in one of the textboxes above.

- Mallory knows that the updated user profile table also has a field called `salary`, which takes an *integer* number.
- This `salary` field in the database is *type sensitive*: in SQL statements, the field can only be set to a *number*, and cannot be set to a string (within ").
- You can assume `"--"` as a comment starter in SQL.

- The syntax of an UPDATE statement is: `UPDATE` ⟨*table*⟩ `SET` ⟨*field*1⟩`=`⟨*value*1⟩`,` ⟨*field*2⟩`=`⟨*value*2⟩, ... `WHERE` ⟨*condition*⟩. In the shown webpage, the condition used is: `WHERE username='Mallory'`.

(**Note:** If still necessary, you can state your own other assumptions regarding the vulnerable target system.)

Enter this into the Address field:

NUS', salary=20000 WHERE username='Mallory'; --

(iv) **[5 marks] (Firewall Configuration)**

Alice, who's a network administrator, wants to deploy a 2-firewall setting to protect her company's network. The machines in her network include:

- `Web-server`: the company's Web server;
- `Internal`: all internal hosts.

In Alice's DMZ, she wants to put her `Web-server`, which accepts both `HTTP` and `HTTPS` traffic. This `Web-server` should be accessible from the Internet as well as `Internal`.

Additionally, there exist the following requirements:

- `Internal` can access all web servers on the Internet using `HTTP` and `HTTPS`, except a set of banned sites `Banned-sites` which could reduce company employees' work productivity.
- `Internal` can use `ICMP` to ping hosts on the Internet.
- `Attackers`, which are Mallory's machines on the Internet, like to DDoS `Web-server`. Hence, they should not be allowed to access `Web-server` via `HTTP` and `HTTPS`.
- As usual, all other network traffic *must be blocked*.

Similar to your tutorial, you can set up the following network partitioning:

`Internal` ← (IN) F_2 (OUT) → DMZ ← (IN) F_1 (OUT) → Internet

Alice has put the following rules at the **outer/front-end** firewall F_1:

No	*Source IP*	*Source Port*	*Dest IP*	*Dest Port*	*Protocol*	*Direction*	*Action*
1	Attackers	*	Web-server	HTTP, HTTPS	TCP	IN	Deny
2	*	*	Web-server	HTTP, HTTPS	TCP	IN	Allow
3	Web-server	HTTP, HTTPS	*	*	TCP	OUT	Allow
4	Internal	*	Banned-sites	HTTP, HTTPS	TCP	OUT	Deny
5	Internal	*	*	HTTP, HTTPS	TCP	OUT	Allow
6	*	HTTP, HTTPS	Internal	*	TCP	IN	Allow
7	*	-	*	-	ICMP	OUT	Allow
8	*	-	Internal	-	ICMP	IN	Allow
9	*	*	*	*	*	*	Deny

Additionally, Alice has specified the following rules at the **inner/back-end** firewall F_2:

No	*Source IP*	*Source Port*	*Dest IP*	*Dest Port*	*Protocol*	*Direction*	*Action*
1	Internal	*	Banned-sites	HTTP, HTTPS	TCP	OUT	Deny
2	Internal	*	*	HTTP, HTTPS	TCP	OUT	Allow
3	*	HTTP, HTTPS	Internal	*	TCP	IN	Allow
4	*	-	*	-	ICMP	OUT	Allow
5	*	-	Internal	-	ICMP	IN	Allow
6	*	*	*	*	*	*	Deny

Note that, for rules on ICMP packets in the two firewalls above, the source and destination ports are irrelevant, and are thus unchecked.

(a) (1 mark) Which rules of F_1 and F_2 allow an internal host to send an outgoing packet to a non-banned external web server?

F1 - 5
F2 - 2

(b) (1 mark) An internal host tries to access an SSH server on the Internet. Which rule of which firewall *first* blocks the outgoing SSH packet?

F2 - 6

(c) (1 mark) A misbehaving internal host attempts to launch a Smurf flood attack by sending out a broadcast ICMP Ping packet to a vulnerable router on the Internet, which still runs outdated software. Which rules of F_1 and F_2 allow or deny such an outgoing packet-sending operation?

F2 - 4 allow
F1 - 7 allow

(d) (2 marks) As the firewall administrator in charge, Alice needs to ensure that an outgoing ICMP packet sent by any internal host to the Internet cannot spoof the source IP address. How the existing rules of F_1 and F_2 should be modified in order to block any attempted IP address spoofing in outgoing ICMP packets?

F1 - 7 change Source IP to Internal
F2 - 4 change Source IP to Internal

(v) **[5 marks] (Authentication Protocol)**

Bob is designing a lightweight protocol that is meant to provide mutual authentication between two parties A and B, who share a secret key k. The protocol can be described as follows:

1. $A \rightarrow B : A, n_A$
2. $B \rightarrow A : n_B, E_k(n_A)$
3. $A \rightarrow B : E_k(n_B)$

The protocol thus basically works in the following manner. In Step 1, A tells B about its identity and his nonce. In Step 2, B tells A about its nonce and an encrypted version of A's nonce. Lastly, in Step 3, A returns to B an encrypted version of the B's nonce. At the end of the protocol, A and B are supposed to believe that the other party in the protocol indeed knows the shared key k. Note that, in the protocol, both parties do keep track of all nonces that they have received from the other party, and will not accept a repeated incoming nonce.

(Hint: Notice that A and B can also have multiple sessions open at the same time.)

(a) (3 marks) The protocol is insecure and can be attacked. Do explain succinctly how it can be attacked by Mallory, who doesn't know k. (If necessary, you can state your own other assumptions about the protocol and parties involved.)

Mallory can authenticate even without knowledge of k
1. Send M, nM to B
2. B sends back nB, Ek(nM)
3. Start a new session with B
4. Send M, nB to B
5. B sends back nB2, Ek(nB)
6. In the first session, send Ek(nB) back to B
7. The first session is authenticated

(b) (2 marks) Explain briefly how you can fix the protocol in order to prevent your described attack.

Modify steps 2 and 3 to include the responder's identity
e.g.
1. A->B: A, nA
2. B->A: nB, Ek(nA, B)
3. A->B: Ek(nB, A)

BLANK PAGE

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

— END OF PAPER —