Mark: 8.25/20 (total score: 12.1/29)

+71/1/40+

Computer Security Foundations | **LEIC**
11/11/2022 | **Duration: 1H** | **Midterm**

This is a multiple-choice test that will be corrected automatically. Please follow these rules:

- Mark your answer using only **blue** and **black** pen. No pencil or light-coloured pens.
- Check only inside each box and be generous on ink. Erased boxes will not be detected automatically.
- Only the boxes matter for the automatic correction. You may underline text or take notes on the sides.

The test is marked for 20 points. There are 30 questions in total, each with 4 options. Each question is worth 20/30 points. Students can check one or two choices per question. The scoring for each question is as follows:

- One checked correct answer (100%).
- One checked incorrect answer (-20%).
- No checked answers (0%).
- Two checked answers, one correct (50%).
- Two checked answers, none correct (-20%).
- More than two checked answers (-20%).

☐0	■0	☐0	■0	■0	☐0	☐0	☐0	☐0
☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1	☐1
■2	☐2	■2	☐2	☐2	☐2	☐2	■2	☐2
☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3	☐3
☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4	☐4
☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5	☐5
☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6	☐6
☐7	☐7	☐7	☐7	☐7	■7	☐7	☐7	☐7
☐8	☐8	☐8	☐8	☐8	☐8	☐8	☐8	■8
☐9	☐9	☐9	☐9	☐9	☐9	■9	☐9	☐9

← code your 9-digit upYYYYXXXXX student number horizontally on the left, and replicate it below. Write also your first and last name below.

Student Number:

up 2 0 2 0 0 7 9 2 8

First and Last Name:

Matilde Silva

Group 1 Introduction *(4 questions)*

Question 1.1 ♣ Which of the following concepts is **not** associated with the risk management approach to security?

0.5/1

[✓ Mitigation.] ✓
☐ Threat analysis.
☐ Cost/benefit analysis.
☒ Security proof. ✓

1/1

Question 1.2 ♣ Pick the **incorrect** statement or indicate that all are correct.

- [ ] A trust model describes which assumptions we can rely on to build security. ✓
- [✓ All are correct.]
- [ ] A security policy describes how security mechanisms are used to realize a security model.
- [ ] A security goal is stated in terms of avoiding loss of value to an asset. ✓

-0.2/1

Question 1.3 ♣ Which of the following options is **not** usually categorised as an exploit?

- [ ] Message that triggers an array access out of bounds.
- [ ] JavaScript performing heap spraying.
- [✓ Overflow in an integer computation.]
- [ ] Text filled into edit box that contains JavaScript.

1/1

Question 1.4 ♣ Which of the following is **not** a common reason for an attacker to compromise a server that answers to requests from a large number of clients/users?

- [ ] Geo-political and strategic motivation.
- [ ] To cause a data breach.
- [ ] As part of a supply-chain attack.
- [✓ To make the machine part of a botnet.] ✓

Group 2 Software Security *(10 questions)*

0.5/1

Question 2.1 ♣ Suppose you intend to use Return Oriented Programming to execute the instructions of functions f_1, f_2, f_3, f_4, f_5 (in some order) and you have placed their addresses in the stack as illustrated below. Which of the following functions could take parameters and the ROP strategy would still work without any changes to the part of the stack shown in the figure?

- [ ] It is not possible to pass any parameters.
- [ ] All functions could take parameters.
- [✓ Function f_3.] ✓
- [ ] Function f_2. ✓

High address	$\&f_3$
	$\&f_1$
	$\&f_5$
	$\&f_4$
Low address	$\&f_2$

-0.2/1

Question 2.2 ♣ A canary built from string terminating bytes offers better security than a random canary because:

- [✓ It makes it more difficult to overwrite the stack.]
- [ ] They can be generated much more efficiently.
- [ ] It is harder to guess than a random canary. ✓
- [ ] Random bytes can be all 0.

Question 2.3 ♣ Recall what you have studied about the classical configuration of the stack region managed by a function f that is called by a function g. Indicate which of the following options is correct for the positioning of the following data pieces: ① **local variables of** f, ② **frame pointer of** g, ③ **parameters passed by** g, ④ **return address back to** g.

-0.2/1

- ☐ ③ ⇒ A, ④ ⇒ B, ② ⇒ C, ① ⇒ D
- ☐ ① ⇒ A, ③ ⇒ B, ② ⇒ C, ④ ⇒ D
- [✓ ③ ⇒ A, ② ⇒ B, ① ⇒ C, ④ ⇒ D]
- ☐ ④ ⇒ A, ③ ⇒ B, ① ⇒ C, ② ⇒ D

High address	A	4
	B	3
	C	1
Low address	D	2

Question 2.4 ♣ One can bypass Address Space Layout Randomization (ASLR) protections by:

1/1

- ☐ Jumping to system code using Return Oriented Programming.
- ☐ Extracting addresses using other vulnerabilities.
- ☐ Trial and error.
- [✓ All of the other choices.] ←

Question 2.5 ♣ Recall the buffer overflow lab that you have performed, where a buffer in a local variable receives an untested input. For the exploitation strategy where the injected code (*shellcode*) is placed at the end of the input bytes, you should:

1/1

- [✓ In the place of the vulnerable function's return address, write another address pointing to a higher place in the stack.] ←
- ☐ None of the other options.
- ☐ In the place of the vulnerable function's return address, write the address of the buffer.
- [✓ In the place of the vulnerable function's return address, write another address pointing to a lower place in the stack.]

Question 2.6 ♣ Recall the buffer overflow lab. You explored a buffer overflow vulnerability by creating a malicious input that overwrites the return address of the vulnerable function with an address within your buffer where you stored *shellcode* to be executed. Which security technique had to be disabled during compilation to allow you to overwrite the return address of the vulnerable function?

-0.2/1

- ☐ None of the other choices
- [✓ Address Space Layout Randomisation]
- [✓ Data Execution Prevention]
- ☐ Stack Canary

Question 2.7 ♣ Consider the code below, which introduced a critical vulnerability in openSSH. An attacker that controls the value of `nresp` can cause a memory management problem because:

1/1

- ☐ It can lead to allocating too much memory.
- ✓ [✓ It can lead to not allocation enough memory.]
- ☐ None of the other choices.
- ☐ It can force the if statement not to execute.

```
nresp = packet_get_int();
if (nresp > 0) {
  response = xmalloc(nresp*sizeof(char*));
  for (i = 0; i < nresp; i++)
    response[i] = packet_get_string(NULL); }
```

Question 2.8 ♣ *Information-flow analysis* encompasses a class of program analysis techniques beyond safety. The C-like program on the right illustrates an example of:

0.5/1

- [✓ Dynamic taint analysis.]
- [✓ Static taint analysis.]
- ☐ Constant-time analysis.
- ☐ Control-flow integrity.

```
int printf(untainted char *fmt, ...);
tainted char *fgets(...);

tainted char *name = fgets(..., network_fd);
printf(name); // FAIL: tainted not untainted
```

Question 2.9 ♣ Certain format strings can make a program crash (with very high probability). Which of the following `printf` commands is very likely to crash the program?

1/1

- ☐ `int n=0; printf("%f%d",n);`
- ☐ `int n=0; printf("%s%x",&n);`
- [✓ `int n=1; printf("%s%d",n);`] ✓
- ☐ `int n=1; printf("%n%f",&n);`

Group 3 Systems Security *(10 questions)*

Question 3.1 ♣ For efficiency reasons, the address space of the Linux kernel is not completely independent from that of user processes due to *kernel mapping*. Which of the following is correct?

-0.2/1

- ☐ Part of the kernel memory space is mapped into the memory space of each user process, to reduce the number of cache misses.
- [✓ Part of the kernel memory space is mapped into the memory space of each user process, to speed up system calls.] ✓
- [✓ Since the discovery of speculative execution attacks such as Meltdown and Spectre, Linux no longer uses *kernel mapping*, as it violates the principle of *separation of privilege*.]
- ☐ Part of each user process's memory is mapped into the kernel memory space, to reduce the size of the address translation tables.

Question 3.2 ♣ Remember the environment variable and `setuid` program lab. The Linux file system associates a owner, a group, and 12 permission bits with each entry. For a file with the 12th bit (the Set-UID bit) active, it can be executed:

1/1

- [✓ Only by users whose set (owner, group, others) in the permission bits, has the executable bit active.] ✓
- ☐ Only by the owner or by a user belonging to the group.
- ☐ Only by a user belonging to the group.
- ☐ Only by the owner.

Question 3.3 ♣ The UNIX filesystem can be seen as an instance of:

1/1

- ☐ Capability lists, since each file enumerates the permissions for all the users who may read, write or execute it.
- ☐ Access control lists, since only the owner of each file (or root) may read, write or execute it.
- ☐ Attribute-based access control, since users may change their group membership without the need to change file permissions.
- [✓ Role-based access control, since each file has an access control list for three fixed roles (owner,group,others) and the association between users and groups may be modified independently.]

Question 3.4 ♣ In the environment variable and `setuid` program lab, we have experimented with environment variables and `setuid` programs. Which of the following sentences is **not** true?

-0.2/1

- ☒ When a process forks a child process, it passes on its environment, excluding some critical variables if the fork is a system call.
- ☐ When the shell forks a child process, it passes on its environment, excluding some critical variables if it has a different effective user id.
- ☐ If the shell detects that it is being run under a `setuid` process, it may drop its privilege.
- [✓ The `system` function allows calling a shell function within a program with the program's environment.]

Question 3.5 ♣ Which of the above is **not** a systems security principle that we have studied in the classes?

-0.2/1

- [✓ Compromise recording]
- ☒ Closed design
- ☐ Separation of privilege
- ☐ Economy of mechanism

Question 3.6 ♣ Under Linux, Docker crucially relies on `seccomp-bpf` to confine containers. Select the **incorrect** choice or mark that all choices are correct.

-0.2/1

- ☐ Using the default Docker `seccomp-bpf` filters, an attacker that acquires root in the container does not directly acquire root in the host OS.
- ☒ All choices are correct.
- ☐ Users can manually configure `seccomp-bpf` filters to block various system calls inside the container.
- [✓ If an attacker has access to certain system calls inside a container, it can exploit Docker to acquire root in the host OS.]

Question 3.7 ♣ Which fundamental mechanism does a UNIX operating system have in place to enforce isolation?

-0.2/1

- [✓ Software fault isolation, by ensuring that the control-flow of user processes never accesses invalid virtual memory regions.] ✓
- ☐ System call interposition, by virtualising the address space of different user processes, monitoring all virtual address translations.
- ☒ Software fault isolation, by virtualising the address space of different user processes, monitoring all virtual address translations.
- ☐ System call interposition, by preventing different user processes to communicate via system calls.

Question 3.8 ♣ The Android operating system is a particular distribution built on top of Linux. Which is **not** an additional domain-specific restriction that Android implements?

- ● It implements a form of Mandatory Access Control, so that no application can change its permissions.
- ☐ It enforces Manifest Permissions per application, to restrict its system capabilities.
- ☒ There is no root user, to prevent applications from escalating privileges. x
- ☐ It isolates different applications by registering each application with a different UNIX user. ✗

-0.2/1

Question 3.9 ♣ Virtual machines are commonly used as a security mechanism. Which of the following sentences is **not** true?

- ☐ A disadvantage of virtual machines is that malware may exploit the virtualisation layer to remain undetected.
- ☒ An advantage of virtual machines is that software may be transparently executed as in a non-virtualised operating system. ←
- ☐ An advantage of virtual machines is that acquiring root in the virtual machine does not grant root in a host operating system.
- ☐ A disadvantage of virtual machines is that malware may detect that is is being virtualised.

1/1

Question 3.10 ♣ Many modern operating systems require system support for an hardware component called a Trusted Platform Module (TPM). Which is the main rationale for that requirement?

- ☐ To protect the user login process. ←
- ☐ To store user's cookies when browsing the web.
- ☒ To prevent malicious bootloaders from compromising the operating system's boot process.
- ● To guarantee that the user only installs official software. ←

-0.2/1

Group 4 Web Security *(6 questions)*

Question 4.1 ♣ According the Same-Origin Policy (SOP), which of the following is allowed?

- ☐ JavaScript code in a page from origin A can inspect the HTML code of a frame from origin B.
- ☒ HTML code in a page from origin A can send a POST request to a page from origin B. ✓
- ☐ JavaScript code in a frame from origin A can exchange data with a frame from origin B.
- ● HTML code in a page from origin A can send and read the response of a GET request to a page from origin B. ✓

0.5/1

Question 4.2 ♣ Consider the following SQL query, written in some server-side library, that is vulnerable to SQL injection. Which of the following statements is true?

```
uName = getRequestString("username");
uPass = getRequestString("userpassword");
sql = 'SELECT␣*␣FROM␣Users␣WHERE␣Name␣="' + uName + '"␣AND␣Pass␣="' - uPass + '"'
```

- ☐ A SQL injection attack that bypasses authentication is only possible because the clause for the `Name` field appears before the clause for the `Pass` field.
- [✓ A SQL injection attack may be able to delete the `USERS` table.] ✓
- ☐ A SQL injection attack would not be possible if the clauses for the `Name` and `Pass` fields did not enclose strings with double quotes (").
- ☐ A SQL injection attack will only be able to read existing data from the `USERS` table.

1/1

Question 4.3 ♣ Which assignment can be seen as a valid analogy in the table below?

- ☐ ①=Frames,②=DOM,③=Images,④=Sub-frames
- [✓ ①=Pages,②=Cookies,③=HTTP,④=Frames]
- ☐ ①=DOM,②=Cookies,③=iFrames,④=Fetch
- ☐ ①=Pages,②=HTML,③=JavaScript,④=Popups

Systems Security	Web Security
Processes	①
Files	②
Sockets	③
Sub-processes	④

1/1

Question 4.4 ♣ Which sentence best characterises a Cross-Site Request Forgery (CSRF) attack? "When a malicious origin can send requests to a server in another origin ..." :

- [✓ ... and trigger a side-effect on the server.] ✓
- ☐ ... and measure side-channels such as response time.
- ☐ ... where the user is already logged in.
- ■ ... and read its response. ✓

0.5/1

Question 4.5 ♣ A classical protection against Cross-Site Scripting (XSS) attacks is for the site to adopt a Content Security Policy (CSP). Which of the following statements is true?

- [✓ XSS attacks using inline scripts are blocked by a default CSP.]
- ■ DOM-based XSS attacks cannot be prevented with a CSP because they occur on the server-side when processing a user request.
- ☐ Stored XSS attacks cannot be prevented with a CSP because the malicious payloads are already stored in the server.
- ☐ Preventing reflected XSS attacks requires both CSP and SRI (Subresource Integrity).

-0.2/1

Question 4.6 ♣ Cookies may be used for various purposes. Name the **incorrect** one below.

- [✓ Encrypting user communication.]
- ☐ Personalising user experience.
- ☐ Tracking user activity.
- ☐ Managing user sessions.

1/1