VICTORIA UNIVERSITY OF
WELLINGTON
TE HERENGA WAKA

EXAMINATIONS – 2025

TRIMESTER 2

FRONT PAGE

CYBR 271

CODE SECURITY
1 NOVEMBER 2025

Time Allowed: TWO HOURS (120 minutes)

Instructions:

- Attempt **ALL** questions in this booklet.
- The exam is worth 120 marks in total.
- No calculators.
- Printed foreign-to-English language dictionaries are permitted.
- Write answers in the spaces provided in the examination booklet.
- **Hand in the examination booklet.**

Sections	**Marks**
A. Supply Chain & Risk Assessment	[20]
B. Low-Level & System Vulnerabilities	[45]
C. Web Application Vulnerabilities	[55]

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

SECTION A Supply Chain & Risk Assessment

1. Supply Chain **(4 marks)**

   Define the security principle of **non-repudiation**. In the context of a software supply chain, **explain** why it is important when a developer commits new code to a shared repository.

2. Risk Assessment **(16 marks)**

   (a) **(8 marks)** A software company develops a popular mobile application. They integrate a third-party analytics library by downloading it from a public software repository. An attacker compromises the public repository and replaces the legitimate library with a malicious version that contains spyware.

   Explain how this supply chain attack could lead to a risk of "User Data Exfiltration" and **propose two** specific mitigation strategies. For each mitigation strategy, explain what it is, how it is implemented, and how it mitigates the attack.

(b) **(8 marks)** Using the **DREAD** risk assessment model, **calculate** a risk score for the "User Data Exfiltration" risk from question 2(a). **Justify** your rating (from 1 to 10) for each of the five DREAD categories.

SECTION B Low-Level & System Vulnerabilities

3. Privilege Escalation **(15 marks)**

   (a) **(4 marks) Briefly explain** what a **Set-UID** program is in a Unix-like operating system and why a vulnerability in such a program is a high-security risk.

   (b) **(6 marks)** Consider a setuid program owned by `root` that contains the following line of C code:

   ```
   int status = system("ls /home/ian");
   ```

   If an attacker can manipulate the `PATH` environment variable, **explain how** they could exploit this program to execute their own malicious code as the root user. Include the specific steps the attacker would take.

(c) **(5 marks)** **Describe** a specific coding practice that would prevent the `PATH` manipulation exploit described in question 3(b).

4. Buffer Overflow **(15 marks)**

(a) **(6 marks)** **Draw** a diagram of a typical stack frame for a function call on a 32-bit x86 system. **Label** the key components (e.g., return address, saved `ebp`, local variables).

(b) **(4 marks)** **Explain** the purpose of a **NOP sled** in a shellcode injection attack.

(c) **(5 marks) Explain** what **Address Space Layout Randomization (ASLR)** is and how it makes buffer overflow attacks more difficult.

5. Format String **(10 marks)**

(a) **(4 marks)** **Explain** how the `%n` format specifier differs from other format specifiers, and how an attacker can exploit this difference to write a chosen value to an arbitrary memory location in a format string attack.

(b) **(6 marks)** A vulnerable C program contains the following line:

```
printf(userinput);
```

When `printf` processes format specifiers like `%x`, it reads values sequentially from the stack starting from where arguments would normally be passed.

Construct a format string that will:

1. Print exactly 100 bytes using the first two format specifiers
2. Read and display the value at the 3rd argument position in hexadecimal

Explain why controlling the exact number of bytes printed is important for format string exploits involving the `%n` specifier.

SECTION C Web Application Vulnerabilities

6. SQL Injection **(10 marks)**

(a) **(2 marks)** **Explain** the fundamental cause of SQL injection vulnerabilities.

(b) **(4 marks)** **Explain** why **prepared statements** are the most effective countermeasure against SQL injection.

(c) **(4 marks)** A web application uses the following PHP code to authenticate users:

```
$sql = "SELECT * FROM users WHERE name = '" . $_GET['name'] . "'";
```

Provide an input for the `name` parameter that would return all records from the `users` table. **Explain** how your payload works.

7. Cross-Site Scripting (XSS) **(10 marks)**

(a) **(4 marks)** **Explain** the difference between Stored XSS and Reflected XSS.

(b) **(4 marks)** A website uses the following HTTP response header:

```
Content-Security-Policy:script-src 'self' 'nonce-aBcDeF123'
```

Explain how this could prevent some XSS attacks.

(c) **(2 marks)** **Explain** why a secret token (effective against CSRF) is **ineffective** against XSS.

8. Cross-Site Request Forgery (CSRF) **(10 marks)**

(a) **(3 marks)** **Describe** the role of browser cookies in enabling a CSRF attack.

(b) **(4 marks)** An attacker wants to trick a logged-in user into adding the attacker (ID 55) as a friend on `social.com`. The website uses this URL to add friends:

```
http://social.com/add-friend.php?friend-id=55
```

Write the HTML code the attacker would place on their malicious website to automatically make the victim's browser send this request.

(c) **(3 marks)** **Explain** how the **Synchronizer Token Pattern** prevents CSRF attacks.

9. Clickjacking **(10 marks)**

(a) **(4 marks)** **Describe** the core technique of a Clickjacking attack using iframes and CSS.

(b) **(4 marks)** **What** is the recommended HTTP header-based countermeasure for Clickjacking? **Provide** an example of this header.

(c) **(2 marks)** **Explain** why a parent page cannot read the DOM of an iframe from a different origin.

10. Shellshock **(15 marks)**

(a) **(4 marks)** **Describe** the fundamental flaw in how Bash parsed environment variables that caused the Shellshock vulnerability.

(b) **(6 marks)** **Explain** how an attacker could exploit the Shellshock vulnerability by manipulating the HTTP `User-Agent` header when targeting a vulnerable CGI script on a web server.

(c) **(5 marks) Describe** what a **reverse shell** is and **explain** why an attacker would use it after a successful Shellshock exploitation.

11. Prompt Injection **(5 marks)**

(a) **(5 marks) Explain** the core similarity between SQL Injection and Prompt Injection. Then **describe** a Goal Hijacking attack with a concrete example.

Student ID: . . . .

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

* * * * * * * * * * * * * * *