



VICTORIA UNIVERSITY OF  
**WELLINGTON**  
TE HERENGA WAKA

EXAMINATIONS – 2023

TRIMESTER 2

FRONT PAGE

CYBR 271  
SECURE PROGRAMMING  
26/10/2023

**Time Allowed:** TWO HOURS (120 minutes)

**CLOSED BOOK**

**Permitted materials:**

- Only silent non-programmable calculators or silent programmable calculators with their memories cleared are permitted in this examination.
- Printed foreign to English language dictionaries are permitted.
- No other material is permitted.

**Instructions:**

- Attempt **ALL** questions in this booklet.
- This test contributes 35% of your final grade.
- There are 120 marks total.
- Write answers in the spaces provided in the examination booklet.
- **Hand in the examination booklet.**

Questions	Marks
1. Security Principles	[10]
2. STRIDE and Security Cards	[20]
3. Libraries, Addressing and Ranking Threats	[15]
4. Reviews and Security Testing	[5]
5. Privilege Escalation	[15]
6. Buffer Overflow	[20]
7. Format String	[15]
8. Cross-site Scripting (XSS)	[10]
9. Cross-site Request Forgery (CSRF)	[5]
10. SQL-Injection	[5]

1. Security Principles

[10 MARKS]

- (a) **(4 marks)** Consider an Internet banking application where the application allows users to query their balance and transfer money between accounts.

- i. **(2 marks)** Define the security principle *secure data at rest* and give a simple example of the principle in the context of the Internet banking application.

- ii. **(2 marks)** Define the security principle *separation of duty* and give a simple example of the principle in the context of the Internet banking application.

- (b) **(6 marks)** You have been hired to review the security of a new smartwatch that has been designed to detect abnormal heart rhythms. The smartwatch will vibrate when the condition is detected giving time for the wearer to seek emergency care.

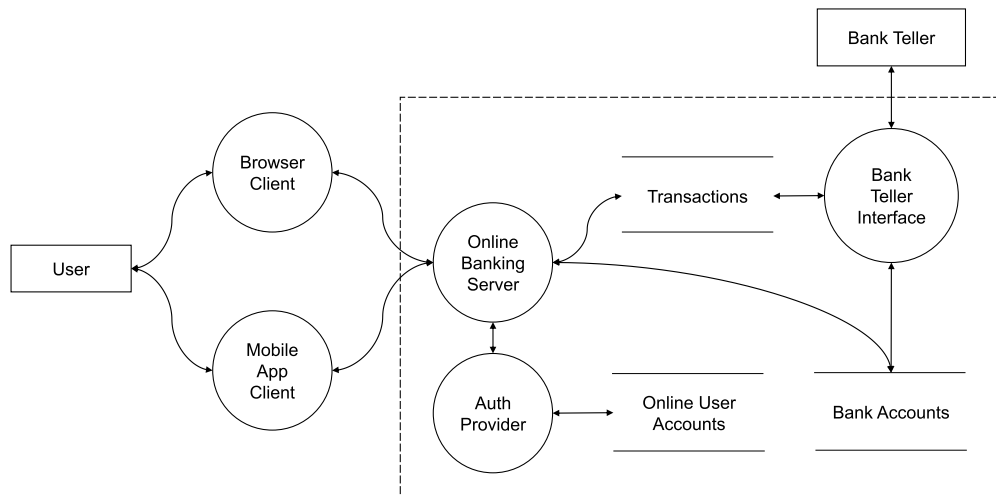
- i. **(3 marks)** The smartwatch collects extra information such as GPS location that is not necessary for its functionality. **Identify** the security principle being violated and **justify** your answer.

- ii. **(3 marks)** When the smartwatch stops measuring heart rhythm due to an error in the software, it doesn't indicate that there is an error to the user. **Identify** the security principle being violated and **justify** your answer.

## 2. STRIDE

[20 MARKS]

Consider the following data flow diagram for an online banking system.



- (a) (3 marks) Name one each of the following: (i) external entity, (ii) process, and (iii) data store shown in the diagram.

- (b) (4 marks) Identify two categories of threats that are applicable to an external entity and give a brief description of each threat.

- (c) (2 marks) Consider the data flow from the Browser Client/Mobile App Client to the Online Banking Server. The data flow consists of transactions (e.g. fund transfers, payments). It traverses the network but is encrypted.

**Describe** an example of a threat applicable to this data flow, and an appropriate mitigation for the mentioned threat.

- (d) **(4 marks)** The Bank Teller Interface is only available in terminals located within the bank (and its branches) and requires special access device issued to authorised bank employees only.

**Describe** a threat where an attacker gains access to the system through spoofing the identity of the Bank Teller. **Propose** two layers of mitigation to this potential attack.

For the first mitigation, **describe** how an attacker can overcome it. The second mitigation must therefore address the weakness of the first mitigation. You need not describe the weakness of the second mitigation.

- (e) **(2 marks)** The Mobile App Client was developed by the bank using a group of trusted developers. Provide **two** valid reasons why it is drawn outside the trust boundary.

(f) **(5 marks)** Security Cards represent a useful methodology for fundamental learning to aiding professional threat modelers, with emphasis on creativity and brainstorming.

i. **(4 marks)** List the **FOUR** main questions this method is aiming to provide answers for.

ii. **(1 mark)** What is a limitation of the Security Cards methodology?

3. Libraries, Addressing and Ranking Threats

[15 MARKS]

- (a) (2 marks) Discuss one advantage and one disadvantage of using checklists in addressing threats.

- (b) (2 marks) Suppose you want to find out whether a particular computer desktop application has been implemented using programming approaches which are unsafe.

Which of the libraries discussed in the lectures should you consult? **Justify** your answer.

- (c) (5 marks) Consider a web application for a cinema which allows group booking discounts and has a maximum of fifteen attendees before requiring a deposit. The booking system reserves the seats, unless it is cancelled. Referring to the 2021 OWASP Top 10 Security Risks:

- What kind of risk is present in this web application?
- How can this risk be exploited?
- What is the impact to the cinema?

Hint: The 2021 OWASP Top 10 Security Risks include: Broken Access Control, Cryptographic Failures, Sensitive Data Exposure, Injection, Insecure Design, Security Misconfiguration, Vulnerable and Outdated Components, Identification and Authentication Failures, Software and Data Integrity Failures, Security Logging and Monitoring Failures, and Server-Side Request Forgery.

- (d) **(2 marks)** Briefly explain why Microsoft has moved away from using DREAD to using Bug Bars?

- (e) **(4 marks)** A colleague recommends that you purchase cyberinsurance for your company to transfer the risk associated with ransomware attacks.

What are **TWO** new risks that might arise from taking out cyberinsurance? Assume that you **do not** publicise that you have taken out cyberinsurance.

4. **Reviews and Security Testing**

**[5 MARKS]**

- (a) **(3 marks)** Name the three code review variations.

- (b) **(2 marks)** Name one advantage and one disadvantage of using static source code analysis tools.

**5. Privilege Escalation****[15 MARKS]**

- (a) **(3 marks)** Unix uses the “9-bit permission mechanism” and groups users into different groups where access permission can be managed per group. Using the `ls -l` command, you have got the following output. Identify the different groups and their corresponding permissions regarding the script file.

```
-rw-r-xr-- 1 bob staff 354 12 Sep 04:53 script
```

- (b) **(3 marks)** What values should be used with the `chmod` command to grant the following permissions to the script file?

- **Others:** Cannot read, write and execute.
- **Owner:** Can read, write and execute.
- **Group:** Only allowed to write and execute.

```
$chmod __ __ __ script
```

- (c) **(2 marks)** Consider the program `mycat` which has the following properties shown by the `ls -l` command.

```
-rwxr-xr-x 1 alice user 354 12 Sep 04:53 mycat
```

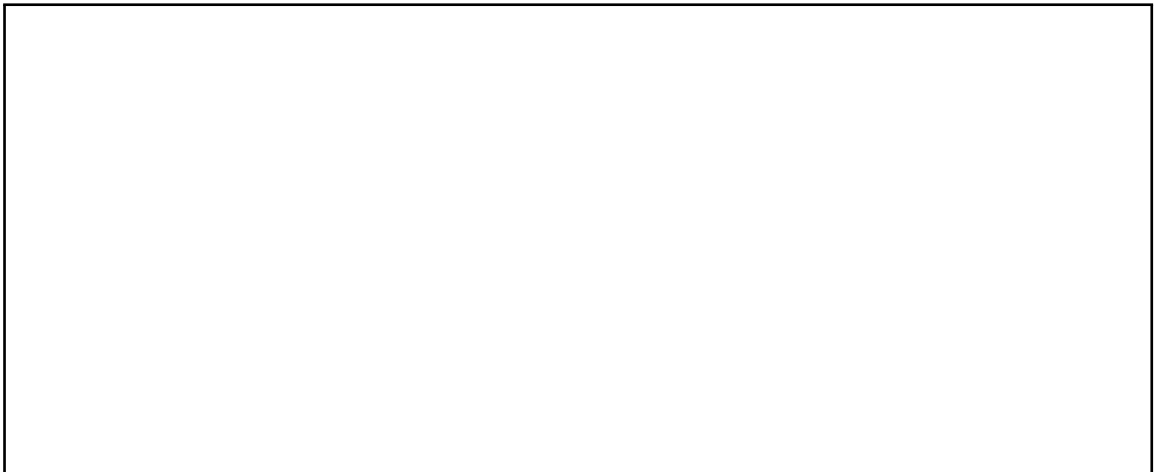
List down the **TWO** commands (in the correct order) that must be executed to convert `mycat` into a Set-UID program.

(Hint: You must use `sudo`.)

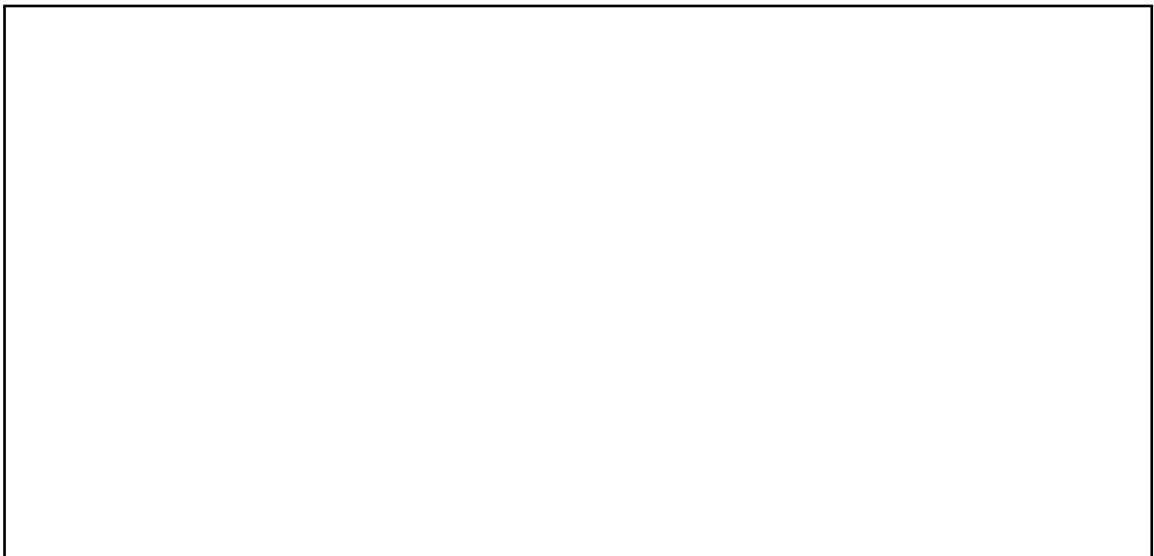
- (d) **(3 marks)** Consider the following C program that uses the function `system()` to execute the external program `cat`. The program expects one command-line argument which should be the name of the text file to be opened and displayed by `cat`.

```
1  #include <stdlib.h>
2  int main(int argc, char *argv[])
3  {
4      char cat[] = "/bin/cat";
5
6      if(argc != 2) {
7          printf("Please specify file to open.\n");
8          return 1;
9      }
10
11     char *cmd = malloc(strlen(cat) + strlen(argv[1]) + 2);
12     sprintf(cmd, "%s %s", cat, argv[1]);
13     system(cmd);
14     return 0;
15 }
```

Assuming the program is Set-UID, how can it be exploited by attackers to potentially escalate privileges? **Describe** how to fix the weakness.



- (e) **(4 marks)** **Rewrite** the above program to fix the problem.



**6. Buffer Overflow****[20 MARKS]**

*Note: Unless stated otherwise, assume a 32-bit computer system is being used.*

- (a) **(10 marks)** Draw the stack frames for the `main()` and `foo()` functions assuming line number 15 is being executed.

```
1  int foo(int w, int x, int y, int z)
2  {
3      int a, b;
4      static buffer;
5      a = x - z;
6      b = a + y;
7      static k=5;
8      return b;
9  }
10
11 int main(int args)
12 {
13     int result;
14     int str = 5;
15     result = foo(args, 3, 2, str);
16     return result;
17 }
```

- (b) **(2 marks)** Calculate the memory address of the “Return Address” field for the `bar()` function. Assume the offset between the start of the `buffer` variable and `ebp` is 64 bytes, and `ebp` is pointing to the address `0xbffea170`.

```

1  int bar(char *str)
2  {
3      int x;
4      char buffer[50];
5      strcpy(buffer, str);
6
7      return 1;
8  }

```

- (c) **(2 marks)** For each of the following variables, specify the part of memory in which it will be located.

1. `static int buffer = 7;`
2. `int *ptr = (int *) malloc(5 * sizeof(int)); //(*ptr)`
3. Initialised local variables
4. Uninitialised global variables

- 1.
  - 2.
  - 3.
  - 4.

- (d) **(2 marks)** Consider buffer overflow countermeasures. Answer the following based upon what you observe in the following figure.

```

[10/26/23] seed@VM$ ./stack
*** Stack smashing detected ***: ./stack terminated
[10/26/23] seed@VM$ █

```

- i. **(1 mark)** State the name of the countermeasure used above.

- ii. **(1 mark)** Briefly discuss how this countermeasure mechanism works.

- (e) **(4 marks)** An attacker has identified a vulnerable program that reads from a file and has created a `badfile` with the aim of getting a Unix shell via a *buffer overflow attack*. The attack surface is the `strcpy()` function that copies the user input into a buffer.

After checking the memory layout for the compiled code using the debugging mode (gdb), they prepared a `badfile` and injected the new return address based on `ebp+96` (hex), where `ebp` is `0xbffffb6a`.

They have turned off address randomisation, made the stack executable and disabled all stack countermeasures. Their attack should be successful; however, a `Segmentation fault error` message has been received instead of a Unix shell when the vulnerable program is executed on the input.

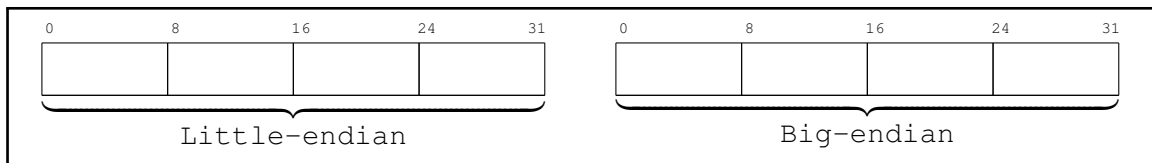
- i. **(1 mark)** Briefly explain the role of the NOP operation (0x90) in the attack.

- ii. **(3 marks)** Explain the main reason why the attack failed.

**7. Format String****[15 MARKS]**

*Note: Unless stated otherwise, assume a 32-bit computer system is being used.*

- (a) **(2 marks)** What is the “Little-endian” and “Big-endian” representations of 0xFF89C39A?



- (b) **(3 marks)** The *width modifier* allows you to control the number of characters to be printed out. Using the %x and %n specifiers along with the width modifier, provide a format string to write the number 250 (decimal) into a variable on the stack if you know its address is 12 bytes above the va\_list pointer.

- (c) **(5 marks)** An attacker targeted a format string vulnerability on a system, where the aim was to print out a secret value stored on the system. Based on some investigations, the attacker was able to find the distance between the address of the secret value and the va\_list pointer. Accordingly, they provided 15 %s specifiers that is enough to advance the pointer to the right position and print out the value.

- i. **(3 marks)** Explain why her attack was not successful.

- ii. **(2 marks)** How can you fix the problem and make the attack succeed?

- (d) **(5 marks)** An attacker decided to exploit the formatting string vulnerability to execute her malicious code. She needs to replace the value of the Return Address with one of the NOP's addresses. She decided to use the length modifier %hn and adopt the fast approach to injecting the address of her malicious code into the Return Address.

Write a single formatting string she needs to use to successfully inject the value 0xbffff226 (the address of the malicious code) into the address 0xbffff384 (the Return Address) if you know the distance between va\_list and the Return Address is 24 bytes.

## 8. Cross-site Scripting (XSS)

[10 MARKS]

- (a) (2 marks) **Briefly** explain the role of the vulnerable website in persistent XSS.

- (b) (4 marks) A website (cybr271.com) utilises the 'Content Security Policy' (CSP) mechanism to prevent XSS attacks and allows script code to be executed based on the following CSP role.

**Explain** the meaning of this CSP.

```
Content-Security-Policy: default-src 'self'; script-src 'self'
                        'nonce-3fX254s' wgtm.ac.nz
                        'sha256-V2kaaafImTjn8RQTWZmF4IfGfQ7Qsqsw9GWaFjzFNPg=';
```

- (c) (2 marks) To defeat XSS attacks, a developer decides to implement filtering on the browser side. Basically, the developer plans to add JavaScript code on each page, so before data are sent to the server, it filters out any JavaScript code contained inside the data. Let's assume that the filtering logic can remove individual well-formed JavaScript keywords from the data reliably with 100 percent accuracy.

Can this approach prevent **all** XSS attacks? **Justify** your answer.

- (d) (2 marks) The *nonce* and *hashing* mechanisms can be used to allow inline script code to run on a page. Which of the two mechanisms is preferred as a countermeasure against XSS attacks? **Justify** your answer.

9. Cross-site Request Forgery (CSRF)

[5 MARKS]

- (a) **(2 marks)** Adopting the *Secret Token* approach represents an effective mechanism to prevent CSRF attacks, where the server injects a secret into its own page after the user has been authenticated. This secret token must be attached with each subsequent request sent from the user's browser to the server; otherwise, the request might be discarded.

Can this approach (i.e., CSRF tokens) also be used to prevent **stored** XSS attacks? **Justify** your answer.

- (b) **(2 marks)** The fundamental cause of CSRF is that the server cannot distinguish whether the request is initiated from the *same-site* or *cross-site*. One solution is where the browser can help by utilising the *Referer* Header field.

**Briefly** explain the main limitation of this solution.

- (c) **(1 mark)** **Briefly** explain why the use of HTTPS doesn't protect against CSRF attacks?

**10. SQL-Injection****[5 MARKS]**

- (a) **(2 marks)** Data sanitisation (including both filtering and encoding approaches) can mitigate SQL-injection attacks, but does not address the fundamental cause of the problem.

**Briefly** explain why “Prepared Statements” is a perfect solution for this type of attacks.

- (b) **(2 marks)** To defeat SQL injection attacks, a web application has implemented a filtering scheme on the client side. Basically, on the page where users type their data, a filter is implemented using JavaScript. The filter removes any special characters found in the data, such as apostrophes, characters for comments, and keywords reserved for SQL statements.

Assume that the filtering logic does its job, and can remove all the code from the data; is this solution able to defeat SQL injection attacks? **Justify** your answer.

- (c) **(1 mark)** The following SQL statement is sent to the database to add a new user to the database, where the content of the \$name and \$passwd variables are provided by the user, but the EID and Salary field are set by the system.

How can a malicious employee set his/her salary to a value higher than 80000?

Assume:

- username is “smithbob”
- password is “pass123”

```
1 $sql = "INSERT INTO employee (Name, EID, Password, Salary)
2   VALUES ('$name', 'EID6000', '$passwd', 80000)";
```

**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.

Student ID: .....

\* \* \* \* \*