MEMORIAL UNIVERSITY

Faculty of Engineering and Applied Science

Department of Electrical and Computer Engineering
St. John's, NL Canada A1B 3X5
Tel: 709 864 8177 Fax: 709 864 4042
https://www.mun.ca/engineering/ece

ECE 7420 / ENGI 9823: Computer Security

Mid-term test
24 Jun 2024

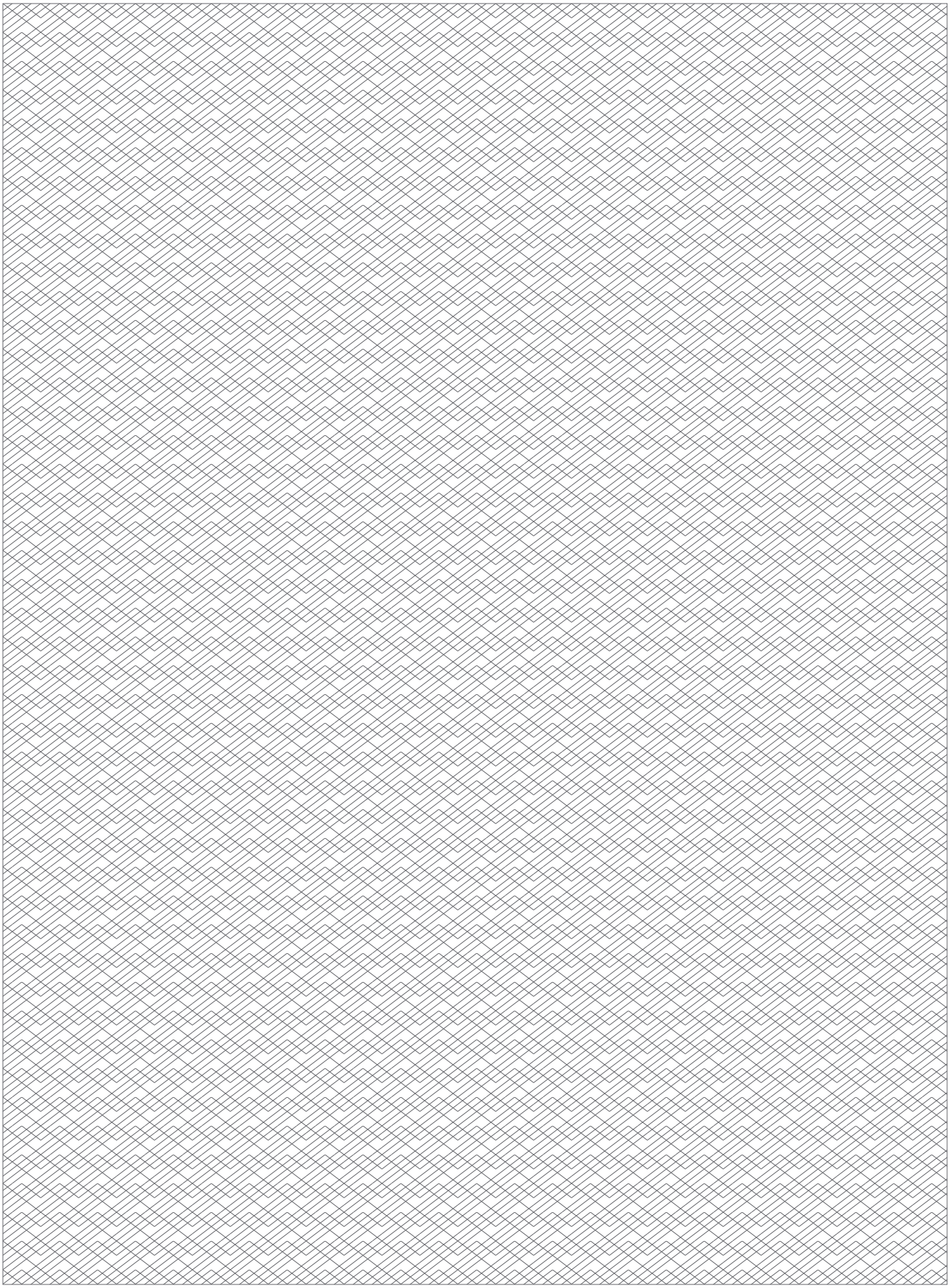
Name:

Student ID:

Instructions

1. This is a closed-book exam: written aids are not permitted.
2. Write all answers in the space provided. I have provided more space than you ought to need.
3. Calculators, phones and all other electronic aids are not permitted.
4. Unless otherwise specified, assume that all code presented compiles and runs.
5. You may detach the reference page at the back of the test.

Question	1	2	3	4	5	6	7	**Total**
Points	4	6	5	14	7	6	8	50
Awarded								



Part 1: Computer security

1. Trust

a) What is a TCB? 2 PTs

b) Why is *trusted* not the same as *trustworthy?* 2 PTs

2. Suppose that you are designing a system to manage library lending, including tracking books lent to patrons, manging inventory at each library in a network and managing requests for inter-library loans.

a) Give an example of an *integrity* policy that might be required in this system. 2 PTs

b) Give an example of a *mechanism* that might be used to help enforce this policy. 2 PTs

c) Give an example of a *vulnerability* that an adversary might use to attack this policy. 2 PTs

Part 2: Software security

Consider the following program:

```c
#include <stdbool.h>
#include <stdio.h>
#include <string.h>

// Prompt user for a password and store the result in pass
void getPasswordFromUser(char *pass);

bool checkPassword()
{
  const char ACTUAL_PASSWORD[] = "12345678";
  char userPassword[8];
  size_t len = sizeof(ACTUAL_PASSWORD);

  getPasswordFromUser(userPassword);

  // Compare first n characters of passwords to see if they match
  return (strncmp(userPassword, ACTUAL_PASSWORD, len) == 0);
}


int main()
{
  int authenticated = checkPassword();

  if (authenticated)
  {
    printf("Welcome.\n");
  }
  else
  {
    printf("Access denied.\n");
  }

  return 0;
}

```

3.a) How long is `ACTUAL_PASSWORD`? 1 PT

b) How does the `getPasswordFromUser` function introduce a vulnerability into this program? 2 PTs
Note: you do not need the source code to `getPasswordFromUser` to answer this question.

c) How would you change this function to remove this vulnerability? 2 PTs

4. Suppose that, just before calling `getPasswordFromUser`, a portion of the program's stack contains:

```
0x16fdfecd0: 00 ed df 6f 01 00 00 00 f0 ec df 6f 01 00 00 00  ...o.......o....
0x16fdfece0: 10 ee df 6f 01 00 00 00 bc 3e 00 00 01 00 00 00  ...o.....>......
0x16fdfecf0: 31 32 33 34 35 36 37 38 00 1d 41 00 01 00 00 00  12345678..A.....
0x16fdfed00: 09 00 00 00 00 00 00 00 61 00 e4 3c 98 ce 78 1b  ................
```

and that, after the call to `getPasswordFromUser`, that memory contains:

```
0x16fdfecd0: 00 ed df 6f 01 00 00 00 f0 ec df 6f 01 00 00 00  ...o.......o....
0x16fdfece0: 10 ee df 6f 01 00 00 00 61 62 63 64 65 00 00 00  ...o....abcde...
0x16fdfecf0: 31 32 33 34 35 36 37 38 00 1d 41 00 01 00 00 00  12345678..A.....
0x16fdfed00: 09 00 00 00 00 00 00 00 61 00 e4 3c 98 ce 78 1b  ................
```

a) Why don't these addresses start with `0x7fff`? 2 PTs

b) What is the address of `ACTUAL_PASSWORD`? 1 PT

c) What is the address of `userPassword`? 1 PT

d) What is the address of `len`? 1 PT

e) What is the value of `len`? 1 PT

f) Why does this value differ from your answer in the previous question about the length of `ACTUAL_PASSWORD`? 1 PT

g) What password did the user enter? 1 PT

h) If this password buffer is overflowed, what is the first thing that will be overwritten? 2 PTs

i) How can an attacker use the vulnerability in `getPasswordFromUser` to cause `checkPassword` to return `true` without knowledge of the correct password? 4 PTs

Part 3: Host security

5.**MAC and DAC**

a) Given the following details of a file: 3 PTs

```
-rwxrw-r--  1 jon  staff   7.6K 24 Jun 07:58 foo
```

Who may perform what actions on this file?

b) What is the Biba MAC policy and how is it used today in commodity operating systems? 4 PTs

6. Explain, with reference to a diagram, how CBC mode works. 6 PTs

7.**Authentication**

a) Why is a *trusted input path* important for biometrics? 2 PTs

b) What benefit does *salting* provide to a password database? 2 PTs

c) Compute the Shannon entropy of the following distribution: 4 PTs

$$\left\{\frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}\right\}$$

Workings:

Answer:

Reference material

`strcmp` and `strncmp`

```
NAME
     strcmp, strncmp — compare strings

LIBRARY
     Standard C Library (libc, -lc)

SYNOPSIS
     #include <string.h>

     int
     strcmp(const char *s1, const char *s2);

     int
     strncmp(const char *s1, const char *s2, size_t n);

DESCRIPTION
     The strcmp() and strncmp() functions lexicographically compare the null-
     terminated strings s1 and s2.

     The strncmp() function compares not more than n characters.  Because
     strncmp() is designed for comparing strings rather than binary data,
     characters that appear after a ‘\0’ character are not compared.

RETURN VALUES
     The strcmp() and strncmp() functions return an integer greater than,
     equal to, or less than 0, according as the string s1 is greater than,
     equal to, or less than the string s2.  The comparison is done using
     unsigned characters, so that ‘\200’ is greater than ‘\0’.

```