COMP3334 – Computer System Security

Assignment

Due Date: 11:59 PM, March 2nd, 2025. (No Extension. Late policy applies.)

Submission: Please submit a PDF file to Blackboard.

Total: 4 Questions

You can use as many pages as you want.

Notes:

(1) Please include your name and student ID in your submission.
(2) Please submit your answer PDF file in advance, to avoid network congestion. Network, device, or related problems cannot be acceptable reasons for late submission.
(3) Please use **electronic editors** such as Word to draft your answers instead of taking photos or screenshots of your hand-written answers.
(4) Do NOT include question text in your submissions.
(5) Please provide **calculation details** in your answers, i.e., the complete logical reasoning path about how you get the final results from the given information. Answers with only final results are not acceptable.
(6) Do NOT write programs to solve the questions in your answers.
(7) If you get a result with more than 3 decimal places during your calculation, round values in **3 decimal places**. (e.g., 0.0423 -> 0.042, 1.3458 -> 1.346)

Question 1: Basic Concepts of Computer System Security. (25/100)

Task 1 (6 Marks): Calculate the following equations:

(1) $10001101 \wedge 01110110$
(2) $10111001 \vee 01100111$
(3) $11100011 \oplus 01110111$

Task 2 (6 Marks): Given an online identity system that allows a user to create a profile from his/her personal information (e.g., identity card number, phone number). An authorized online service provider can read this profile when it provides service to the user. What are the requirements on C.I.A. associated with this online identity system? Give one example of each requirement.

Task 3 (7 Marks): This is a C program that reads a password and allows a user to perform a special action if the input password is correct.

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

void authorize() {
    char password[16];
    gets(password);     // read input from standard input, and store it in the array password
    // We omit the necessary code here.
    return;
}

void password_exam() {
    // We omit the code here for password examination.
    // The program will end if the input from authorize() does not pass the examination.
    return;
}

void special_action() {
    // We omit the code here for special action.
    return;
}

int main() {
    authorize();

    password_exam();

    special_action();
    return 0;
}
```

`gets()` is a function in C's standard library to read a character string from standard input (keyboard by default). It does not check whether the input is small enough to be stored in the specific array.

Suppose that the memory address of `authorize()`'s first code is 0x000023B1, `password_exam()`'s is 0x000028EF, and `special_action()`'s is 0x000025C2. *0x* is a prefix that indicates it's a hexadecimal number, which is not a part of the number.

A part of memory layout of the function `authorize()` is as follows.

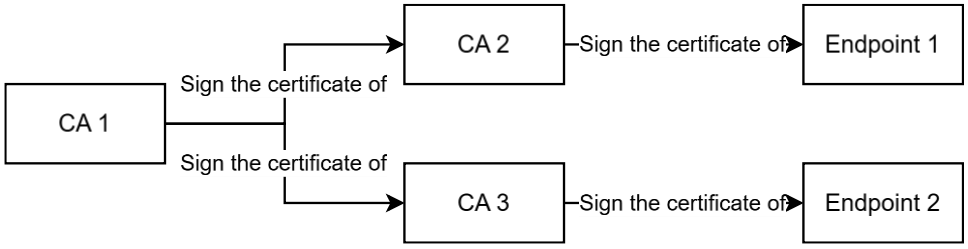
```
Memory Layout (order: top to bottom):
-           +---------+
            |         | password[16]
16 bytes    |         |
            |         |
-           |         |
-           +---------+
4 bytes     |   EBP   |
-           +---------+
4 bytes     | retaddr |
-           +---------+
```

In the layout, the array `password` occupies 16 bytes, followed by an `EBP` with 4 bytes (a variable which can be set as any values). Then the last one is `retaddr` with 4 bytes, which stores the memory address of the instruction which the computer should execute after the function `authorize()`.

If you are an attacker who does not know the correct password but wants to execute `special_action()`, provide a feasible way to execute it. Please provide details of your attack and explain why it works.

Tips: A 2-unit hexadecimal number occupies 1 byte. Each unit corresponds to 4 bits in its binary representation.

Task 4 (6 Marks): The figure below is a chain of certificates. A user trusts CA 2. Does this user trust Endpoint 2? Does this user trust Endpoint 1? Explain your reasons.



Question 2: Classic Cryptography. (25/100)

Task 1 (4 Marks): What is the difference between authentication and authorization?

Task 2 (6 Marks): Given a scenario illustrated in the picture below, Alice wants to send a message to Bob via Eve.

What are the main differences between the scenario that Eve is a passive attacker and the scenario that Eve is an active attacker?

Provide an example on both scenarios **respectively** and show which CIA properties are broken in these two examples. These two examples cannot be the same.

Task 3 (10 Marks): Given a plaintext string *POLYUCOMP*, what is the ciphertext by Caesar Cipher if the key is 7? Provide the calculation details.

Task 4 (5 Marks): In our computer, what is the main difference between how a truly random number is generated and how a pseudorandom number is generated?

Question 3: Modern Crypto Tools I. (25/100)

Given the following data, execute the last round of AES (each byte in the result matrix should be in a **2-unit hexadecimal** number).

The input of the last round in hexadecimal is $\begin{bmatrix} 0a & 1b & 5d & 2c \\ 1a & 4f & e2 & c7 \\ a9 & b3 & bc & d3 \\ ca & ad & d9 & c2 \end{bmatrix}$.

The round key of the last round in hexadecimal is $\begin{bmatrix} b6 & 8f & 48 & 23 \\ fd & 9f & 0a & f6 \\ cf & 78 & 6b & 12 \\ 6b & 96 & 23 & e9 \end{bmatrix}$.

Tips: A 2-unit hexadecimal number occupies 1 byte. Each unit corresponds to 4 bits in its binary representation. For example, $9_{(16)} = 1001_{(2)}$, $a_{(16)} = 1010_{(2)}$. Thus $9a_{(16)} = 10011010_{(2)}$. If you want to convert a number from hexadecimal to binary, the workflow is reversed (i.e., divide it into two 4-bit binary numbers and convert them respectively).

The table below is the S-box in AES in hexadecimal, which is used in *ByteSubs* of AES. The column is determined by the **least** significant nibble, and the row is determined by the **most** significant nibble. For example, given a hexadecimal value $9a_{(16)}$, its substitution result from this S-box is $b8_{(16)}$, which is at 9's row and a's column.

AES S-box

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Question 4: Modern Crypto Tools II. (25/100)

Task 1 (5 Marks): Given a cryptographic hash function that has 200-bit output, if we want to find a pair of x and x' whose digests are identical, how many trials are you expected (i.e., with about 50% chance)? Does this function meet the collision resistance requirement? Explain your reasons.

Task 2 (5 Marks): Explain why the collision exists in the hash function.

Task 3 (4 Marks): Why is MD5 deprecated? Explain your reasons.

Task 4 (6 Marks): Given a message with the length of 29 bytes, if the block size is 4 bytes, pad this message by ZeroLength method. The padded data should be in hexadecimal.

Task 5 (5 Marks): Given the message m and the secret k, build an MAC function based on a SHA-3 function $SHA3(x)$ without HMAC.