A35276

Calculators may be used in this examination provided they are not capable of being used to store alphabetical information other than hexadecimal numbers



UNIVERSITY OF BIRMINGHAM

School of Computer Science

Third Year Undergraduate

06-30231

LH Security of Real-World Systems

Main Summer Examinations 2025

Time allowed: 2 hours

[Answer all questions]

Note

Answer ALL questions.

Question 1

(a) A possible attack against a cryptographic protocol is a replay attack. Explain what a replay attack is and describe a common method of defending against them. **[8 marks]**

(b) A more advanced protocol security property is post compromise security. Explain what post compromise security is and describe how it is possible for a protocol to provide this. **[9 marks]**

(c) Give an example of a widely used protocol that is safe against replay attacks, but does not provide post compromise security. Explain how the protocol stops replay attacks but doesn't provide post compromise security. Say if you think this makes the protocol unsafe to use and explain your answer. **[9 marks]**

(d) Is it possible for a protocol to provide post compromise security but still be vulnerable to replay attacks? Explain your answer. **[7 marks]**

Question 2

Consider the following code:

```
1  void handleFiles(char fileName[]) {
2      char userChoice[8];
3      while (true) {
4          gets(userChoice);
5
6          printf("You picked: ");
7          printf(userChoice);
8
9          if (strcmp(userChoice, "1") == 0) {
10             printData(fileName);
11         }
12
13         if (strcmp(userChoice, "2") == 0) {
14             saveData(fileName);
15         }
16     }
17 }
18
19 void printData(fileName) {
20     char noOfLines[16];
21     fgets(noOfLines, sizeof(noOfLines), stdin);
22     printFileLines(noOfLines,fileName);
23  }
```

(a) The code contains two vulnerabilities. Say what they are and give the line numbers they occur on. **[4 marks]**

(b) Assuming the user entered option "1" and the program was run on a 32-bit system **without** stack canaries, sketch the stack layout just as the fgets on line 21 is about to be executed. Your sketch should include the stack frames of the handleFiles and printData functions, including any registers' values, argument or values that may have been stored on the stack. **[10 marks]**

(c) For a 32-bit system, with no ASLR or stack canaries what is the easiest way that an attacker could exploit this program to make it run a function of the attackers choosing. **[4 marks]**

(d) Now assume that the program is run on a 64-bit machine **with** stack canaries, sketch the stack layout just as the fgets on line 21 is about to be executed. Your sketch should include the stack frames of the handleFiles and printData functions, including any registers' values, argument or values that may have been stored on the stack. **[10 marks]**

(e) Is it still possible for an attacker to make the program jump to a function of their choice on a 64-bit machine with stack canaries? Explain your answer. **[6 marks]**

Question 3

The following C (pseudo)code runs on a microcontroller and overall fulfils the following functionality: it takes an externally supplied 16-byte input block `in[]` and a 4-byte PIN `pin[]`. It then checks if the PIN matches the internally stored one, and if that succeeds, decrypts the input with a secret key, modifies it, encrypts it, and sends it back. If the PIN is wrong, the code instead sends an encrypted error counter (which is only updated every 16th PIN check failure).

```
1  const char pin_stored[4] = { ... };
2  const char key[16] = { ... };
3  static char error_cnt = 0;
4  static char aes_err = 0
5  ...
6  void check_input(char in[16], char pin[4]) {
7    char out[16] = { 0 };
8
9    for(int i = 0; i < 4; i++) {
10     if(pin[i] != pin_stored[i]) {
11       error_cnt++;
12       if(error_count > 16) {
13         aes_err++
14         out[0] = aes_err;
15         error_cnt = 0;
16       }
17
18       aes_enc(out, key); // encrypt out using key in-place
19       send(out, 16); // send to user
20       return;
21     }
22   }
23
24   // Pin check succeeded
25   aes_dec(in, key); // decrypt input
26   doSomeChange(in, out); // assume that this writes the result to out
27   aes_enc(out, key); // encrypt out using key in-place
28   send(out, 16); // send to user
29   return;
30 }
```

(a) First, assume the attacker controls `in[]` and `pin[]`, and can inject a single fault that causes one instruction to be skipped. Briefly describe how the attacker can then bypass the PIN check to reach line 24? **[6 marks]**

(b) Now, assume instead that the attacker can inject one bit flip before SubBytes in the final round of the AES encryption `aes_enc()`. The attacker knows which bit is flipped, and can also target any byte in the AES state. How can this be used to recover `key` without knowing the correct PIN, using the differential attack from the

lecture? You do not need to give all formulas, instead give the steps of the attack in bullet points. **[10 marks]**

(c) Now, assume instead that the attacker cannot inject faults, but instead precisely measure the time between `in[]` and `pin[]` being input, and the response from the above code being sent using `send()`. How can this side channel be used to recover the PIN? **[7 marks]**

(d) How could the fault attack from (a) be mitigated, assuming the attacker can only inject a single fault? You do not need to give code; briefly describe your countermeasure in 2–3 sentences. **[5 marks]**

(e) How could the side-channel attack from (c) be mitigated? You do not need to give code; briefly describe your countermeasure in 2–3 sentences. **[5 marks]**

This page intentionally left blank.

Do not complete the attendance slip, fill in the front of the answer book or turn over the question paper until you are told to do so

Important Reminders

- Coats/outwear should be placed in the designated area.
- Unauthorised materials (e.g. notes or Tippex) must be placed in the designated area.
- Check that you do not have any unauthorised materials with you (e.g. in your pockets, pencil case).
- Mobile phones and smart watches **must** be switched off and placed in the designated area or under your desk. They must not be left on your person or in your pockets.
- You are not permitted to use a mobile phone as a clock. If you have difficulty seeing a clock, please alert an Invigilator.
- You are not permitted to have writing on your hand, arm or other body part.
- Check that you do not have writing on your hand, arm or other body part – if you do, you must inform an Invigilator immediately
- Alert an Invigilator immediately if you find any unauthorised item upon you during the examination.

Any students found with non-permitted items upon their person during the examination, or who fail to comply with Examination rules may be subject to Student Conduct procedures.