SENG2250/6250 System and Network Security

School of Electrical Engineering and Computing

Semester 2, 2020

Lab 2 Solutions

Objectives

1) Review the knowledge of Topic 2 cryptographic techniques (symmetric-key cryptography).
2) Apply cryptographic techniques for problem-solving.
3) Implement cryptographic operations in programming.

Part 1 Review Questions

1. Explain in which cases would be suitable to use symmetric-key encryption, which cases would better to use asymmetric-key encryption. Give an example for each.

   Answer:

   Symmetric-key encryption can be used if the secret (key) is pre-shared between users. Pubic key based encryption can be used if there is no key shared prior to communication.

2. What is the perfect secrecy?

   Answer:

   Perfect secrecy gives no advantage to an adversary when a ciphertext is presented.

3. Describe the encryption and decryption processes of the CBC operation mode.

   Answer:

   Refer to the lecture slides L2-S36-37 and the short video.

4. What is a cryptographic hash function?

   Answer:

   A cryptographic hash function provides the following properties: (refer to the slide L2-S55)

   - Pre-image resistant (one-way)
   - Second pre-image resistant:

- Collision resistant

5. How would stream cipher relate to block cipher?

   Answer:

   For example, a block cipher can be used with operation modes to generate a key stream for a stream cipher.

Part 2 Exercises

6. **Cryptanalysis**: Apply cryptanalysis to reveal the (meaningful) plaintext of the following ciphertext:

   KRHPH FL BX BAAWH ZX KRH KPHH

   (The ciphertext was generated by using the monoalphabetic substitution cipher.)

   Answer:

   THERE IS AN APPLE ON THE TREE

7. **Triple-DES**
   a. Find out the meet-in-the-middle attacks. (e.g., https://en.wikipedia.org/wiki/Meet-in-the-middle_attack.)
   b. Why does double-DES have (approx.) 2^{57} security level.

      Answer:

      Go through the plaintext side, there are 2^{56} operations. From the final ciphertext side, there are 2^{56} operations. Therefore, we need around $2^{56} + 2^{56}$ operations to break double-DES encryption.

   c. Why does Triple-DES (3 independent keys) have (approx.) 2^{112} security level.

      Answer:

      Go through the plaintext side, there are $2^{56} \times 2^{56}$ operations; From the final ciphertext side, there are 2^{56} operations. Therefore, we need around $2^{56} \times 2^{56} + 2^{56}$ operations to break triple-DES encryption. In this case, the key length is 168-bit, but the EFFECTIVE key length is 112-bit.

d. Why is the middle portion of the triple-DES is decryption rather than encryption?

Answer:

As in the case where a single key is used across all three DES blocks the product is equivalent to that of single DES (M ==> $E_k(M)$ ==> $D_k(E_k(M))$ = M ==> $E_k(M)$), therefore providing backwards compatibility even in updating the cipher.

8. Hash Functions

a. Is the following function H a hash function? Why?

$$H(x) = x \ mod \ 65537, x \in \mathbb{N}$$

b. Is the above function H a cryptographic hash function? Why?

9. **Programming**

Fast modular exponentiation ($b^e \ mod \ n$) is an essential operation for many modern security algorithms. In Lab 1, you have implemented the modular exponentiation function. But it is slow if the base and exponent are very large. We introduce a fast modular exponentiation as below. Write a (C/C++/Java/Python) program to implement the fast modular exponentiation operation based on the following pseudocode.

```
function powmod2(base b, exponent e, modulus n) {
    if n  = 1
        return 0
    rs = 1
    while (e > 0) {
        if (e & 1) == 1
            rs = (rs * b) mod n
        e = e >> 1
        b = (b*b) mod n
    }
    return rs
}
```

Use the above implementation to find the solutions

$$3^3 \ mod \ 7 = 6$$
$$10^8 \ mod \ 133 = 93$$
$$3785^{8395} \ mod \ 65537 = 355$$
$$17^{45} \times 17^{61} \ mod \ 1023 = 1006$$
$$17^{45+61} \ mod \ 1023 = 1006$$

Try to understand the algorithm.

Part 3 Discovery (external readings)

10. Self-study: Vigenère cipher (e.g., https://en.wikipedia.org/wiki/Vigenère_cipher)

a. Encrypt the plaintext:

using the tabula recta of the Vigenère cipher.

b. Is the Vigenère cipher secure against cryptanalysis?

Answer:

No, Vigenère cipher is insecure against cryptanalysis.

11. Self-study: Birthday Paradox (e.g., https://en.wikipedia.org/wiki/Birthday_problem)

a. What is the birthday paradox?

Answer:

It is a probability problem that considers the probability of two people (from a randomly selected set) have the same date of birth.

b. How does it relate to the security of hash functions?

Answer:

It can be used to analyse the probability of having a collision of a hash function. It is important to evaluate the security level of a hash function.