Assignment 1- Tutorial
Cryptography & Security

CS-457 Introduction to Secure Systems

Prof. Evangelos Markatos

Giannis Vlachogiannakis
gvlach@csd.uoc.gr

Computer Science Department
University of Crete

Part 1 - One time pad

One-Time Pad

- It’s a theoretically unbreakable cipher.
- It uses a pre shared-key that is at least the size of the plaintext.
- The algorithm XORs each byte of the plaintext with the corresponding key byte.

One-Time Pad

Plaintext: HelloWorld

Key: randombyte

Output: (H⊕r)(e⊕a)(l⊕n)(l⊕d)(o⊕o)(W⊕m)(o⊕b)(r⊕y)(l⊕t)(d⊕e) =

Hex: 3A 04 02 08 00 3A 0D 0B 18 01

One-Time Pad

- In order to generate a random key, use /dev/urandom (use a Linux system)

- Decryption can be done by XORing the ciphertext with the key.

- You have to store the key used for encryption, in order to successfully decrypt the ciphertext

One-Time Pad - The Assignment

- Implement the One-Time Pad encryption and decryption methods.
- You are given 2 hexes that are ciphertexts produced from applying OTP to 2 different phrases but using the same key
  - “72814c04ba04a4f8a05 …”
  - “6f904804a949f6febc5c …”
- You need to find the key and the 2 initial plaintexts.

One-Time Pad - The Assignment

- You need to find the key and the 2 initial plaintexts.
- Make use of words.txt file also.
  - https://raw.githubusercontent.com/dwyl/english-words/master/words.txt

XOR properties will be on your side

Part 2 - Caesar Cipher

Caesar Cipher

- The Caesar cipher is one of the oldest and simplest substitution ciphers.
- Each letter in the original message is replaced by a letter corresponding to a certain number of letters up or down in the alphabet.
- Because the shift is fixed, there are very few possible keys, making it highly susceptible to brute-force and statistical attacks.

Caesar Cipher

Plaintext: Cryptography is awesome

Ciphertext: Hwduytlwfumd nx fbjxtrj

Key: 5

Caesar Cipher - The Assignment

- You need to implement encryption and decryption methods.
- The next step is to write code that breaks the algorithm using:
  - brute force
  - English letter frequency

Part 3 - Stream Cipher

Stream Cipher

- A stream cipher encrypts data continuously as a "stream" of bits or bytes, rather than grouping data into large blocks.
- It generates a continuous sequence of pseudorandom bits (the keystream) which is then mathematically combined - usually via XOR - with the plaintext bits.

Stream Cipher

Plaintext: 1 0 1 1 0 0 1 (letter ‘W’)

Keystream: 0 1 1 0 1 0 1

Ciphertext: 1 1 0 1 1 0 0 (result of Plaintext XOR Keystream)

- Decryption of ciphertext is done by XORing it with the keystream.

Stream Cipher - The Assignment

- Implement a stream cipher algorithm.
- Encryption and Decryption will be done by XORing with the generated keystream.
- Generate keystream using a seeded pseudorandom generator (PRNG).
- Final step is to demonstrate with examples what are the issues of reusing the same keystream.

Part 4 - Timing Attack

Timing Attack

- A timing attack is a type of side-channel attack.
- Instead of breaking the mathematics an attacker observes the physical implementation (how much time a request is processed).
- Server logic is given for this assignment.

Timing Attack

```
def insecure_check(guess):
    for i in range(len(SECRET)):
        if i >= len(guess):
            return False
        if guess[i] != SECRET[i]:
            return False
        time.sleep(0.03) # intentional leak
    return len(guess) == len(SECRET)
```

Timing Attack

- Imagine secret password is “CAT”.
- Attacker guesses “AAA”
  - Server rejects
  - Response Time: ~0.00 seconds

Timing Attack

- Attacker guesses “BAA”
  - Server rejects
  - Response Time: ~0.00 seconds

Timing Attack

- Attacker guesses “CAA”
  - ‘C’, Bingo - server sleeps for 0.03 seconds
  - ‘A’, Bingo - server sleeps for 0.03 seconds
  - Response Time: ~0.06 seconds

Timing Attack

- The attacker knows that “CA” are the correct first two letters.
- Only by measuring the time it takes the server to say “Wrong Password”.
- The server does not explicitly confirm the correct guessed letters.

Timing Attack - The Assignment

- Implement the vulnerable server logic.
- Implement the attacker program that measures the response time of the server and recovers the secret password one character at a time.
- Demonstrate experimentally how the attack succeeds even though the server does not directly reveal any information about the password.

Part 5 - Secret Sharing

Secret Sharing

- Three friends have a common bank account.
- They deposit some money and agree to withdraw it after 10 years.
- To make sure that no one of them will get the money alone, they decide to split the account password (D) in three pieces.

Secret Sharing

- One of them suggested that if something happens to any of the three, the other two will not be able to withdraw the money.
- Thus they agreed that any two of them will be able to withdraw the money.

Secret Sharing

- To achieve this, the friends decide to use the geometric equation of a straight line: **y=ax+D**
  - **D (The Secret):** This is the y-intercept of the line (the exact point where the line crosses the y-axis, meaning x=0)
  - **a (The Slope):** A randomly chosen secret number that determines the angle of the line.

Secret Sharing

- Each friend is given a set of coordinates (a point) on the specific line: (x1, y1), (x2, y2), (x3, y3).
- **Why 1 share is useless:** If a friend only has a single point, an infinite number of straight lines can pass through that single point.
- **Why 2 shares work:** If two friends combine their coordinates, they can use simple algebra to calculate a and then solve for the y-intercept (D) to get the password.

Secret Sharing - The Assignment

- The goal of this part is to implement a program to create the three points.

- And a program to reconstruct the secret out of any two points.

Notes

Notes

- The submission is done through **e-learn** platform, subscribe to the lesson and you will find the submission there.
- You can use the course's mailing list (hy457@list.csd.uoc.gr) or the Office Hours for questions.
- This assignment should be implemented using C on Linux-based machines.
- A good practice will be to have a **Makefile** for compilation of your assignment and also a **README** file describing what you have achieved and what you don't.
- The use of AI generated code is **strictly** prohibited.


THAKS FOR LISTENING
ANY QUESTIONS?

Thank you for listening!

CS-457 Introduction to Secure Systems

Giannis Vlachogiannakis
gvlach@csd.uoc.gr

Computer Science Department
University of Crete