ACIT 4630 – Lab 5 – Cryptography

Notes:

This lab should be done with your partner (if you have one).

Instructions:

Please do the following tasks on the SEED VM.

Symmetric Cryptography

You are given a ciphertext.txt (right click and select save Link As to download the text file) and asked to decrypt it using `openssl enc` command.

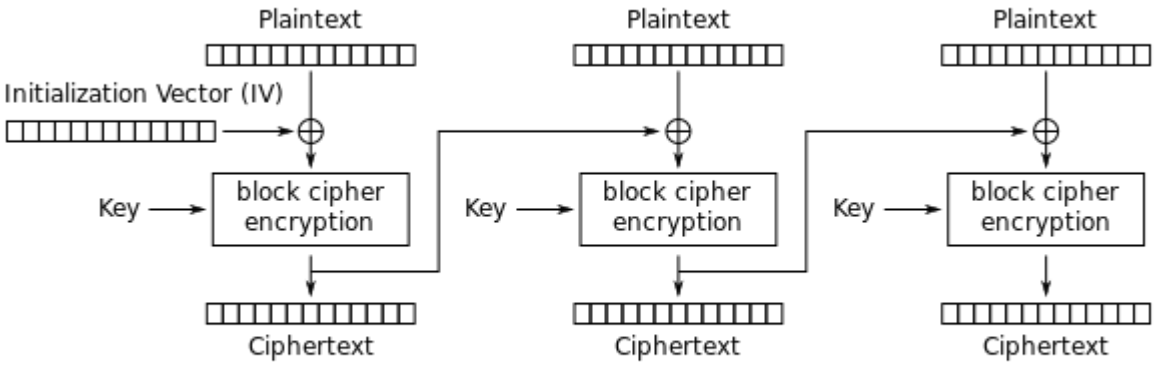
Type `man enc` on the terminal to see how you should use this command.

Here's some of the info you need:

- key: 00112233445566778889aabbccddeeff
- initialization vector (IV): 0102030405060708090a0b0c0d0e0f
- algorithm used: -aes-128-cbc

Answer the following questions:

- **Q1.** Note the name of the algorithm. What does each part refer to (aes, 128, cbc)?
  - Watch this video to understand what CBC mode is



Cipher Block Chaining (CBC) mode encryption

- **Q2.** What is the plaintext of the message?
- **Q3.** Is the provided key part of a key pair?
- **Q4.** Should this key be shared publicly?

Asymmetric Cryptography

RSA (Rivest-Shamir-Adleman) is one of the first public-key cryptosystems and is widely used for secure communication. The RSA algorithm first generates two large random prime numbers and then uses them to generate public and private key pairs, which can be used to do encryption, decryption, digital signature generation, and digital signature verification.

You need to download this code on the VM and **uncomment** and **replace** different parts of it in each of the following tasks. After each change, to compile the code, run the following command in the folder where the code is:

```
gcc -o rsa rsa.c -lcrypto
```

and then run `./rsa` to see the results.

Note: The provided C code uses hex strings as input and output.

- To convert an ASCII string to a hex string to use in the C code, use the following Python command:

```
python3 -c 'print("the_ascii_string".encode("ascii").hex())'
```

- To convert a hex string back to ASCII string, use the following Python command

```
python3 -c
'print(bytearray.fromhex("the_HEX_string").decode())'
```

Note: Imagine that e and d in the C code are **Alice**'s public and private keys respectively.

For each task below take a screenshot of the change you made in the C code, explain your changes, and include a screenshot of the results.

Task 1 - Encryption

Bob wants to send the message "A top secret!" confidentially to Alice (the quotations are not included). Encrypt this message using the correct key.

- **Q5.** What's the ciphertext?
- **Q6.** Verify that it can be decrypted back to the original message.

Task 2 - Decryption

Alice has received the following encrypted message. Decrypt the ciphertext C with the correct key and convert the result back to a plain ASCII string.

```
C =
8C0F971DF2F3672B28811407E2DABBE1DA0FEBBBDFC7DCB67396567EA1E2493F
```

- **Q7.** What's the message in plain text?

Task 3 - Signing a message

Alice has generated a signature for the following message, using SHA256 hash.

```
M = I owe you $2000.
```

- This command gives you the hash value (in hex) of the above message using SHA256: `echo "I owe you $2000" |sha256sum`
- What's the signature? **Note:** The hash is **NOT** the signature.

Please make a slight change to message M, such as changing $2000 to $2500, and sign the modified message.

Q8. Compare both signatures and describe what you observe.

Task 4 - Verifying a Signature

Bob receives a message M = "Launch a missile." from Alice with a signature S (already in hex). We know that Alice used SHA256 to make the signature. Assuming the public key provided in the C code belongs to Alice, verify if this message is indeed from Alice or not.

```
S =
D96BE0AE035D94A7C88BA0FE518589717415CCBF880A1172BA48E2D014C5F0C8
```

Q9. Suppose that the signature is corrupted, such that the last byte of the signature changes from C8 to C9, i.e., there is only one bit of change. Please repeat this task and describe what will happen to the verification process.

Submission For Lab 5:

- Create a report answering any questions in the lab above and including any required screenshots (i.e., for Tasks 1 to 4 under asymmetric cryptography)
- Walk through your report with your instructor.
- Submit your report to the Learning Hub in PDF format.

This lab is worth a maximum of 10 marks.