## Question 1

### b)

openssl enc -aes-128-cbc -d -in encrypted -out decrypted1 -iv 143124152 -K 00112233445566778899887766554

### c)

Welcome to CS 6324: Information Security

## *Question 2*

### a)

openssl dgst -sha1 encrypted

SHA1(encrypted)= dab3eddc62bd38370293515b29bff769ec520d8a

### b)

openssl dgst -sha256 encrypted

SHA256(encrypted)= 0e276bd77b7e5462fe4598079777d08dc22b1cb87207d2dc94621742d447338b

### c)

SHA-1 is a cryptographic hash function used by NIST. SHA-1 creates a hash for the certificate or the file without revealing their content. Breaking the SHA-1 means createing a forged certificate so browsers will accept it. Malicious files can also be forged in place or normal files.

SHA-1 is also widely used in TLS certificate signature, GIT versioning system, document signature and backup system. SHA-1 certificates are considered insecure by 2017. SHA-0 has already been broken using computing power of a smartphone.

Shattered attack has been accomplished using GPU to compute SHA-1. Attacking using the CPU is easier to implement, but using the GPU is far more efficient. Using

The implication of a SHA-1 attack it reduced the attack time of brute force to a reasonable amount of time and financial cost.

## *Question 3*

### a)

### 

### b)

* + 1. The output is recognizable. Same pixels output the same encrypted result.

### c)

* + 1. openssl enc -aes-128-cfb -in temoc.bmp -out temoc-cfb.bmp -K 0011223344556677 -iv 0
    2. openssl enc -aes-128-cbc -in temoc.bmp -out temoc-cbc.bmp -K 0011223344556677 -iv 0

### d)

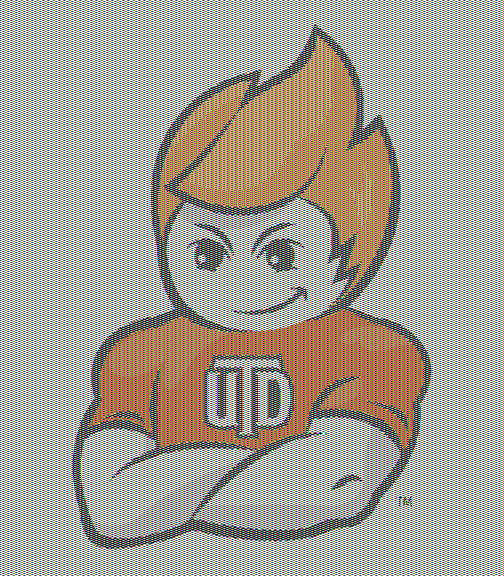
CBC: Cipher Block Chaining. This fixes the issue of ECB by applying XOR on the plaintext with the previous ciphertext block before encrypting it. This makes the ciphertext block depend on all plaintext blocks up to the encryption point. The key remains unchanged. The strength is this ciphertext will result in a pseudo random output unlike ECB. The weakness is it cannot be done in parallel. An IV is added to the first block to make the encryption unique.

PCBC: Propagating Cipher Block Chaining, a type of CBC. This goes one step further by applying XOR of the plaintext block to both the previous ciphertext and plaintext block. PCBC causes small changes in ciphertext to propagate indefinitely.

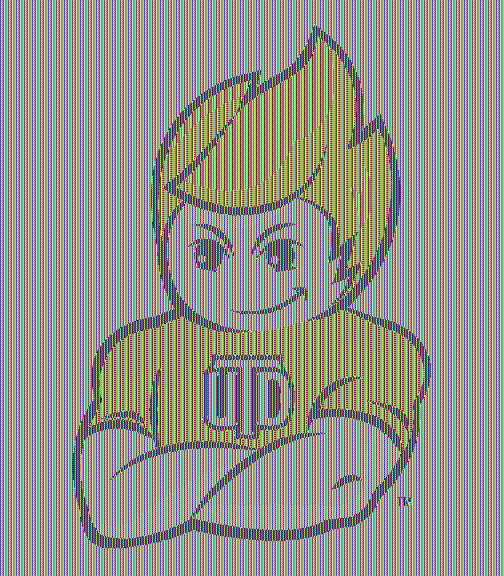
CFB: Cipher Feedback. CFB makes a block cipher into a self-synchronizing stream cipher.

### e)

openssl enc -des-ofb -in temoc.bmp -out temoc-cfb.bmp -K 0 -iv 0



openssl enc -des-ofb -in temoc.bmp -out temoc-ofb1.bmp -K 0 -iv 2698549198794132

An IV is only a nonce used to avoid applying the same encryption to different pieces of data. It does not make data in the same piece random. Without the key, same data in the same piece outputs the same ciphertext so the attacker can recognize the pattern.

### f)

Authenticated Encryption means the ciphertext is is sent with a Message Authentication Code (MAC). If the MAC does not match the ciphertext then the message is not authenticated. AE maintains both confidentiality and integrity. The modes are: Galois/Counter Mode (GCM), EAX mode.

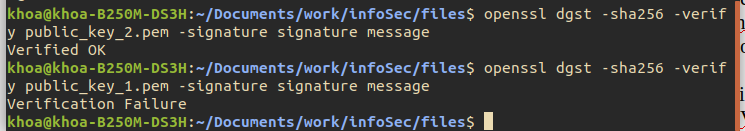
I like GCM most because it can be implemented with reasonable hardware resource. It is also supported in many software. (OpenVPN, TLS) and it can take advantage of parallel computing.

## *Question 4*

### a)

We sign the hash message for efficiency reason. Hashed message is smaller.

### b)

 It was signed by Bob.

## *Question 5*

### a)

p=47

q=71

p\*q = 3337

Totient = 3220

d = (79^-1) mod 3220 = 1019

### b)

Answer = (2423^79) mod 3220 = 547

## *Question 6*

### b)

### 

### c)

### d)

The program performed the RSA algorithm to find all the numbers. Then it encrypted using the public key e and decryptedusing the private key d.

### e)

RSA-OAEP is a padding scheme for the RSA algorithm. It uses a G and H hash function. The program pads the input to a certain size before applying the hash function and then reduces the size to a certain length. RSA-OAEP forces the attacker to obtain the entire message to be able to decode instead of having to work on a small part of the message. (chosen ciphertext attack).

## *Question 7*

### a)

First: TLS libraries.

Who was responsible: NSA. The defender is NIST.

What happened:

Second: OpenSSL tool

Who was responsible: a Debian package manager. The defender: none.

What happened: the removal of a few lines of code due to mistakes of the debugging tool reduces the PRNG of OpenSSL to only 15 bits, so an attacker can deduce the state of the PRNG very easily.