SIES (Nerul) College of Arts, Science and Commerce

NAAC Re-Accredited 'A' Grade

Sri Chandrasekarendra Saraswathy Vidyapuram, Plot

1-C, Sector V, Nerul, Navi Mumbai-400 706.

PROJECT REPORT ON

Cryptography

(hashing,digital signature & checksum)

Submitted By:

Darren Dsouza

(ROLL NO:02)

[MSc Computer Science Part-1 (SEM-1)]

2022-2023

Subject: Software Defined Networking

Project Guide : Prof. Flosia Simon

SIES (Nerul) College of Arts, Science and Commerce

NAAC Re-Accredited 'A' Grade

Sri Chandrasekarendra Saraswathy

Vidyapuram, Plot 1-C, Sector V,

Nerul, Navi Mumbai-400 706.

CERTIFICATE

This is to certify that the Project entitled Cryptography is successfully completed by **Darren Dsouza** of **Part-1(Sem-1) Masters in Science (Computer Science)** as per the requirement of University of Mumbai in part fulfillment for the completion of PG Degree of Masters of Science (Computer Science). It is also to certify that this is the original work of the candidate done during the academic year 2022-2023.

Roll No: **02** Date of Submission:

Subject: **Software Defined Networks**

Prof.Flosia Simon
(Project Guide)

Index

Introduction:

- This short Project is made on the topic of cryptography and its primitives.We look at a few techniques used such as hashing,digital signature and checksums.

- These techniques are implemented to show the CIA(confidentiality,Integrity and Authenticity) of Message Transmission.

- The implementation showcases basic cryptographic techniques used for message security.

- This Documentation contains information regarding the project implementations such as screenshots of codes and output and basic definition of the concepts used.

Technical Discussion:

Cryptography:

- Cryptography is the technique of securing information and communications through use of codes so that only those persons for whom the information is intended can understand it and process it.
- The Project consists of sub-parts mainly Hashing,digital signature and checksum implementations

Hashing:

- Hashing is the process of transforming any given key or a string of characters into another value**.**
- For Hashing we have used MD5 and SHA1 algorithm which takes as input a string and outputs a 128-bit hash encoded digest and 160-bits hash encoded digest respectively

Digital Signature:

- A digital signature is an electronic, encrypted, stamp of authentication on digital information such as email messages, macros, or electronic documents.
- In Digital Signature,We Have used symmetric(fernet encryption) and asymmetric(RSA encryption) for encrypting and authenticating messages.
- A Short example has also been shared which shows exchange of signed message between two hosts in cisco packet tracer.

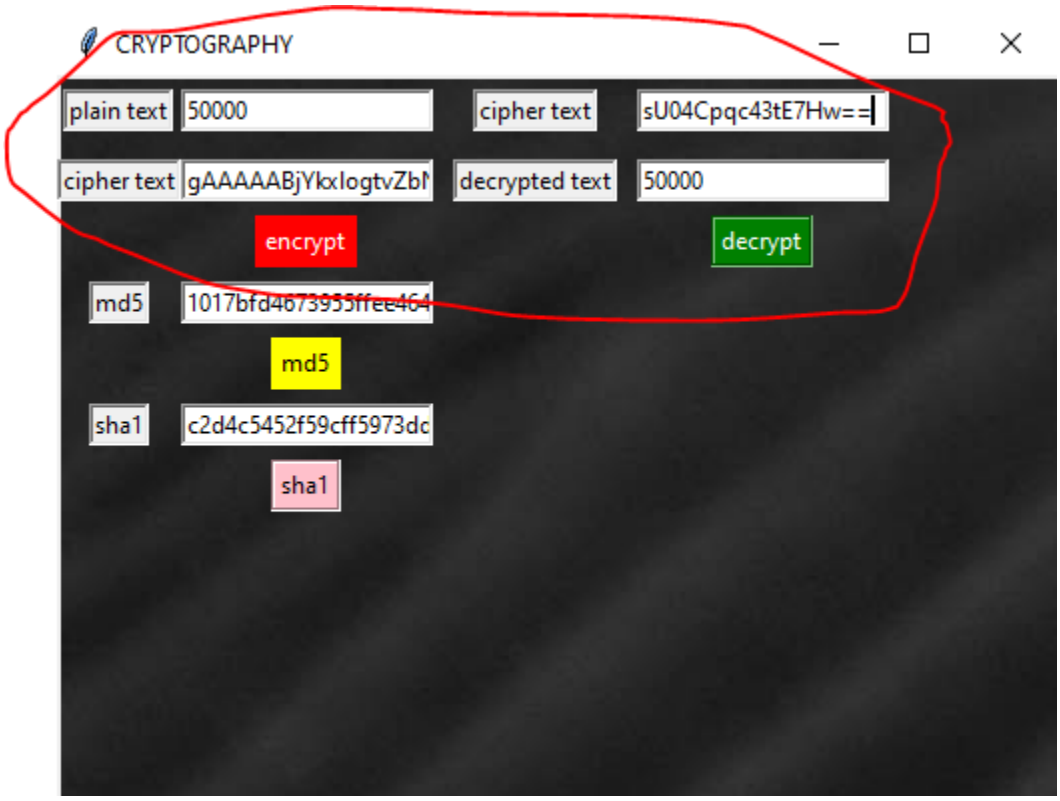
Checksum:

- A checksum is a value that represents the number of bits in a transmission message and is used by IT professionals to detect high-level errors within data transmissions.

- Checksum is used for error validation.The program below takes as input the sum of the message and computes on both sender and receiver side whether the message received is tampered or not.
- Checksum implementation is in python.

Programs:

Symmetric Key encryption:

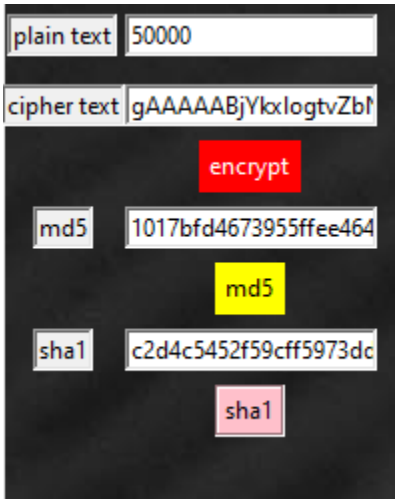


-In this project demonstration,I have first implemented symmetric encryption using fernet encryption which uses a single key for encryption of data.below is an image showing the use of a single key.

```
    pt = e1.get()
    key=Fernet.generate_key()
    global f
    f=Fernet(key)
    ct = f.encrypt(bytes(pt,encoding='ascii'))
    e2.insert(0, ct)

def decryptMessage():
    ct1 = e3.get()
    pt1 = f.decrypt(bytes(ct1,encoding='ascii'))
    e4.insert(0, pt1)
```

MD5 & SHA1:



One of the prominently used encryption methods:MD5 & SHA1

The MD5 message-digest algorithm is a cryptographically broken but still widely used hash function producing a 128-bit hash value.The above image shows the output in hexadecimal notation.

SHA1 takes an input and produces a 160-bit (20-byte) hash value known as a message digest

Below is the code for the function:

```
def encryptMd5():
    message=e1.get()
    encode_msg=hashlib.md5(bytes(message,encoding='ascii'))
    encode_msg=encode_msg.hexdigest()
    e5.insert(0,encode_msg)

def SHA1():
    message=e1.get()
    encode_msg=hashlib.sha1(bytes(message,encoding='ascii'))
    encode_msg=encode_msg.hexdigest()
    e6.insert(0,encode_msg)
```

ASymmetric Key encryption:

```
from Crypto.PublicKey import RSA

keypair=RSA.generate(bits=1024)
print(f"Publickey: (n={hex(keypair.n)},e={hex(keypair.e)}")
print(f"Pvtkey: (n={hex(keypair.n)},d={hex(keypair.d)}")
```

```
Publickey: (n=0xdf3f71e82b34fd7ea199ee39f5e82791574c7d1e24321
Pvtkey: (n=0xdf3f71e82b34fd7ea199ee39f5e82791574c7d1e2432117
```

The Rsa algorithm is used to demonstrate asymmetric key encryption. We use crypto.publickey module to have a pair of public and private keys.

below are the images to show how the messages are encrypted and decrypted(encrypted using private key and decrypted using public key):

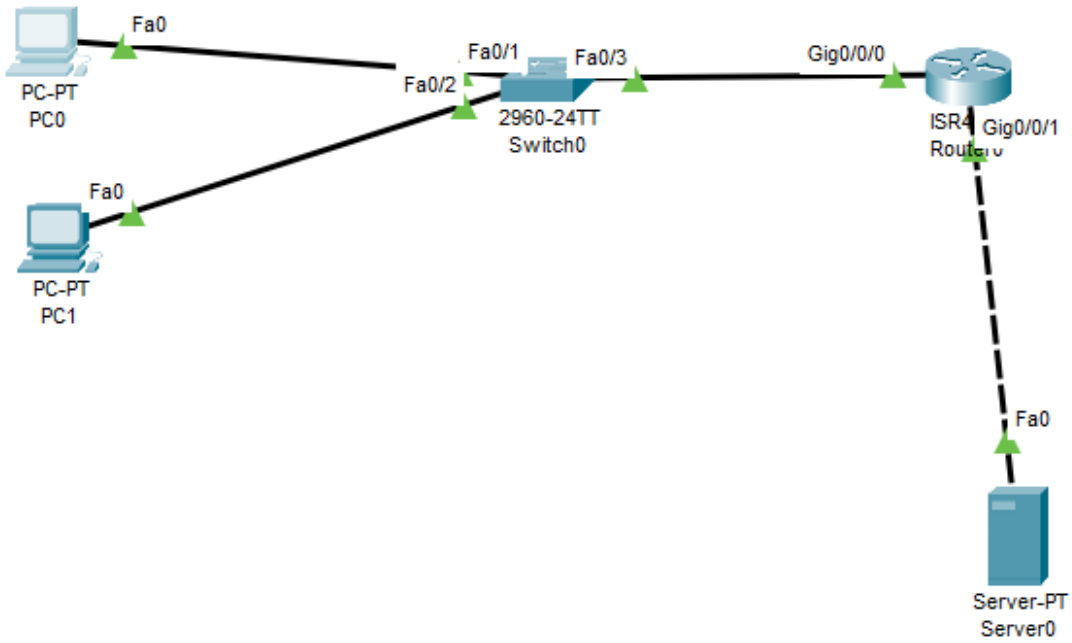
```
msg=b'4655758ebacb8cebecbfd44adfffbe39d4ed8e4b'
hash = int.from_bytes(msg,byteorder='big')
signature=pow(hash,keypair.d,keypair.n) #pvt key
print("Signature:",signature)
print("hash:",hash)
```

```
Signature: 12967738752524227337904443500202856571943028 3
hash: 435639139902041648145277274855620606150737377981295
```

```
#enter signature below to verify
signature=12967738752524227337904443500202856571943028034219810424
hashfromsignature=pow(signature,keypair.e,keypair.n) #public key
print("Signature valid:",hash==hashfromsignature)
print("hashfromsignature:",hashfromsignature)
```

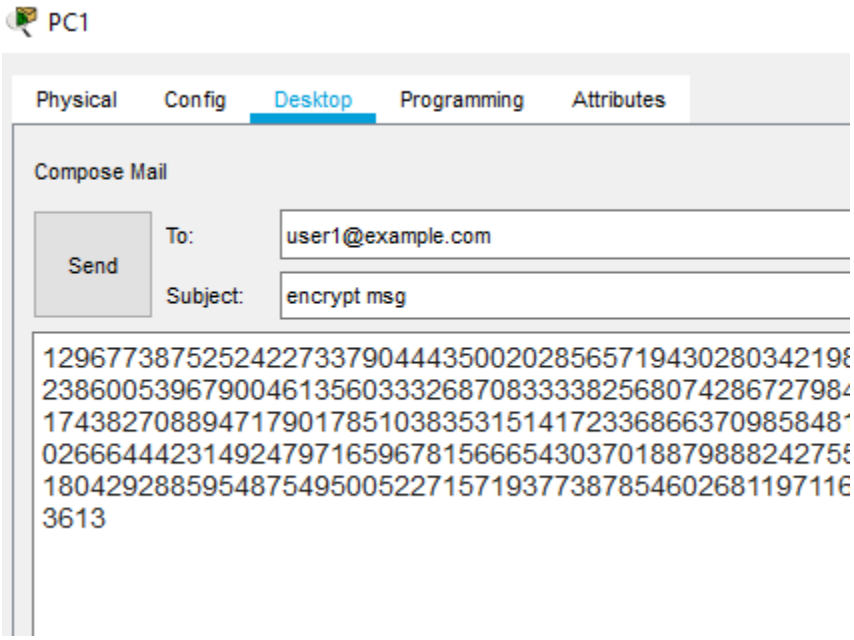
```
Signature valid: True
hashfromsignature: 435639139902041648145277274855620606150737377981
```

SMTP for Message Passing:

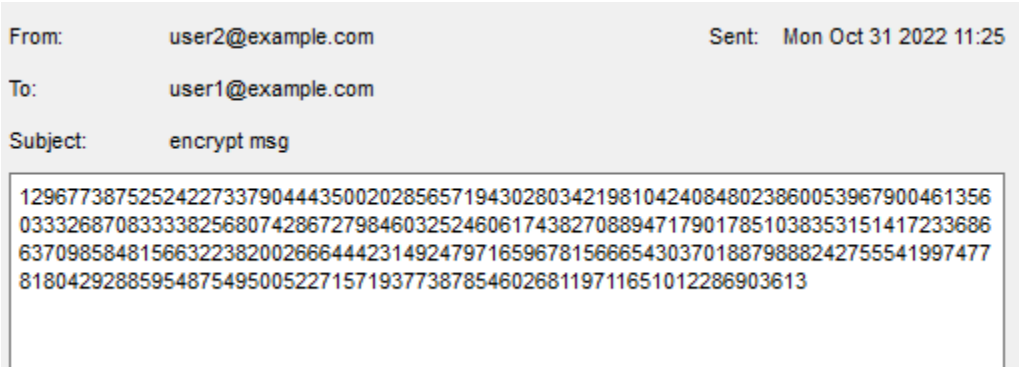


The above network is created in Cisco Packet Tracer where we use an smtp server to exchange the digital signature(keys) between two hosts and show the encryption and decryption using the keys.

sending message from user2:



Message received to user1:



The signature received is decrypted using the public key and the message is correct:

```
#enter signature below to verify
signature=1296773875252422733790444350020285657194302803421981042408
hashfromsignature=pow(signature,keypair.e,keypair.n) #public key
print("Signature valid:",hash==hashfromsignature)
print("hashfromsignature:",hashfromsignature)
```

Checksum:

```
SENDER SIDE CHECKSUM:  0111001011011011
RECEIVER SIDE CHECKSUM:  0001101001001100
Receiver Checksum is equal to 0. Therefore,
STATUS: ACCEPTED
>>>
```

The checksum is implemented in python program which verifies the correctness of message being exchanged between sender and receiver.the message received is correct only if the checksum at receiver is 0.

below image shows an example of error in transmission

```
SENDER SIDE CHECKSUM:  0111001011011011
RECEIVER SIDE CHECKSUM:  0001011001001100
Receiver Checksum is not equal to 0. Therefore,
STATUS: ERROR DETECTED
>>>
```

Result:

- Confidentiality:Hashing
- Authentication:Digital Signature
- Integrity:Checksum

These are the cia triads that are above programs that help in ensuring message protection.

References:

- https://www.geeksforgeeks.org/fernet-symmetric-encryption-using-cryptography-module-in-python/
- https://www.geeksforgeeks.org/md5-hash-python/
- https://www.geeksforgeeks.org/sha-in-python/
- https://cryptobook.nakov.com/digital-signatures/rsa-sign-verify-examples
- https://www.geeksforgeeks.org/implementing-checksum-using-python/?ref=rp
- https://www.youtube.com/watch?v=D0N1EMQe9SA
- https://github.com/DarrenDsouza47/SIES-sem1