Cryptography

Symmetric Encryption:

Symmetric Encryption used the same key for encryption and decryption. It useful for large data but if the key is intercepted then the security is broken.

Algorithms: AES, DES, 3DES, Blowfish

Asymmetric Encryption:

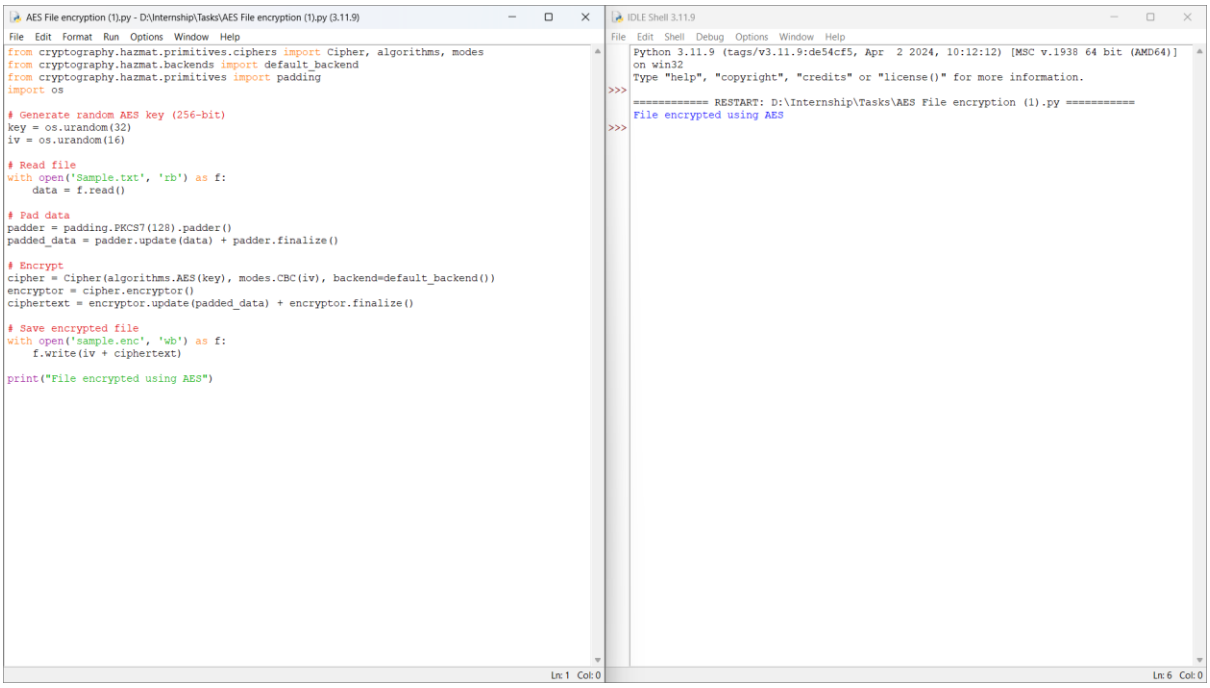
Asymmetric Encryption uses a pair of keys: one public key for encryption and one private key for decryption. Provides authentication and digital signatures. It slower than Symmetric Encryption.

Algorithms: RSA, ECC, DSA

AES Encryption:

AES stands for Advanced Encryption Standard. It is a symmetric encryption algorithm used to protect data by converts plaintext into ciphertext using secret key.

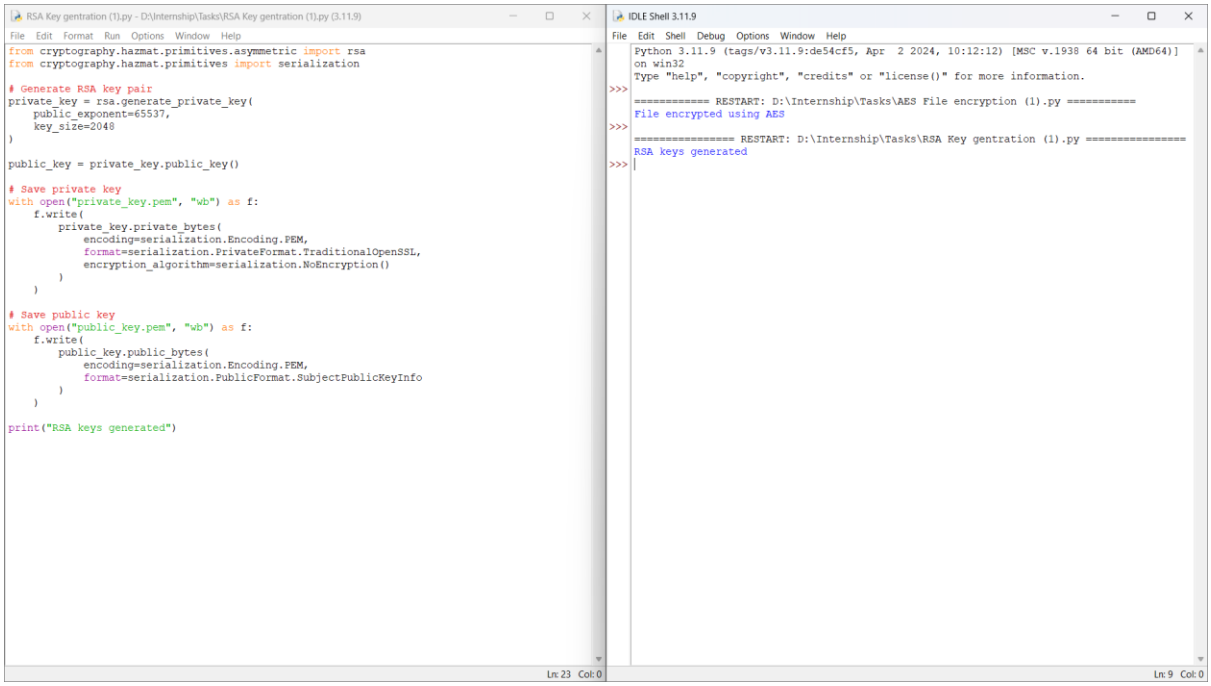
In this lab, I used python programming for AES Encryption. I encrypted a text file using python.



RSA Key Generation:

It is an asymmetric encryption algorithm. It uses two keys for encryption and decryption.

In this lab also I used python for generate RSA key. This program generates public key and private key and stored those keys in files.

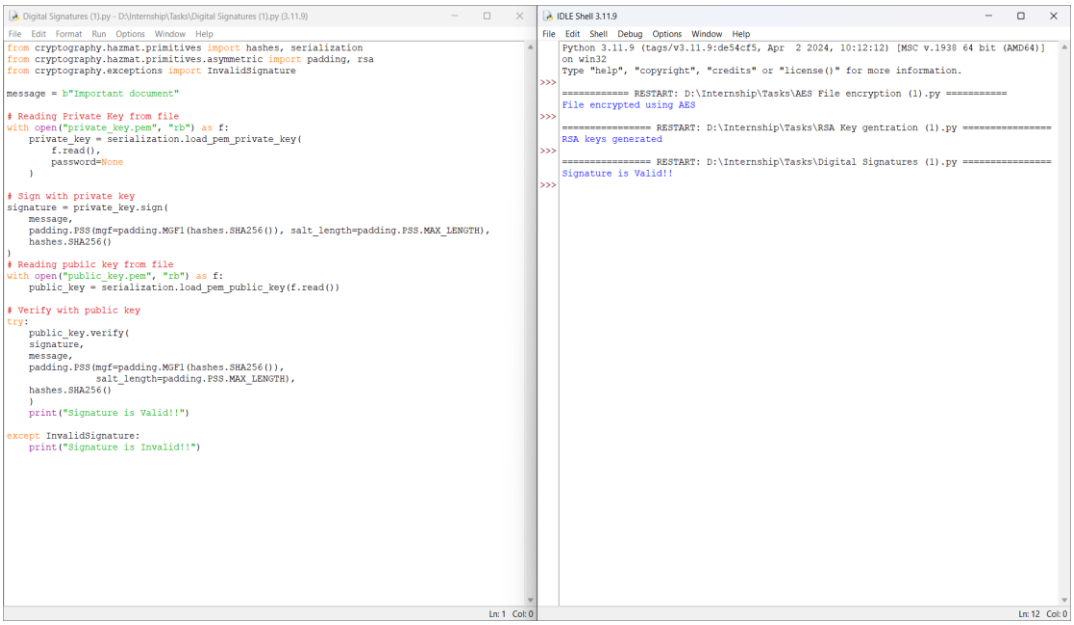


```
from cryptography.hazmat.primitives.asymmetric import rsa
from cryptography.hazmat.primitives import serialization

# Generate RSA key pair
private_key = rsa.generate_private_key(
    public_exponent=65537,
    key_size=2048
)

public_key = private_key.public_key()

# Save private key
with open("private_key.pem", "wb") as f:
    f.write(
        private_key.private_bytes(
            encoding=serialization.Encoding.PEM,
            format=serialization.PrivateFormat.TraditionalOpenSSL,
            encryption_algorithm=serialization.NoEncryption()
        )
    )

# Save public key
with open("public_key.pem", "wb") as f:
    f.write(
        public_key.public_bytes(
            encoding=serialization.Encoding.PEM,
            format=serialization.PublicFormat.SubjectPublicKeyInfo
        )
    )

print("RSA keys generated")
```

```
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024, 10:12:12) [MSC v.1938 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: D:\Internship\Tasks\AES File encryption (1).py ==========
File encrypted using AES
>>>
========== RESTART: D:\Internship\Tasks\RSA Key gentration (1).py ==========
RSA keys generated
>>>
```

Digital Signature Verification:

The RSA keys are the signatures of the file. When the signature is mismatched, those files can't decrypt. I used python for this lab also.



```
from cryptography.hazmat.primitives import hashes, serialization
from cryptography.hazmat.primitives.asymmetric import padding, rsa
from cryptography.exceptions import InvalidSignature

message = b"Important document"

# Reading Private Key from file
with open("private_key.pem", "rb") as f:
    private_key = serialization.load_pem_private_key(
        f.read(),
        password=None
    )

# Sign with private key
signature = private_key.sign(
    message,
    padding.PSS(mgf=padding.MGF1(hashes.SHA256()), salt_length=padding.PSS.MAX_LENGTH),
    hashes.SHA256()
)
# Reading public key from file
with open("public_key.pem", "rb") as f:
    public_key = serialization.load_pem_public_key(f.read())

# Verify with public key
try:
    public_key.verify(
    signature,
    message,
    padding.PSS(mgf=padding.MGF1(hashes.SHA256()),
                salt_length=padding.PSS.MAX_LENGTH),
    hashes.SHA256()
    )
    print("Signature is Valid!!")

except InvalidSignature:
    print("Signature is Invalid!!")
```

```
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024, 10:12:12) [MSC v.1938 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: D:\Internship\Tasks\AES File encryption (1).py ==========
File encrypted using AES
>>>
========== RESTART: D:\Internship\Tasks\RSA Key gentration (1).py ==========
RSA keys generated
>>>
========== RESTART: D:\Internship\Tasks\Digital Signatures (1).py ==========
Signature is Valid!!
>>>
```

Hash Generation:

Hash is the fingerprint of the file. We need to generate hash for the file first when the file was created. The receiver also needs to generate hash for the same file before he decrypts it. If it changed then the file must intercept by someone.

Like before labs in this lab also I used python for generate hash.