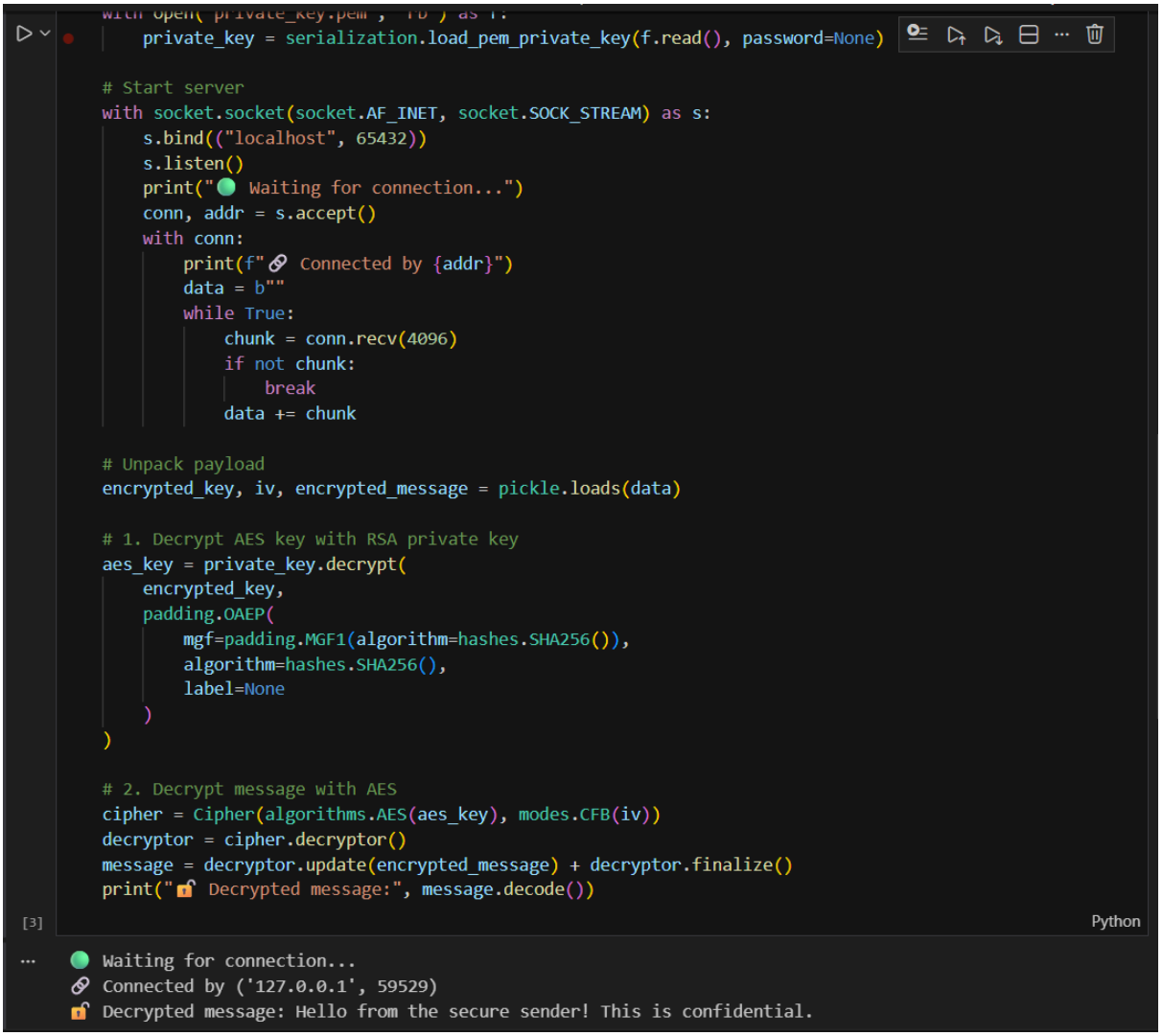
Lab 2

This lab demonstrates how **secure communication works in real networks** using a combination of:

- **RSA (asymmetric encryption)** for secure key exchange
- **AES (symmetric encryption)** for fast data encryption
- **Python sockets** for sending encrypted data between two computers

The lab simulates how secure systems such as **HTTPS and VPNs** protect data in real life.

Generation of public and private keys and saving of the keys



Starting the server, unpack the payload and decrypt AES key with RSA private key and descrypt message with AES

Lab 2

private_key.pem

```
-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQDAj/LHgo+4GPDo
QsvwEXULznZ1tQv74ZIwWjHDA7Zvp+Lns9AlHcztBj5lSE9wkQthwKiOLi1xh7rw
xl8obNs+D4sj/u8usQNXD6RFYAQJtlYenxVoczqYMD2lxXZ5kM0rSA3ArnwAIk7W
i2/Iss57jq4idI0LgiZqoOl5x4KapC78nxCgxVjOkF0Iv3tnBdjsBCE6UrPsYZ6a
uXF5O1TtZotWfmYay4jIPsE6YGouiJeji7xogmsQna1sMWZ7voHFpL7MwAnsqUSQ
U1g0sgVfcJQVTbpFp2F1l9yr7HY96EXunOA7JYrUhDMPjVaR1GpJOMXku4Ln+gv+
SJ9A8IjpAgMBAAECggEAXBJ4Gz0qSvRh/Cw0Jw2x0Xte2NlAaadppwT5Vi2oRasY
q4EmvRiKq6qYmwUANuqjbpIYBiO+RtKHaCjxxvqFuh7kgMmcw86PskkgWquWQq6U
bv/R1KeEFNC8KU9JXscF0vQY31COqeG9oIf7YxNb92g5iu//oMkSZVmbSUdEA3w3
LTYJ4S3BXf3m2+ZYyyp6W4JL3N8uYvlk9F9YgFNL9avTD79G3YEmzn916UxSpz+5
JbyxTOnOtKyYNt85adh458lpVYa+zq/bQEz9wfKmRcUhJW0FVTSL3TzFlp0nqoJD
wZGpcpq8KO/cJITZL8kJp1WEUoiGNktwmBm5mJOFoQKBgQD/k55uXbP6qxHtaMZZ
ersBxkpiycthCMKMIJL4P3Fxas8Cjxk96YHKzdM7VyaIV5qt8598Mwo2oqkKi9J+
YH8Tdn/163hGaM5Jqhz6aWUvXd32hyVXlkmoOfenChHVk+Pg7GSP6PitZxDEIhno
1ioQDON3boV46hN74Jgeea+z2wKBgQDA4Zt4T0rcPtGYQv6Ey2Q4BYhGQaS1GhCY
q+qiArpMh4+OCyZ95pX6PUsivHOf7nAqFXBHMFisb5ADYQ/CD3fgScP7q59OOMWo
e+A+K6G7VmCPvolJPkjdwJ7QXjAgolrPPxZvQ9P9nj02KaLD5mPjBeX5oQuNK9Uf
N40yZcPziwKBgEAhjabKEoh69sOo/+SiHFzkofPKCSOvLAo3fkmWueOo1osg10em
2W/HWBLS2pNp0CGIOWPT9uMgy+Qek9Isapa6rH9L9+FLGUB41E6ttbs+BZ1/6gnh
Z1g0O4WJzFHYi1z8VHucDRKOAzIEEJZRbzqZ0VY9wHZH3l2+TM2Q4VrzAoGBAJrZ
xZ6OUjSGPnE2f+4O+HO9hU2aT+iRd3eKtgHBnN3qC4dnTkzPveK8KMFHeLCSlfxD
X/PXTamKm50K5MLQ9lXcAvrfKzN6gr2kTkC7eMnVLArbVo618XwOs7HnnE3m6m7u
YE2Nr/kyfEQqoyERq2nNB3glIggBg8XfLH2kk+/HAoGAazHsmRvQdbsToxs63Te3
neeoWu4flLZajXpwhH74XbwovBa8H95oc2nuVFmVZK4VfbSJHmrNT/2BbZ4psN6k
xBve2g0PGPsQBIoG+vr1VpMVjn0w45y50w5kWPljExKQvNVOv35uALX5e5PhXYq1
ZlNlV0ytKJOyXSD3JAKq2CA=
-----END PRIVATE KEY-----
```

Begin and end of the generation of the public key

public_key.pem

```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAwI/yx4KPuBjw6ELL8BF1
C852dbUL++GSMFoxwwO2b6fi57PQJR3M7QY+ZUhPcJELYcCoji4tcYe68MZfKGzb
Pg+LI/7vLrEDVw+kRWAECbZWHp8VaHM6mDA9pcV2eZDNK0gNwK58ACJO1otvyLLO
e46uInSNC4ImaqDpeceCmqQu/J8QoMVYzpBdCL97zwXY7AQhOlKz7GGemrlxeTtU
7WaLVn5mGsuIyD7BOmBqLoiXo4u8aIJrEJ2tbDFme76BxaS+zMAJ7KlEkFNYNLIF
X3CUFU26RadhdZfcq+x2PehF7pzgOyWK1IQzD41WkdRqSTjF5LuC5/oL/kifQPCI
6QIDAQAB
-----END PUBLIC KEY-----
```

Lab 2

sender.py

```python
# sender.py
import socket
import os
from cryptography.hazmat.primitives import serialization, hashes
from cryptography.hazmat.primitives.asymmetric import padding
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
import pickle
# Load recipient's public key
with open("public_key.pem", "rb") as f:
    public_key = serialization.load_pem_public_key(f.read())
# Message to send
    message = b"Hello from the secure sender! This is confidential."
# 1. Generate random AES key and IV
    aes_key = os.urandom(32) # AES-256
    iv = os.urandom(16)
# 2. Encrypt message with AES (CFB mode)
    cipher = Cipher(algorithms.AES(aes_key), modes.CFB(iv))
    encryptor = cipher.encryptor()
    encrypted_message = encryptor.update(message) + encryptor.finalize()
# 3. Encrypt AES key with RSA (recipient's public key)
    encrypted_key = public_key.encrypt(
    aes_key,
padding.OAEP(
    mgf=padding.MGF1(algorithm=hashes.SHA256()),
    algorithm=hashes.SHA256(),
    label=None
)
)
# 4. Package: (encrypted_key, iv, encrypted_message)
payload = pickle.dumps((encrypted_key, iv, encrypted_message))
# 5. Send via socket
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect(("localhost", 65432))
    s.sendall(payload)
print("✅ Encrypted message sent!")
```

Full code