Lab Assignment 5

AP21110010302
Krish Srivastava
CSE – E
Network Security – CSE 315L

1. **Write a code to simulate the working procedure of digital signature by using from py_ecc.bls.ciphersuites library in multi client environment.**

   SERVER.py

```python
import socket
from py_ecc.bls import G2ProofOfPossession as bls

def verify_signature(public_key, message, signature):
    return bls.Verify(public_key, message, signature)

def handle_client_connection(client_socket):
    public_key = client_socket.recv(1024).decode()

    while True:
        message = client_socket.recv(1024).decode()
        if not message:
            break
        signature = client_socket.recv(1024).decode()

        if verify_signature(public_key, message, signature):
            print("Signature is valid.")
        else:
            print("Invalid signature.")

    client_socket.close()

def main():
    host = '127.0.0.1'
    port = 12345

    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind((host, port))
    server_socket.listen(5)

    print(f"Server listening on {host}:{port}")
```

```
    while True:
        client_socket, _ = server_socket.accept()
        print("Accepted connection from", client_socket.getpeername())
        handle_client_connection(client_socket)

if __name__ == "__main__":
    main()
```

CLIENT.py

```
import socket
from py_ecc.bls import G2ProofOfPossession as bls
from random import randint

def generate_keypair():
    private_key = randint(1, bls.curve_order)
    public_key = bls.SkToPk(private_key)
    return private_key, public_key

def sign_message(private_key, message):
    signature = bls.Sign(private_key, message)
    return signature

def main():
    host = '127.0.0.1'
    port = 12345

    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect((host, port))

    private_key, public_key = generate_keypair()
    print("Public key:", public_key)

    client_socket.send(public_key.encode())

    while True:
        message = input("Enter message (type 'exit' to quit): ")
        if message == 'exit':
            break

        signature = sign_message(private_key, message)
        print("Signature:", signature)

        client_socket.send(message.encode())
        client_socket.send(signature.encode())

    client_socket.close()
```

```
if __name__ == "__main__":
    main()
```

2. **Setup and configure a certificate authority using Easy-RSA, distribute Certificate Authority's public certificate in a LAN (/ NAT) network, create certificate signing request, and revoke certificates.**

```
Enter New CA Key Passphrase:
Re-Enter New CA Key Passphrase:
Using configuration from /home/kali/easy-rsa/pki/3ebf0368/t
emp.6c2e8daa
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be
 incorporated
into your certificate request.
What you are about to enter is what is called a Distinguish
ed Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [Easy-RSA
 CA]:attnx

* Notice:

CA creation complete and you may now import and sign cert r
equests.
Your new CA certificate file for publishing is at:
/home/kali/easy-rsa/pki/ca.crt
```

```
┌──(kali㉿kali)-[~/easy-rsa/pki]
└─$ cat ~/easy-rsa/pki/ca.crt
-----BEGIN CERTIFICATE-----
MIIB6jCCAXCgAwIBAgIUYZZvuzFwNcY7/cJeAf0tsXGqK6UwCgYIKoZIzj0EAwIw
EDEOMAwGA1UEAwwFYXR0bngwHhcNMjMwNTE4MjA0MDAzWhcNMzMwNTE1MjA0MDAz
WjAQMQ4wDAYDVQQDDAVhdHRueDB2MBAGByqGSM49AgEGBSuBBAAiA2IABE+qJ/fQ
qbt7d+iu23cmcvQtYqfyfh8zZP9OeG0xXCW3+O8a9p7hz2qNM2qIYnb+nXkl0zmj
zv2frjwP+QzcmwNRScSe27kA7vw+fKiRPxtr6OuUeddoRalCNsoNQV21uKOBijCB
hzAMBgNVHRMEBTADAQH/MB0GA1UdDgQWBBSjJlLYyvv5ErIaVe7bV9uB6SCdFjBL
BgNVHSMERDBCgBSjJlLYyvv5ErIaVe7bV9uB6SCdFqEUpBIwEDEOMAwGA1UEAwwF
YXR0bniCFGGWb7sxcDXGO/3CXgH9LbFxqiulMAsGA1UdDwQEAwIBBjAKBggqhkjO
PQQDAgNoADBlAjAd00UpAX4+HuLTAkxpWa65qqmrLgkKKz/ga9aBBA2UpHDIjRbI
snPDUymKSK6tRjsCMQDUCDtx+H+8HcRlRZKMChk4D5tP2h4O18TLwu71n4LXSKMV
nf33zqq23EQ5/Y+1L2g=
-----END CERTIFICATE-----
```

```
┌──(kali㉿kali)-[~/easy-rsa]
└─$ openssl genrsa -out achyut-server.key

┌──(kali㉿kali)-[~/easy-rsa]
└─$ openssl req -new -key achyut-server.key -out achyut-server.req
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IN
State or Province Name (full name) [Some-State]:New Delhi
Locality Name (eg, city) []:Delhi
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Achyut
Organizational Unit Name (eg, section) []:asdasd
Common Name (e.g. server FQDN or YOUR name) []:sdad
Email Address []:aasdasda@asdas

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:attinx
An optional company name []:attinx
```