

Network Security A3

Soumya Mohapatra (2021103), Harshit Pal (2021255)

Introduction:

The assignment consisted of building a certification system with a central certification authority and 2 clients 1 of which needs to send messages to others while the other sends replies. Certificates sent by the CA can have custom validity (this can be changed if wanted to a constant). Both clients ask for their own and each other's certificate. The certificate has the following fields: the client ID (port), public key of the requested client, time of issuing and duration of validity along with Certificate Authority id it is concatenated with its hash and encrypted by CAs private key for transfer.

Assumptions:

1. Each of the correspondents knows each other's public key. (in reality C1 and C2 may generate their own keys and enroll it to the CA)
2. They all are on the same machine i.e for the socket address the port number is different. This is for consistency since we use port number as their IDs not the entire socket
3. C1 is the establisher of connection. (For sake of convenience)

Key Components:

RSA Algorithm:

1. Provided in rsa.py which when run can generate key pairs for all three nodes. The length of the key can be set arbitrarily. We have set it to 8 digits for sake of readability. Generally it should be set to 100 digits with n being 300 digits
2. The module provides three utilities rsa_encrypt, rsa_decrypt for encryption and decryption (they are both the same, just different names for sake of readability). Takes in input bytes and returns encrypted bytes according to the key. We use block encryption by converting the bytes into integers and dividing it into segments according to N.
3. Rsa_generate_key_pair generate two key pairs and N for two given random_prime

Certificate Validity Checker:

1. Checks the validity of the certificate by decrypting the entire message and then decrypting the sha256 hash encrypted using CAs private key. Comparing the digest tells

us about the integrity of the message. It is non-repudiable since we use asymmetric encryption over hashing

Encryption Between Clients:

1. After the clients have received their certificate they extract the public keys of the certificate. Anyone who sends a message first encrypts it with its private key and then with the other client's public key. Decryption happens in the reverse process. If the certificate expires the key information is deleted from the client side.

Sample output:

```
mint@mint-HP-Laptop-15s-fr2xxx:~/dev/drive-download-20240331T103211Z-00
01$ python3 CA.py
Certification Authority is now listening for requests...

Client at ('127.0.0.1', 37186) has connected

Enter validity duration for certificate (in mins): 11

Replying to Client: b'D\x04\x08f\x0be\x06*\x0cp\x1d\x81\x17\xe0\xf5\xc
4\x0da\x0a\x06\x01\x03\xe3\x0a\x0b\x12\xe1\x08\x05\x0a\x0b\x0b-6\x04\x05
9\x0c\x03\x03\x0b0r\x0a\x0c\x0806E\x08\x0a\x08f\x0ca\x06\x0b1\x02\x08BTN\
x04\x03\x09a\x0af\x0a2h\t(\x7fm(\x0b\x08j0\x99 \xc3\x0d\xfd\x0cf\x15\x1eeh0
\x1e7\x0ff-\xc2\x0f0x8a\x02\x0d1n\x087\x0c\x0b620\x0f4\x094\x0cd\x092\x0c\x1f8
\x0f3\x07fd\x0e9\x0b\x0c5b\x14VZ\\\x04\x0d\x0fa\x0bc\x0cd'

Enter validity duration for certificate (in mins): 11

Replying to Client: b'\x06\x05\xffPHI\x0c\x2L\x09\x0f\x07\x092L\x0b\x0e\x0f
3(\x0e\x03\x0c\x14:\x0e0\x92\x0b\x03\x0f0\x0d3\x0ac\x0c\x0e0\x0b26f\x0a5\x0c
ar\x01\x911\x087\x06:\x0f\x0f1f\x0f3-\x0f2h\x06\x1c\x0cfh\x1a-\x0f1\x089\x0e
\x0ee\x0cF\x00\x0f0\x0d\x0f5\x0b0\x01r\x0e4\x0cc7\x03\x0ae\x03f\x0ce\x083\x190\x
92y(2(\x0e,\x01\x02\x0e75B00\x05\x0e4*\x05\x0a\x01\x0d3|\x0a*\x04\x0da;\x0e
1N\x1c\x0isp)\x12\x0f9\x0a\x026)\x0f5\x0d\x03\x0ba\x07\x10'

Client at ('127.0.0.1', 58426) has connected

Enter validity duration for certificate (in mins): 13

Replying to Client: b'\x04\xe2577Lz\x0f9\x0a\x01\x0e-\x09\x0b4\x08c'6 \x
0b4\x08H(\x0f1\x0e0a\x18L\x04\x0e1e006\x0f\x0a81\x17\x0ec0\xff\x132\x08c\x
e2V\x01\x0a0M)\x01\x0f\x15193":b\x0f3\x08a\x07f\x1c@x0d1\x0b\x0e4Yr\x0af\x0c
5\x0f3\x07f1\x0aep"'\x0b\x094"0\x0f6[F]\x11E\x0d4F\x0f\x0c0\x0a\x0e9\x0f4;\x1e\x0c
5\x011\x0d857\x0cb\x0dK\x085UT\xff.A\x0841g\x0f2\x0a\x0e*\x0d\x012\x07\x16\x
08-\x0a\x06-

Enter validity duration for certificate (in mins): 13

Replying to Client: b'\x11\x08f\x0cd5\x0bY\x02\x07\x17\x13\x0b5W\x0a4\x0e4\
xd5j\x07\x11\x080\x0b4\x07(6k\x02\x0fb(\x02\x01\x0a\x08\x0abN\x0b46;0a\x0e9
\x08bp\x03\x0d9\x0e90\x0d9M\x05\x0e7w\x0ae\x0cd\x02\x0ek\x06\x0a3\x130\x0d7\x0c9\
\x0a08\x08\x0e7\x0b7\x0c3\x021\x0d9\x0cFHA\x0cb\x08:\x07f\x0eb\x0ff)"'\x0b-\x0f
b1\x0d3\x0dc:\x0dbN\x0b0\x0b0\x0b0\x07f\x0e3\x12\x0c3\x0f0\x0f0\x00\x0f1\x0e0\x1c
7\x1a5\x11\x0cc\x0c0\x0ab00_7:\x1c\x09f\x02\x0b6"\x0f8c'

connection timed out
connection timed out
[]

mint@mint-HP-Laptop-15s-fr2xxx:~/dev/drive-download-20240331T103211Z-00
01$ python3 CA.py
Traceback (most recent call last):
  File "/home/mint/dev/drive-download-20240331T103211Z-001/CA.py", line
22, in <module>
    my_sock.bind(my_addr)
OSError: [Errno 98] Address already in use

mint@mint-HP-Laptop-15s-fr2xxx:~/dev/drive-download-20240331T103211Z-00
01$ python3 C1.py
Requesting my Certificate from CA...

Received Certificate from CA: 5551|7548004284869,7734692935499|2024-03
-31 20:43:22.303620|11|5550|02f00ab6e9e39abd711708e5b61ae10d7b9d1e4fe3
b0fe7023102b06df97ff3

My ID: 5551
My Public Key: 7548004284869,7734692935499
Time of Issue: 2024-03-31 20:43:22.303620
Duration of Certificate: 11 mins
ID of CA: 5550

Enter Y to Ask for your certificate: y
Requesting other Client's Certificate from CA...

Received Certificate of other Client (Soumya) from CA: 5552|6777405829
9,7734692935499|2024-03-31 20:43:27.775763|11|5550|f968a9560ea5a9ea79
3a6cdc3b569f42ab59cc05f09c133303f910b406001

Client ID: 5551
Client Public Key: 7548004284869,7734692935499
Time of Issue: 2024-03-31 20:43:22.303620
Duration of Certificate: 11 mins
ID of CA: 5550

11
(67774058299, 7734692935499)
5552
Enter Message to send to client or (0) to Exit: afaf
Reply from client: ef
Enter Message to send to client or (0) to Exit: hello
Reply from client: hello2
Enter Message to send to client or (0) to Exit: hello3
Reply from client: hello4
Enter Message to send to client or (0) to Exit: q
[]

mint@mint-HP-Laptop-15s-fr2xxx:~/dev/drive-download-20240331T103211Z-0
01$ python3 C2.py
Requesting my Certificate from CA...

Received Certificate from CA: 5552|67774058299,7734692935499|2024-03-
31 20:43:34.589135|13|5550|dc6fc2e0a44bf81ee04956181b803e9ee7aea68ba8
025500937692627b0222d01

My ID: 5552
My Public Key: 67774058299,7734692935499
Time of Issue: 2024-03-31 20:43:34.589135
Duration of Certificate: 13 mins
ID of CA: 5550

Enter Y to Ask for other client's certificate: y
Requesting other Client's Certificate from CA...

Received Certificate of other Client (Harshit) from CA: 5551|75480042
84869,7734692935499|2024-03-31 20:43:39.988974|13|5550|bc53cce30a92c8e
0142d3b09162795bda7ca466cf3284c8c84bc712e2a7aa4f8

Client ID: 5551
Client Public Key: 7548004284869,7734692935499
Time of Issue: 2024-03-31 20:43:39.988974
Duration of Certificate: 13 mins
ID of CA: 5550

Received a message from client (('127.0.0.1', 46132)): afaf
Enter your reply: ef
Received a message from client (('127.0.0.1', 35176)): hello
Enter your reply: hello2
Received a message from client (('127.0.0.1', 42682)): hello3
Enter your reply: hello4
```

Conclusion:

The program successfully sends messages from one end to the other. A lot of decisions that are made are for readability hence the code can be made more secure by following suggestions above.