Team : - Ajinkya Bokade (CS17BTECH11001), [Rushikesh Tammewar](mailto:cs17btech11041@iith.ac.in)(CS17BTECH11041)

For the following questions, Ajinkya (CS17BTECH11001) acts as (Alice and Charlie) and Rushikesh (CS17BTECH11041) acts as Bob.

**Part A: Secure file transfer between Alice (student A) and Bob (student B)**

**Step 1**: Create RSA (2048) key pairs for Alice and Bob and exchange public keys over email. Password protect your respective private keys

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl genpkey -aes256 -algorithm RSA -pkeyopt rsa\_keygen\_bits:2048 -out cs01-private-key.pem |



This command takes a passphrase as input and generates a password protected private key. It uses the AES256 algorithm to password protect private key. To generate a private key, RSA algorithm is used (key size is 2048 bits).

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl pkey -in cs01-private-key.pem -text |



This command allows users to view private keys in text format.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl pkey -in cs01-private-key.pem -out cs01-public-key.pem -pubout |



This command allows you to generate a public key from a private key as input and also asks a passphrase to the user which was used to generate the private key.

Alice Public key

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl pkey -in cs01-public-key.pem -text -pubin |

This command is used to view public key in text format.

-----BEGIN PUBLIC KEY-----

MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAm8qL3XzIh00ew6gx07+W

9ojil3H23s6Qei8oHU2wzN9+qpuP5lfB13ty4kXg75QyQ5yOLskm3BFmojB82oMd

yF8cnQ1DMKbu12ZpTzAB4t9TFcN3ta6AnukuVF2OFAMC3QVId8afao4Gdu6hQoQF

+V1AFEs/av+v08vKta4KZ+NJS71aMTtbfkwJDAeVK8xX1AKLXE+QYZUZjQ79JUcR

U+AgN4h4mqI4gF3Blaf8Z23PcrUvsQBZGKCayAzihxeYpFkXgKYp0SEohWF98Ete

RIg7OeTzk1TEmzdv2TqbESvQ2K4KwHx3G0njBAgQnonBmRdhFnW8VHjSUVZAWib0

1wIDAQAB

-----END PUBLIC KEY-----

RSA Public-Key: (2048 bit)

Modulus:

00:9b:ca:8b:dd:7c:c8:87:4d:1e:c3:a8:31:d3:bf:

96:f6:88:e2:97:71:f6:de:ce:90:7a:2f:28:1d:4d:

b0:cc:df:7e:aa:9b:8f:e6:57:c1:d7:7b:72:e2:45:

e0:ef:94:32:43:9c:8e:2e:c9:26:dc:11:66:a2:30:

7c:da:83:1d:c8:5f:1c:9d:0d:43:30:a6:ee:d7:66:

69:4f:30:01:e2:df:53:15:c3:77:b5:ae:80:9e:e9:

2e:54:5d:8e:14:03:02:dd:05:48:77:c6:9f:6a:8e:

06:76:ee:a1:42:84:05:f9:5d:40:14:4b:3f:6a:ff:

af:d3:cb:ca:b5:ae:0a:67:e3:49:4b:bd:5a:31:3b:

5b:7e:4c:09:0c:07:95:2b:cc:57:d4:02:8b:5c:4f:

90:61:95:19:8d:0e:fd:25:47:11:53:e0:20:37:88:

78:9a:a2:38:80:5d:c1:95:a7:fc:67:6d:cf:72:b5:

2f:b1:00:59:18:a0:9a:c8:0c:e2:87:17:98:a4:59:

17:80:a6:29:d1:21:28:85:61:7d:f0:4b:5e:44:88:

3b:39:e4:f3:93:54:c4:9b:37:6f:d9:3a:9b:11:2b:

d0:d8:ae:0a:c0:7c:77:1b:49:e3:04:08:10:9e:89:

c1:99:17:61:16:75:bc:54:78:d2:51:56:40:5a:26:

f4:d7

Exponent: 65537 (0x10001)

Bobs Public Key:



-----BEGIN PUBLIC KEY-----

MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAvI8o/7M4dQ4Kgh8nteOJ

UWYXlM6ZwulThfAHO2WxZzFv004knzpaTgLtDHtWp4jRKjbIWdZs0RQSWrYBkO9l

l995/D8Izyr782w+Eq1ieMCVbOAmqVu1LH4gwbQKXiD8JudXaFs9dSXDSKAwzwFP

jmKwtyKihYK63B5AcG+roglzagtwG9R0o+fXierHkG5jDnSIpXs8vNn8B/wNgRFL

I/W2hRH38InTnUNCasBPcq64aqiQCixgM38iaKarFLwJKBUoeIOIVFGShu1oSt+p

hlkTJvHz41MkYqgL6DR0tbzUhIRZvzvu21j8VD2+zzmXwlKgId+Nwl2PbFSVyHl4

4wIDAQAB

-----END PUBLIC KEY-----

RSA Public-Key: (2048 bit)

Modulus:

00:bc:8f:28:ff:b3:38:75:0e:0a:82:1f:27:b5:e3:

89:51:66:17:94:ce:99:c2:e9:53:85:f0:07:3b:65:

b1:67:31:6f:d3:4e:24:9f:3a:5a:4e:02:ed:0c:7b:

56:a7:88:d1:2a:36:c8:59:d6:6c:d1:14:12:5a:b6:

01:90:ef:65:97:df:79:fc:3f:08:cf:2a:fb:f3:6c:

3e:12:ad:62:78:c0:95:6c:e0:26:a9:5b:b5:2c:7e:

20:c1:b4:0a:5e:20:fc:26:e7:57:68:5b:3d:75:25:

c3:48:a0:30:cf:01:4f:8e:62:b0:b7:22:a2:85:82:

ba:dc:1e:40:70:6f:ab:a2:09:73:6a:0b:70:1b:d4:

74:a3:e7:d7:89:ea:c7:90:6e:63:0e:74:88:a5:7b:

3c:bc:d9:fc:07:fc:0d:81:11:4b:23:f5:b6:85:11:

f7:f0:89:d3:9d:43:42:6a:c0:4f:72:ae:b8:6a:a8:

90:0a:2c:60:33:7f:22:68:a6:ab:14:bc:09:28:15:

28:78:83:88:54:51:92:86:ed:68:4a:df:a9:86:59:

13:26:f1:f3:e3:53:24:62:a8:0b:e8:34:74:b5:bc:

d4:84:84:59:bf:3b:ee:db:58:fc:54:3d:be:cf:39:

97:c2:52:a0:21:df:8d:c2:5d:8f:6c:54:95:c8:79:

78:e3

Exponent: 65537 (0x10001)

Alice Private key

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ cat cs01-private-key.pem |

This command outputs encrypted private key to terminal.

-----BEGIN ENCRYPTED PRIVATE KEY-----

MIIFLTBXBgkqhkiG9w0BBQ0wSjApBgkqhkiG9w0BBQwwHAQIUTgOlXCC7qUCAggA

MAwGCCqGSIb3DQIJBQAwHQYJYIZIAWUDBAEqBBCQi64MRcdIq8VVVYEvWMdhBIIE

0AabKHPz5LgQyIyQfwoSCbSoZw3O3MJKsvlkcy1o/LrnudLfYNHoZrqzlQtmTs5f

CpYspPqUMRrb1BecPOrAYiDaZPXK6XqN3WrXac8yMXP5O4U0ndCDd3LsqwuBgHqG

XBLspK5WTNmsqXWEP43eCxQ4HwOleaDrQL54staE4ztAyy+s7JE1StlJtLuDsMMm

SFimiT9jVqu55SrIuweddPZU4sMlC3iymi2HGH2hjMWRL3c2Oup3NJcYH+abDgvT

Q5OdDhWj/PiVmA14MJea3XHd/031lPK0E8fKig2OufI7B9vXRjhbfYsrRL8i6/RA

n3f8zwjdfsNs0aXFFaDnhop3zsy/XRa5xyyKdzbp8BtTZlX3HnqJTjDOquRvli5f

QpDxI25cdjyJR5hX/WWOn9iuCO+h3yKGVamECYg007RGxI2OySz2dpnvHzNQKkBq

FthWuSCFsoSr/YA5/iRu97uZHE2xH1COjOF+5u9I147kxEw6XNtywO4imiIccn9b

GBszMQZMflDrq8u7luaNdAbKsqAWzv5pWI6K9BNlZvgkY7DDr4EcUTyeJHEGt3Qy

h2QSwRRA+P3rotHOdMaw7W0W/odjCUcOtNMXgncz8gYuEQoPHDNX2Zk7dbI7plWF

rkuMKc1CfzVdJdL0FLfUzJvhHbqa2GNoxRM498cm8AOcjujYER8CsRLtyakYdiKG

6Ngz8XIeUaAPAH9g2Sl+fSyahw14oWMxVWAReYJEivI6XbYn7CmDMYCiJtbGO2AE

xalBG4iggqkm4tyD2/+y2OcFTzCMMKiNieYJyanQzQ1sbxZNwe+pEBj4QenLmbrs

1qDMsZwpBdOUNf5PFC46pS1/XnUeuWUNwvIBKJQC02MHZ4LSM/wRYoMSIwAmdeY6

SRnRtaq2ZMEVY4wxau7Uo0DvDx6QUhRIVBB6M+tayeDafujYkhxNduct33/Z79ou

1Ngpqdu7ou6UeNUJ5LrrZFNxs5XRohm2ykkEMbsoeJ7mlgpnlUumEA4ZhQjYqB+x

acF5QSNcyXIMf5Zpkk9yirYje0dvjk+mO7y/ALuTqBTBfoOZTzQElJMKW0o9AkHe

nH5bx0bCuLsIHwREtjix8Syd+GhsbcpkQ6OGoygY1+sbEd0BK3MURGteGJwy7Tv9

ALhUUKqjAkk/4kSGw+yovFQcv9T88KInKfwRt3AjnN0B8MZd+APTdNy1NnGbTQfa

zeZ2eQ4EpcnaiS6RGP3QYGlLVgYSH+LVVM29rHAeH62O3FBB+e99HCkzU+69b6gg

Z7WeIkDsrNNzglFaunskjhgAh4ur73qH3ykH78pc814ay5jJf9NCF43DBSMpwem0

DSaDoO2pl7O4EijYtBIFZiDsD/07lF5WzKTTBUa1rcR6e1nyJjxFTrljSIi9QwJv

Cwxr2LhuqpDlTRk1MctZXJg1N0QYxOQTXOaw3MZYjro1QzJpUviemRh1P9zCPBaR

RhixgPRJ5L/s6MPcJ0Q9BYNwcHtQE3cnuz9dTKbHO0A3hRZfN2EqmvugsxVXhKKe

8uLw9Q3NZmyf++RZC37zVpy/FTeNwjKuAfJfnm8hjmPW/AL6X5rGfJwVBAYn+LP8

trKE7ZrCkJazz5Sl/X5TBca7BVmk7hDO3ZKjavZzFg2O

-----END ENCRYPTED PRIVATE KEY-----

Bobs private key:



-----BEGIN ENCRYPTED PRIVATE KEY-----

MIIFLTBXBgkqhkiG9w0BBQ0wSjApBgkqhkiG9w0BBQwwHAQII2jLQDNAEVMCAggA

MAwGCCqGSIb3DQIJBQAwHQYJYIZIAWUDBAEqBBCTAXfexEd/UowRlcy02WCKBIIE

0HYqvmkIZTlk4oYCfPkc/UzGtXuNuFrMqt3fSU+Q4ErhJfpYlKwCk5qKJAnN8Y7Q

kt6Ta1YgX+IwnLsg60/WwbI452v/M3RejvAAzUjOttF3rN8YAETQHTW8Fg1yWCrW

xPLkn1s1uwv0LXL149bDA2CjtJcAMTYzkK7k7n8FgefT+1rrfb9W+KdrTFh5Q3FK

fUhoEx+xEpMMMsxu+VuMrCA566znK+YuwXBUEb4n5YJOkB+HhmyE7fMeTu9VEAGz

Yr5Z84PHSRKEKucN0KpEvjd778mkgytG4BN+ykM+EFzzOEFKk1BUcZ5gCtbioXl2

LziXP/43qIFQ2IBqDBVadEfvXQEHkbtzFwkqxWNoyWjipGkeyZg61eRcqn+/eE86

XxW41cehSOd9VvkLVRkH75mSTrIdxZCTAut7R+VA4PoMq6cFxC3rnnak0dRTIpFM

Lb3L7iBBgSB9q0KfUijnbJwac2xPUAlDGmBUtDCdc673kRYfMHzyA+zJw1nol4IF

eXJkJUA5RAeDYDg2LR49kED/YqPHDM/UwxXYlJFb2EPSWXaHKNSQvVP5IcWqudoD

3niboPIw7w5v7kUj4iTo4syFKs/RjgylfKwIv4TNhxVESXTVT4y8kRDTgM70vZ7s

MXwsfws7gvsyd/nj7Q0JRoEpBAq4Lv2q+zzlUw/JH8uf7GyJ81xCYA4fPFbw5qMK

FlQoBom15L3QLQpRUxWF50TSk2uaO/TNQvC6yJNAcns3qkBP58IPsl2rfrPZuGFV

lFnpio0gFSJFGISChr1wBO4nw4Ot2hVWb4w6uYCvfQe1h504G55rJUxh7bB10N1X

q3lBw9fIxmdfOX6r8ZiLIQ/yy217bykhzlrej+eP/iJK6swpPP0u5bFrrC+RC/HE

FDF7B962MYmimiCFnAmVfXTpRY1Xl8goekJI3+Zo0XUcNy2roR8+9bNDdIDsELih

er8ZVYvyG+mjZB+MzkW8FnB0L4lwaweQyKUQ7ryulYScD5yr5cL4wJ6aQZmBZdxu

Hp5sHihzPF8dmv2OnqkdIBgf0HMcmrYDNFC/dyZIC5wjwB/fBK9Z1tHMQ23ERvNW

3zgo8BGgrcFQKpzyrqnkNQ/RvHiZIOV260RVgh66xVkTzGlukac5G9and9UDmEza

gy9vWoTVGQB4f719kagPHNYutNEIlYrDsEv7kb9c5KogkiB5W0C4uzFGnOtRJeXY

CwbaOl+afNFinMzBRlPalEPzY7loYbZTiwFeojEsq9y7G2iIdlLS+q/XzNxgnbnt

QQ0yNNOZBtPkiI7rFNJ9GlyuCZ6gaZq3xdpAJt60lIGCHkRglSjq72cl0E4x7mhM

/i5KdaqH76JTU8Wd9fUFoW7/fCkNXYzXQvTBD50cqOK1jUNsysUbBKQg1n9yq5h2

ZWOPs6LCrV8BwYk7EVrQjj7NULguM+iHqATe+47YeeHuA/dfYLqjetq/7+zUlTFu

2yueSW4b5es61JPB9s45eRDLgZtAnGerCb2NI9EGcDsGiTIcW7F2CqMuWDsS79+K

6IMoW4pNAHJPt1P+3U4/WIcmevJYf6yGtFB6acUojHPEFIluzxU7IKicLplwmq2U

urmw9o0TDVrkaxg44LHxStbDj4iI6nL6HGjYxozd74aE

-----END ENCRYPTED PRIVATE KEY-----

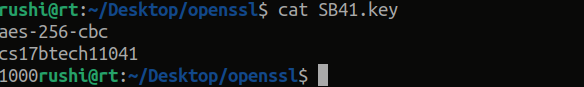
**Step 2**: Alice creates a text file named SA.key with this info <symmetric encryption algo, passphrase and no. of iterations>. Bob also does the same thing. These serve like keys for decrypting files exchanged in each way.

SA01.key contains info “aes-256-cbc, cs17btech11001, 1000”. AES-256 symmetric encryption algorithm is used to generate key for decrypting file sent from Alice to Bob. So by using the passphrase and number of iterations and symmetric encryption algorithm, a key would be generated by repeatedly applying the sha256 algorithm to passphrase initially and to output digest afterwards for a given number of iterations to generate the key. This key is used to decrypt any file sent by Alice to Bob.

Alice SA01.key



Bob SB41.key:



**Step 3**: Alice has to securely send SA.key to Bob. Devise a mechanism in such a way that only Bob can see that message and verify it indeed came from Alice. Similarly, Bob has to securely send his SB.key to Alice and prove its authenticity and integrity.

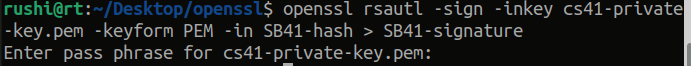
Alice has SA.key. Now, she has to securely send it to Bob. For this, she will hash SA.key using SHA1 algorithm to generate digest and then encrypt it using her private key. Then she will send this digitally signed file along with encrypted SA.key (encrypt using Bob public key so that only Bob can see it) to Bob. Thus Bob can verify that the file is indeed sent to him by Alice by verifying the digitally signed file. So Alice sends encrypted SA.key and signed file to Bob.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl dgst -sha256 SA01.key > SA01-hash |



This command is used to hash SA01.key ,SB41.key and generate digest respectively.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl rsautl -sign -inkey cs01-private-key.pem -keyform PEM -in SA01-hash > SA01-signature |



This command is used to sign the digest with Alice's and Bobs private key respectively.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl rsautl -encrypt -pubin -inkey cs41-public-key.pem -in SA01.key -out enc-SA01-key.txt |

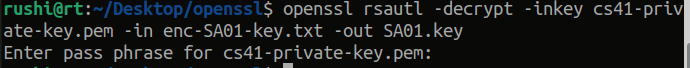


This command is used to encrypt SA01.key and SB41.key using Bob's public key and Alice’s public key respectively.

After this, Alice sends encrypted SA01.key (enc-SA01-key.txt) and digitally signed digest of SA01.key (SA01-signature) to Bob.

**Verification of SB.key by Alice and SA.key by Bob**

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl rsautl -decrypt -inkey cs01-private-key.pem -in enc-SB41-key.txt -out SB41.key |



This command is used by Alice to decrypt the encrypted SB-41.key sent by Bob. (Here only Alice can decrypt it using her own private key and no one else).

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl rsautl -verify -inkey cs41-public-key.pem -pubin -keyform PEM -in SB41-signature > generated-hash |



This is used by Alice to generate a hash digest of the signature file sent by Bob using the public key of Bob she had received by mail.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl dgst -sha256 SB41.key > actual-hash |



This is used by Alice to generate the actual hash of SB41.key (using sha2) sent by Bob.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ diff generated-hash actual-hash |



This is used to compare generated-hash and actual-hash. This proves that indeed Bob has sent the message since the sha2 hash of decrypted SB41.key and hash obtained by decrypting signature of hash sent by Bob using his public key are the same. (proof of authenticity)

**Step 4**: Alice encrypts a large file (some PDF) with SA.key and sends it to Bob so that he could decrypt it with the same SA.key. Similarly, Bob should send some large file securely to Alice.

A higher iteration count increases the time required to brute-force the resulting file.

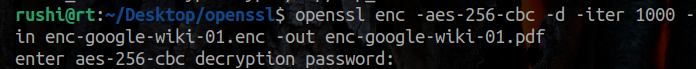
|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl enc -aes-256-cbc -e -iter 1000 -salt -in Google-wiki-01.pdf -out enc-google-wiki-01.enc |



This is used by Alice to encrypt the test-file using aes algorithm and number of iterations mentioned in SA.key and while taking input passphrase and outputting encrypted file.

She then send it to Bob through email.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl enc -aes-256-cbc -d -iter 1000 -in Report.enc -out decrypted-file.dec |



The encrypted file sent by Alice can be decrypted by Bob using same encryption algorithm and number of iterations and passphrase mentioned in SA.key sent by Alice earlier.

**Part B: Alice (Browser), Bob (web server) and Charlie (Root CA)**

**Step 1**: Charlie generates a self-signed certificate named charlier-ca.pem or charlie-ca.crt as he is the root CA

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl req -newkey rsa:2048 -nodes -keyout charlie-private.pem -x509 -days 365 -out charlie-ca.crt |

This command will generate a self-signed certificate charlie-ca.crt along with charlie-ca.key private key (rsa algorithm with key size 2048 bits). Certificate is valid for 365 days.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl x509 -in charlie-ca.crt -text |

This command is used to view the charlie-ca.crt in text form.

Charlie-ca.crt content:-

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl x509 -in charlie-ca.crt |

-----BEGIN CERTIFICATE-----

MIID3zCCAsegAwIBAgIUeTOpcjQjsNpnyqesWGPpOWpXvf0wDQYJKoZIhvcNAQEL

BQAwfzELMAkGA1UEBhMCSU4xCzAJBgNVBAgMAk1IMQwwCgYDVQQHDANOZ3AxITAf

BgNVBAoMGEludGVybmV0IFdpZGdpdHMgUHR5IEx0ZDEQMA4GA1UEAwwHY2hhcmxp

ZTEgMB4GCSqGSIb3DQEJARYRY2hhcmxpZUBnbWFpbC5jb20wHhcNMjEwMjA3MTYy

NTUwWhcNMjIwMjA3MTYyNTUwWjB/MQswCQYDVQQGEwJJTjELMAkGA1UECAwCTUgx

DDAKBgNVBAcMA05ncDEhMB8GA1UECgwYSW50ZXJuZXQgV2lkZ2l0cyBQdHkgTHRk

MRAwDgYDVQQDDAdjaGFybGllMSAwHgYJKoZIhvcNAQkBFhFjaGFybGllQGdtYWls

LmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANEAGI1gAzMcPk6+

eVXOFLp3WDiYpveQdG4Dsw00dFu63BB9ueX803shbER0vVWcD/fWIvqukimQmKr1

KvCCpdfxaodkbcDi/77nFT4hy3MjHMrL+19V1MI79c8H320Dv/K4D4BqnhDU9iBf

o0lXAElQXJzex5d/b6dHguLOVE/nt0GCCiSFgbjbpOP24BMDTZfTdyAsmoriHrBD

ijQ3dOYEC8AFroC+qJOJGDlS/URM2XOaU/4/KreqIlpxb2xv7HLr1ZMq19KnjFY+

AkAh8ZcimqiTEdSFFz1OT1iw03Rh2ksAUQh2HGTIBcuvKHd1cSM5v7NfTmO2ca6I

trrD3HcCAwEAAaNTMFEwHQYDVR0OBBYEFE2gW1k0/iPq1/8bYLYZQg2nePwpMB8G

A1UdIwQYMBaAFE2gW1k0/iPq1/8bYLYZQg2nePwpMA8GA1UdEwEB/wQFMAMBAf8w

DQYJKoZIhvcNAQELBQADggEBAKJsfErjGRtBA7vb6+HqQ2G/L3pwk3q9i1yeq8I1

Liw+BcPI8DAItOizuJYLNOkmEK78aH9xilpCP/GqNhPhQ0C8j85+GQ2kL+Q7DNFb

+UMHyHaN2TA47ejV37Nb37ashYlnFOyOKqjG7aWj9dsfip3yLzFZ9yfel8YPxmpX

nz6B8GK2n8aLwYvueVji+OcZlrHLj4TldlmoS1JvTEl4v0YcWDymgXSv3mRITZt3

XpBMFDIoh10HkNj/+EKNygme81ox5l1V/dK18MV8nkdTL2aHy6giZgOJYCfv+OW1

F9lDOLQZ0AYuY6AV+QCI68ZEfugWp1xNAwOxZwljwRmnP1M=

-----END CERTIFICATE-----

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl pkey -in charlie-public.pem -text -pubin |

Charlie-public.pem

-----BEGIN PUBLIC KEY-----

MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA0QAYjWADMxw+Tr55Vc4U

undYOJim95B0bgOzDTR0W7rcEH255fzTeyFsRHS9VZwP99Yi+q6SKZCYqvUq8IKl

1/Fqh2RtwOL/vucVPiHLcyMcysv7X1XUwjv1zwffbQO/8rgPgGqeENT2IF+jSVcA

SVBcnN7Hl39vp0eC4s5UT+e3QYIKJIWBuNuk4/bgEwNNl9N3ICyaiuIesEOKNDd0

5gQLwAWugL6ok4kYOVL9REzZc5pT/j8qt6oiWnFvbG/scuvVkyrX0qeMVj4CQCHx

lyKaqJMR1IUXPU5PWLDTdGHaSwBRCHYcZMgFy68od3VxIzm/s19OY7Zxroi2usPc

dwIDAQAB

-----END PUBLIC KEY-----

RSA Public-Key: (2048 bit)

Modulus:

00:d1:00:18:8d:60:03:33:1c:3e:4e:be:79:55:ce:

14:ba:77:58:38:98:a6:f7:90:74:6e:03:b3:0d:34:

74:5b:ba:dc:10:7d:b9:e5:fc:d3:7b:21:6c:44:74:

bd:55:9c:0f:f7:d6:22:fa:ae:92:29:90:98:aa:f5:

2a:f0:82:a5:d7:f1:6a:87:64:6d:c0:e2:ff:be:e7:

15:3e:21:cb:73:23:1c:ca:cb:fb:5f:55:d4:c2:3b:

f5:cf:07:df:6d:03:bf:f2:b8:0f:80:6a:9e:10:d4:

f6:20:5f:a3:49:57:00:49:50:5c:9c:de:c7:97:7f:

6f:a7:47:82:e2:ce:54:4f:e7:b7:41:82:0a:24:85:

81:b8:db:a4:e3:f6:e0:13:03:4d:97:d3:77:20:2c:

9a:8a:e2:1e:b0:43:8a:34:37:74:e6:04:0b:c0:05:

ae:80:be:a8:93:89:18:39:52:fd:44:4c:d9:73:9a:

53:fe:3f:2a:b7:aa:22:5a:71:6f:6c:6f:ec:72:eb:

d5:93:2a:d7:d2:a7:8c:56:3e:02:40:21:f1:97:22:

9a:a8:93:11:d4:85:17:3d:4e:4f:58:b0:d3:74:61:

da:4b:00:51:08:76:1c:64:c8:05:cb:af:28:77:75:

71:23:39:bf:b3:5f:4e:63:b6:71:ae:88:b6:ba:c3:

dc:77

Exponent: 65537 (0x10001)

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ cat charlie-private.pem |

Charlie-private.pem

-----BEGIN PRIVATE KEY-----

MIIEvwIBADANBgkqhkiG9w0BAQEFAASCBKkwggSlAgEAAoIBAQDRABiNYAMzHD5O

vnlVzhS6d1g4mKb3kHRuA7MNNHRbutwQfbnl/NN7IWxEdL1VnA/31iL6rpIpkJiq

9SrwgqXX8WqHZG3A4v++5xU+IctzIxzKy/tfVdTCO/XPB99tA7/yuA+Aap4Q1PYg

X6NJVwBJUFyc3seXf2+nR4LizlRP57dBggokhYG426Tj9uATA02X03cgLJqK4h6w

Q4o0N3TmBAvABa6AvqiTiRg5Uv1ETNlzmlP+Pyq3qiJacW9sb+xy69WTKtfSp4xW

PgJAIfGXIpqokxHUhRc9Tk9YsNN0YdpLAFEIdhxkyAXLryh3dXEjOb+zX05jtnGu

iLa6w9x3AgMBAAECggEBAMDar800vX8aHryWsmvk/+eYXMXMxrVGjMk/jKcwsfY7

FPn6qeF5TMQf8UGx30vx+CIfbKOZFrJYBSgrlvXb+KCs2joZB9WnIYrMvSNaF9wk

mFrhoxEzqGGa5pWllBuKDEMyaMt8lmK9wGfbGFeCencsOmlegRpgs2zOZg5XiiE3

6v/FJpmdHnEupEoeuFq68+ErjBddeJRR2A6GXEyfeq/p2ZHmadeSFxsO3aFw+vyZ

tNNX7vwE7iatr6VMr2kAM2M5P7D1FFCTiFZBwIGpuhsVK1u25b1Hj95n18XbzC4q

fduLqW3a51WiBw4EZnXQOjlixTkDlX6iYX0+FGjaToECgYEA8KG8XFdcv2yO2Gpa

dvIqmgVU83j64A2mMtCfnd3CSavktWpYiAgQKnxX7dKrBP+iiRMtfYOnam2Y/Dx0

ULHwS7QJeRyVp/rk8eOvydnKzk6IQd1fQU2JeOKzRjeGKttiw24am2EXcpV+kSRs

sn2vhEEA4Di5n+wSv7XFWpEKc9cCgYEA3lkx+utGizBL/yS9MNQpC6PEp0jWaNtJ

OqKpx0r4O35ga8f/t1dDy6WZC8j3SBvNo2/BMjRfVqWKCA6cvlKIPr+6aaCu4rkw

P5hMcPVHR/lIqSL5Fcpg9y8dW+PuPel+GlV+1GOYLejQ4/p+3e8tv80tx4Lw3rD5

iTdBfHViyGECgYBdgcwLahtte4bqzDFJRYOOhIHldt6uUD+EQZO1KeaE8FoYsTGj

8tYtXFJi5Vl1wMztTVmBC7bz4ZDwe6GLdgn1sQl5WcVCGHn9EUhPPOXVU6kZRyfr

3XCNXDLkF1F4t/XiEZgfRxRwf21Gg4wFtFePIh9wAjsHzRngC4ecUP5p0wKBgQDS

InfizNIZn2ZjVmRarlewQXEMLAE+jefSO4K673ZUO6aCljxtGFLhNrrCcYaKJoSc

7nQmr2pe6AngfXbTRAqxSt+xDgPaD+yyXYv4wUZBli1+c6f2WRJ06ZkUjABPkMj1

uFEqBon+8WXh345gzrds6ktWn8BmpdjfPlJsfkACYQKBgQCepfebQ5z7A936kJBk

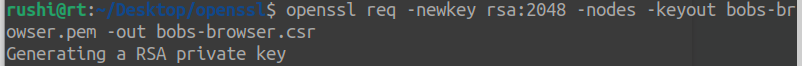
LFcuMSZWWb7vsBKQ34V/oMUquXvv6ctgr7VwgV1NEfK+t1LUa5CwmMDsOSiGv9zE

XOqU0S7EDPrcdDlDUNdCYpmwb8Ipn/7/ElULjCUanSb8ls/xDXJIY1wXO510t2Mq

D2nXz82yJ5uu9jsKeEWBNxxyTA==

-----END PRIVATE KEY-----

**Step 2**: Bob generates CSR named bob-browser.csr and emails it to Charlie for providing end-user cert named bob-browser.crt



This command used by Bob will generate CSR along with private key (RSA algorithm with key size 2048 bits).



Certificate Request:

Data:

Version: 1 (0x0)

Subject: C = in, ST = mah, L = nanded, O = Internet Widgits Pty Ltd, CN = rushi

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public-Key: (2048 bit)

Modulus:

00:b9:ba:30:3c:0a:64:98:91:ce:f9:51:b7:8f:02:

9c:f9:8d:1e:79:b2:12:bf:4d:4a:b6:f4:6f:d1:6c:

73:3b:63:3c:6f:a0:a0:ec:6d:e1:6f:7c:e2:d0:b1:

dc:8e:ec:c5:df:71:67:91:85:3f:d3:e9:89:f5:9e:

aa:a8:a5:51:44:2a:5d:45:43:66:ca:af:38:42:e5:

f1:f8:88:f5:ac:45:5c:ea:e0:9f:54:b3:ba:f0:6c:

39:de:ea:54:25:2a:3a:f0:52:4a:d8:49:40:c5:e2:

3f:97:95:a4:f7:ba:b4:a4:ad:11:45:d6:f3:01:d3:

4a:a9:c5:96:25:07:f0:69:f5:5a:58:0c:d8:77:cb:

18:ff:f0:d6:52:fe:8d:16:99:20:16:b8:2f:32:10:

52:1a:6e:d8:7f:2c:66:5e:20:a6:40:8a:26:7e:6f:

1c:1e:e8:8d:83:11:95:d2:2e:a9:43:62:1f:f3:77:

cc:a0:82:de:12:5b:db:8b:2c:fb:09:c4:51:bf:15:

9e:4f:8d:30:80:ed:89:39:12:e4:80:17:96:3c:29:

ca:79:96:8f:2d:c0:50:45:0b:41:1b:52:e9:1d:6f:

c2:81:76:93:bf:bf:63:5f:1b:39:5c:15:42:2f:3c:

f8:61:8f:70:c3:26:01:0e:f3:f6:99:7e:bc:07:f7:

99:75

Exponent: 65537 (0x10001)

Attributes:

unstructuredName :xeta

challengePassword :rushi

Signature Algorithm: sha256WithRSAEncryption

9d:ff:ae:68:11:b8:7c:55:54:10:84:41:0d:71:98:44:03:01:

64:3b:95:09:75:48:d7:12:07:26:b9:46:7f:f1:82:59:93:1f:

82:65:85:f1:87:35:73:95:45:ab:68:28:51:0d:0d:23:6e:52:

a2:ab:a4:28:27:ed:ed:be:61:e0:7d:b9:90:2e:18:a1:81:c0:

6c:c0:9b:5f:1b:6e:64:d8:f3:4d:52:7d:5d:0d:a7:8f:a2:6b:

aa:18:2e:d7:f0:16:b0:8b:de:7c:62:58:1a:77:ae:e6:28:7f:

16:c4:8f:76:af:0d:be:e5:0e:44:5e:99:ad:73:e2:ee:80:8d:

2b:ae:74:06:30:63:f6:16:f1:78:66:1d:7c:3c:33:c6:d5:19:

a2:35:36:1b:e4:b4:65:15:b2:a9:70:07:41:ac:b6:a6:28:bc:

56:ec:e6:d2:6e:91:ca:ef:f6:26:42:56:21:66:f8:c5:78:d0:

c7:29:53:3c:ef:3b:aa:e6:ec:d3:ba:60:81:fa:e1:a4:2f:bb:

64:89:e4:f9:2b:ef:3f:6f:a3:02:f5:70:e3:f6:17:db:d2:93:

c6:04:93:50:df:13:b5:8e:f1:8a:3e:16:8b:48:6f:77:60:d0:

ca:f1:90:d3:9e:72:42:bb:cb:94:6e:99:78:75:9a:4a:18:69:

de:ed:d3:af

-----BEGIN CERTIFICATE REQUEST-----

MIICzzCCAbcCAQAwXzELMAkGA1UEBhMCaW4xDDAKBgNVBAgMA21haDEPMA0GA1UE

BwwGbmFuZGVkMSEwHwYDVQQKDBhJbnRlcm5ldCBXaWRnaXRzIFB0eSBMdGQxDjAM

BgNVBAMMBXJ1c2hpMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAubow

PApkmJHO+VG3jwKc+Y0eebISv01KtvRv0WxzO2M8b6Cg7G3hb3zi0LHcjuzF33Fn

kYU/0+mJ9Z6qqKVRRCpdRUNmyq84QuXx+Ij1rEVc6uCfVLO68Gw53upUJSo68FJK

2ElAxeI/l5Wk97q0pK0RRdbzAdNKqcWWJQfwafVaWAzYd8sY//DWUv6NFpkgFrgv

MhBSGm7YfyxmXiCmQIomfm8cHuiNgxGV0i6pQ2If83fMoILeElvbiyz7CcRRvxWe

T40wgO2JORLkgBeWPCnKeZaPLcBQRQtBG1LpHW/CgXaTv79jXxs5XBVCLzz4YY9w

wyYBDvP2mX68B/eZdQIDAQABoCswEwYJKoZIhvcNAQkCMQYMBHhldGEwFAYJKoZI

hvcNAQkHMQcMBXJ1c2hpMA0GCSqGSIb3DQEBCwUAA4IBAQCd/65oEbh8VVQQhEEN

cZhEAwFkO5UJdUjXEgcmuUZ/8YJZkx+CZYXxhzVzlUWraChRDQ0jblKiq6QoJ+3t

vmHgfbmQLhihgcBswJtfG25k2PNNUn1dDaePomuqGC7X8Bawi958Ylgad67mKH8W

xI92rw2+5Q5EXpmtc+LugI0rrnQGMGP2FvF4Zh18PDPG1RmiNTYb5LRlFbKpcAdB

rLamKLxW7ObSbpHK7/YmQlYhZvjFeNDHKVM87zuq5uzTumCB+uGkL7tkieT5K+8/

b6MC9XDj9hfb0pPGBJNQ3xO1jvGKPhaLSG93YNDK8ZDTnnJCu8uUbpl4dZpKGGne

7dOv

-----END CERTIFICATE REQUEST-----

Bob emails bob-browser.csr to Charlie. Charlie provides Bob with end-user certificate bob-browser.crt.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl x509 -req -days 360 -in bobs-browser.csr -CA charlie-ca.crt -CAkey charlie-private.pem -CAcreateserial -out bob-browser.crt |

This command is used by CA Charlie to generate end-user certificate bob-browser.crt

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl x509 -in bob-browser.crt -text |

Bob-brower.crt contents

Certificate:

Data:

Version: 1 (0x0)

Serial Number:

09:bc:20:68:9f:79:34:ee:42:5a:22:f2:13:46:a8:23:01:ab:f8:5d

Signature Algorithm: sha256WithRSAEncryption

Issuer: C = IN, ST = MH, L = Ngp, O = Internet Widgits Pty Ltd, CN = charlie, emailAddress = charlie@gmail.com

Validity

Not Before: Feb 7 17:01:55 2021 GMT

Not After : Feb 2 17:01:55 2022 GMT

Subject: C = in, ST = mah, L = nanded, O = Internet Widgits Pty Ltd, CN = rushi

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public-Key: (2048 bit)

Modulus:

00:b9:ba:30:3c:0a:64:98:91:ce:f9:51:b7:8f:02:

9c:f9:8d:1e:79:b2:12:bf:4d:4a:b6:f4:6f:d1:6c:

73:3b:63:3c:6f:a0:a0:ec:6d:e1:6f:7c:e2:d0:b1:

dc:8e:ec:c5:df:71:67:91:85:3f:d3:e9:89:f5:9e:

aa:a8:a5:51:44:2a:5d:45:43:66:ca:af:38:42:e5:

f1:f8:88:f5:ac:45:5c:ea:e0:9f:54:b3:ba:f0:6c:

39:de:ea:54:25:2a:3a:f0:52:4a:d8:49:40:c5:e2:

3f:97:95:a4:f7:ba:b4:a4:ad:11:45:d6:f3:01:d3:

4a:a9:c5:96:25:07:f0:69:f5:5a:58:0c:d8:77:cb:

18:ff:f0:d6:52:fe:8d:16:99:20:16:b8:2f:32:10:

52:1a:6e:d8:7f:2c:66:5e:20:a6:40:8a:26:7e:6f:

1c:1e:e8:8d:83:11:95:d2:2e:a9:43:62:1f:f3:77:

cc:a0:82:de:12:5b:db:8b:2c:fb:09:c4:51:bf:15:

9e:4f:8d:30:80:ed:89:39:12:e4:80:17:96:3c:29:

ca:79:96:8f:2d:c0:50:45:0b:41:1b:52:e9:1d:6f:

c2:81:76:93:bf:bf:63:5f:1b:39:5c:15:42:2f:3c:

f8:61:8f:70:c3:26:01:0e:f3:f6:99:7e:bc:07:f7:

99:75

Exponent: 65537 (0x10001)

Signature Algorithm: sha256WithRSAEncryption

63:8f:ce:8e:20:d9:1e:e7:0a:cd:6f:12:55:6e:3c:c1:cb:e7:

95:af:9b:ad:17:b4:5b:0c:05:34:b8:b3:44:5b:e8:ed:48:85:

92:3c:97:80:0c:2b:e4:6a:e9:99:6c:82:52:55:de:db:6f:4e:

f3:5a:a5:d5:61:d1:b7:d8:78:57:55:a0:d9:38:5d:9f:b7:c0:

02:61:12:d4:8d:b6:77:62:e5:d1:6b:7f:51:01:5d:36:b5:3a:

10:4e:b4:09:78:80:27:9f:61:32:57:39:f6:d5:28:4a:95:e0:

d1:9e:5c:5d:86:46:ad:64:e5:d1:f9:c6:2a:17:97:4d:0d:73:

c4:5c:70:e0:f7:b9:bd:06:e2:90:c0:ef:d8:71:10:cd:2a:7c:

d3:46:42:d9:5c:a0:e9:b8:c6:c5:4d:5b:3a:2e:bb:01:59:0b:

14:a9:7f:0a:b3:15:7b:2d:8e:05:be:c5:5d:79:65:26:4b:a4:

a2:79:7c:ce:98:a8:05:75:e7:21:7b:73:d6:f2:f0:9b:1c:db:

a5:2e:f9:4e:e6:c4:ac:6d:ca:5b:bb:57:d4:24:da:69:e0:fa:

cb:f9:d8:e6:22:7f:9c:05:7c:39:90:62:52:61:71:88:ee:bd:

06:af:f0:ba:19:02:fe:3b:c3:3d:63:c3:d8:93:e6:b6:47:75:

5f:1d:86:6b

-----BEGIN CERTIFICATE-----

MIIDZTCCAk0CFAm8IGifeTTuQloi8hNGqCMBq/hdMA0GCSqGSIb3DQEBCwUAMH8x

CzAJBgNVBAYTAklOMQswCQYDVQQIDAJNSDEMMAoGA1UEBwwDTmdwMSEwHwYDVQQK

DBhJbnRlcm5ldCBXaWRnaXRzIFB0eSBMdGQxEDAOBgNVBAMMB2NoYXJsaWUxIDAe

BgkqhkiG9w0BCQEWEWNoYXJsaWVAZ21haWwuY29tMB4XDTIxMDIwNzE3MDE1NVoX

DTIyMDIwMjE3MDE1NVowXzELMAkGA1UEBhMCaW4xDDAKBgNVBAgMA21haDEPMA0G

A1UEBwwGbmFuZGVkMSEwHwYDVQQKDBhJbnRlcm5ldCBXaWRnaXRzIFB0eSBMdGQx

DjAMBgNVBAMMBXJ1c2hpMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA

ubowPApkmJHO+VG3jwKc+Y0eebISv01KtvRv0WxzO2M8b6Cg7G3hb3zi0LHcjuzF

33FnkYU/0+mJ9Z6qqKVRRCpdRUNmyq84QuXx+Ij1rEVc6uCfVLO68Gw53upUJSo6

8FJK2ElAxeI/l5Wk97q0pK0RRdbzAdNKqcWWJQfwafVaWAzYd8sY//DWUv6NFpkg

FrgvMhBSGm7YfyxmXiCmQIomfm8cHuiNgxGV0i6pQ2If83fMoILeElvbiyz7CcRR

vxWeT40wgO2JORLkgBeWPCnKeZaPLcBQRQtBG1LpHW/CgXaTv79jXxs5XBVCLzz4

YY9wwyYBDvP2mX68B/eZdQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBjj86OINke

5wrNbxJVbjzBy+eVr5utF7RbDAU0uLNEW+jtSIWSPJeADCvkaumZbIJSVd7bb07z

WqXVYdG32HhXVaDZOF2ft8ACYRLUjbZ3YuXRa39RAV02tToQTrQJeIAnn2EyVzn2

1ShKleDRnlxdhkatZOXR+cYqF5dNDXPEXHDg97m9BuKQwO/YcRDNKnzTRkLZXKDp

uMbFTVs6LrsBWQsUqX8KsxV7LY4FvsVdeWUmS6SieXzOmKgFdeche3PW8vCbHNul

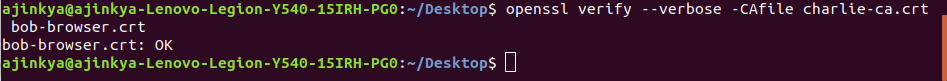
LvlO5sSsbcpbu1fUJNpp4PrL+djmIn+cBXw5kGJSYXGI7r0Gr/C6GQL+O8M9Y8PY

k+a2R3VfHYZr

-----END CERTIFICATE-----

**Step 3**: Alice gets crts of Charlie and Bob and verify that Bob's certificate is valid.

|  |
| --- |
| ajinkya@ajinkya-Lenovo-Legion-Y540-15IRH-PG0:~/Desktop$ openssl verify --verbose -CAfile charlie-ca.crt bob-browser.crt |



This command is used by Alice to verify that Bob’s certificate is valid. It uses bob-browser.crt and charlie-ca.crt to verify if bob-browser.crt is valid.