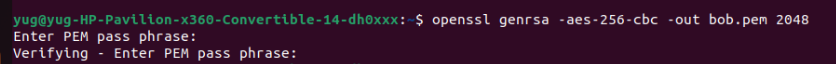
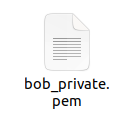
**Openssl Assignment**

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

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

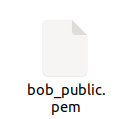
**Bob’s**

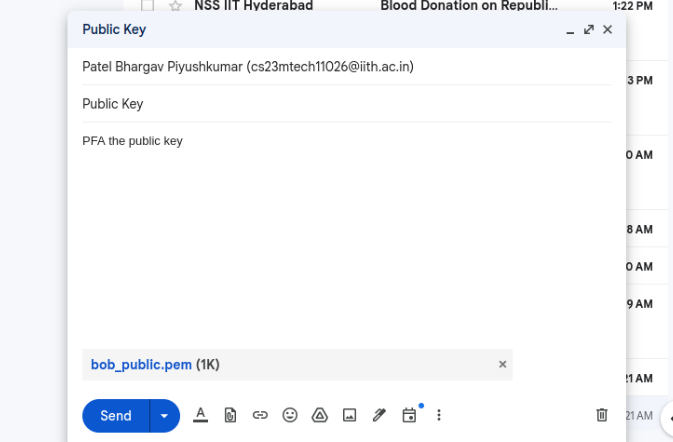
*Generating password protected private key*



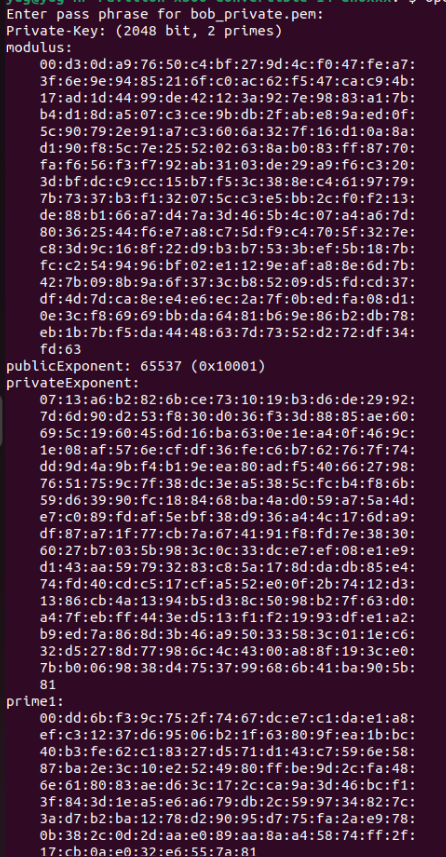
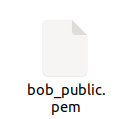
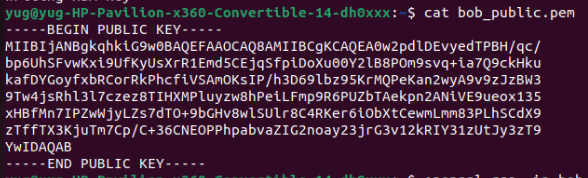
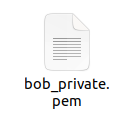
*Extracting Public key from Private key* 







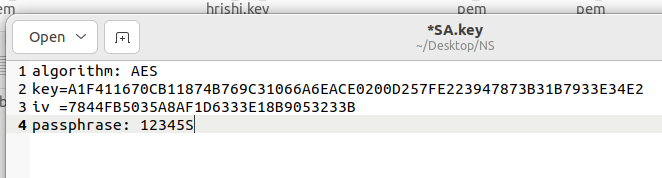


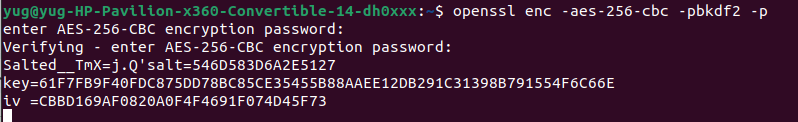


Similarly the same scenario happens at Alice's end. On completion of this both Alice and Bob have each other’s public key that is exchanged over trusted third party i.e. mail.

**2. Alice creates a text file named SA.key with this info <symmetric encryption algo, its parameters and passphrase>. Bob also does the same thing (SB.key). These serve like keys for decrypting files exchanged in each way.**

Bob and Alice generate a file name SA.key and SB.key respectively. Which contains the name of algorithm, passphrase and parameter <Key, IV> to the specified algorithm which inturn is generated from PBKDF2.





**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 without any tampering. Similarly, Bob has to securely send his SB.key to Alice and prove its authenticity and integrity.**

This step involves exchange of two files from both alice and bob i.e. one containing the encrypted version of S[A/B].key and the other the signed digest of the same keys. The two files combined can take care of Integrity, Confidentiality, authenticity and non repudiation.



Encrypting SA.key using Alice’s public key:



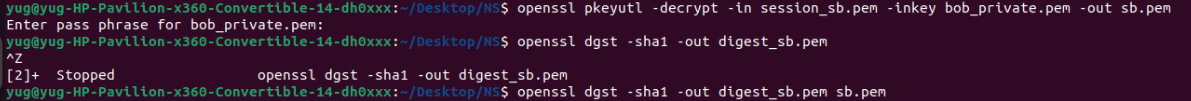
Signing the Digest of SA.key using Bob’s Private key:



*Bob’s Efforts in Sending his keys (SA.key) to Alice*



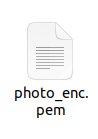
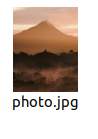
Decrypting SB.key and Generating its hash

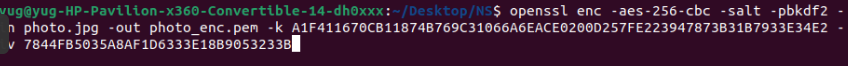


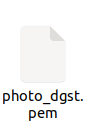
Verifying it with the sign received



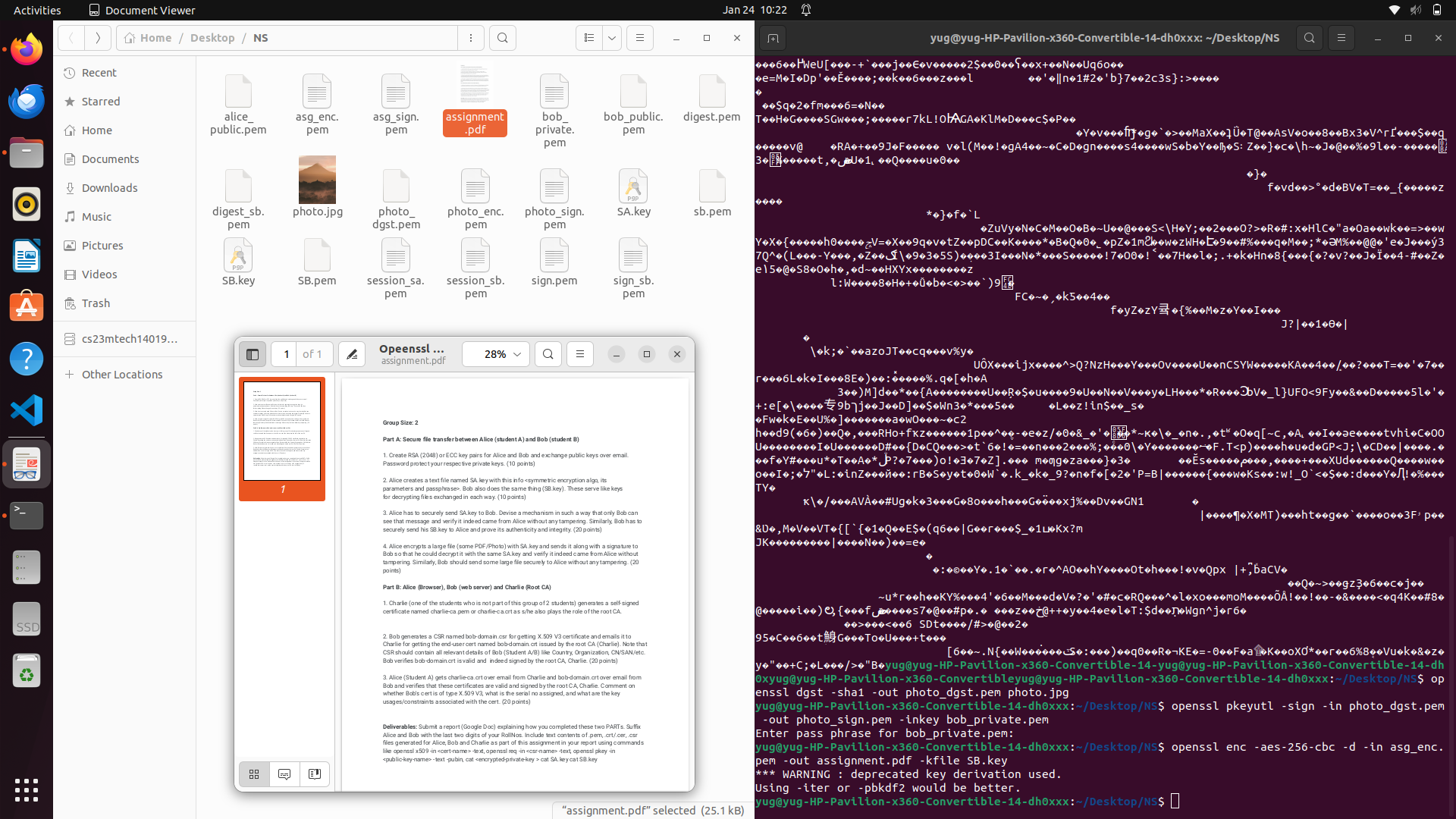
*Bob’s Efforts in Verifying Alice’s Key (SB.key)*

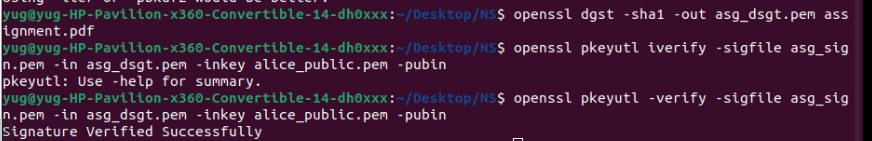
**4. Alice encrypts a large file (some PDF/Photo) with SA.key and sends it along with a signature to Bob so that he could decrypt it with the same SA.key and verify it indeed came from Alice without tampering. Similarly, Bob should send some large file securely to Alice without any tampering.**

Bob sends a *Photo.jpg* encrypted using SA.key along with its sign value. 



Bob Decrypting and Verifying the Assignemnt.pdf file that Alice Sends him Securely.

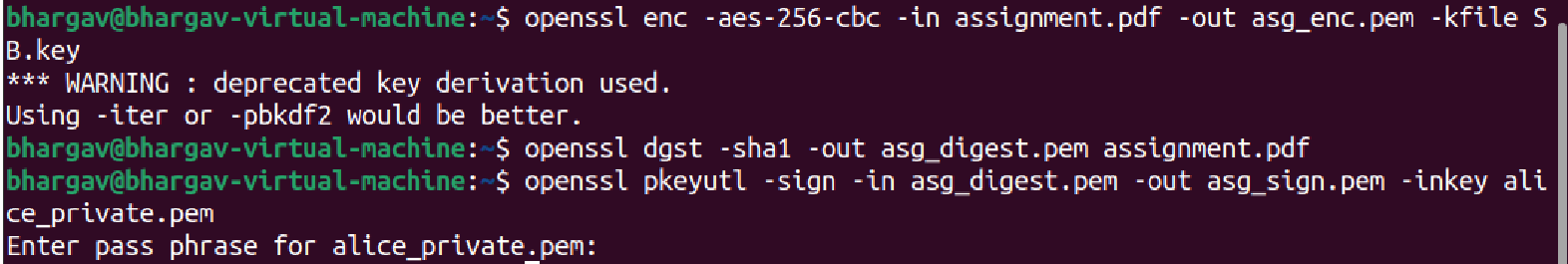




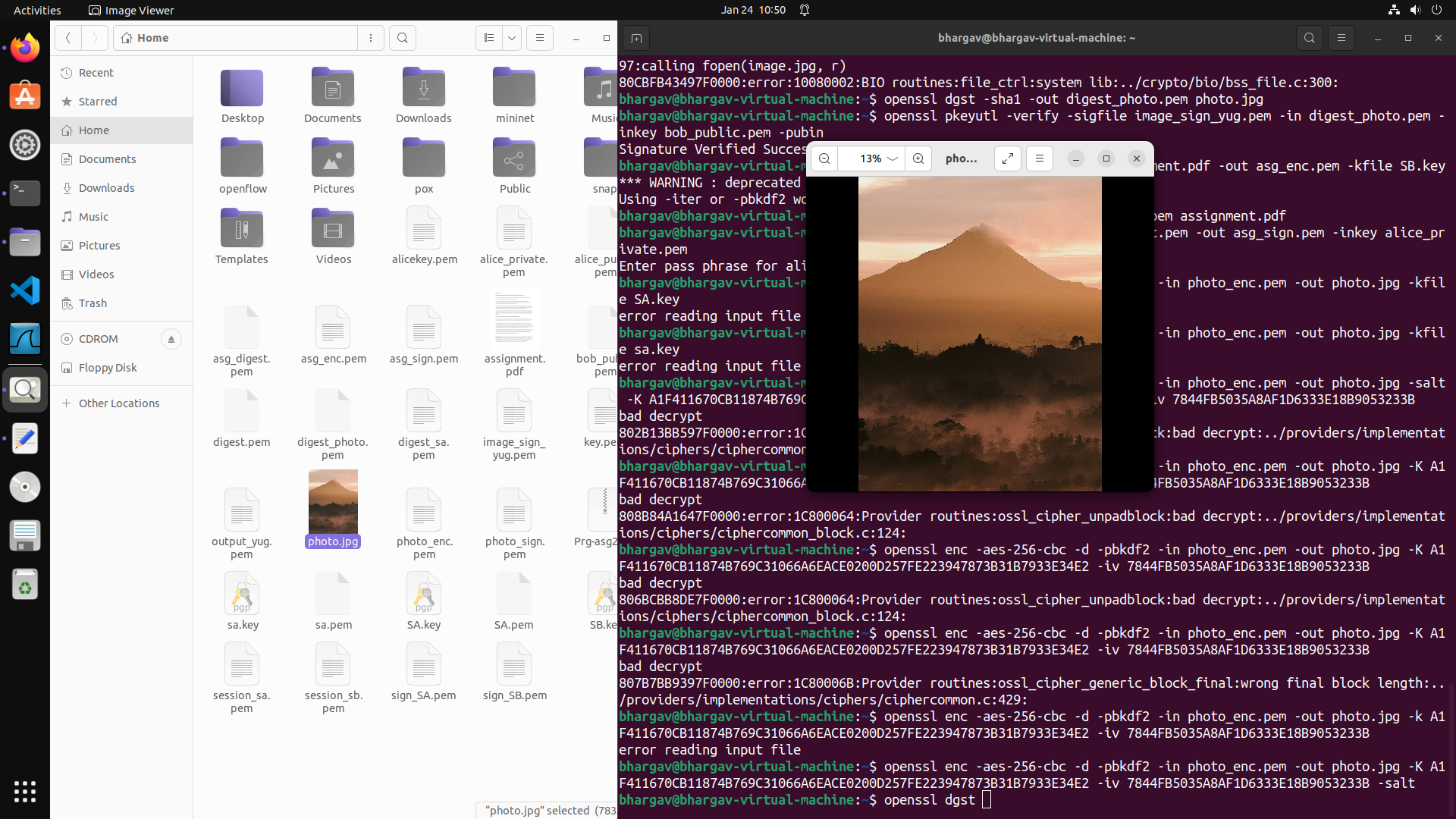
Alice sends *assignment.pdf* encrypted using SB.key along with its sign value.





****

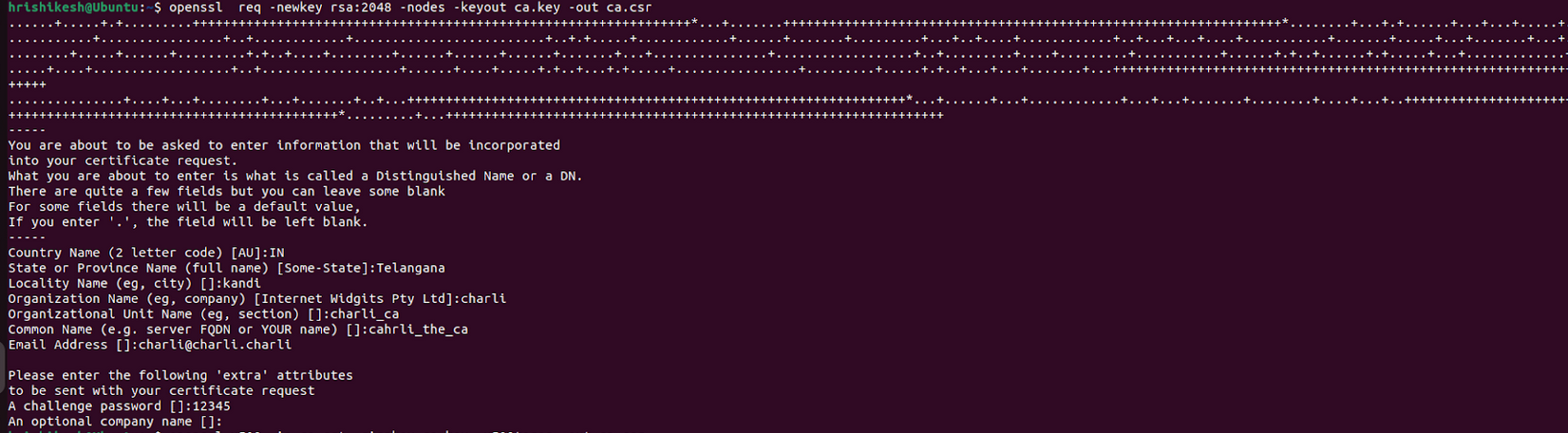
Alice Decrypting and Verifying the photo.jpg file that Bob Sends him Securely.

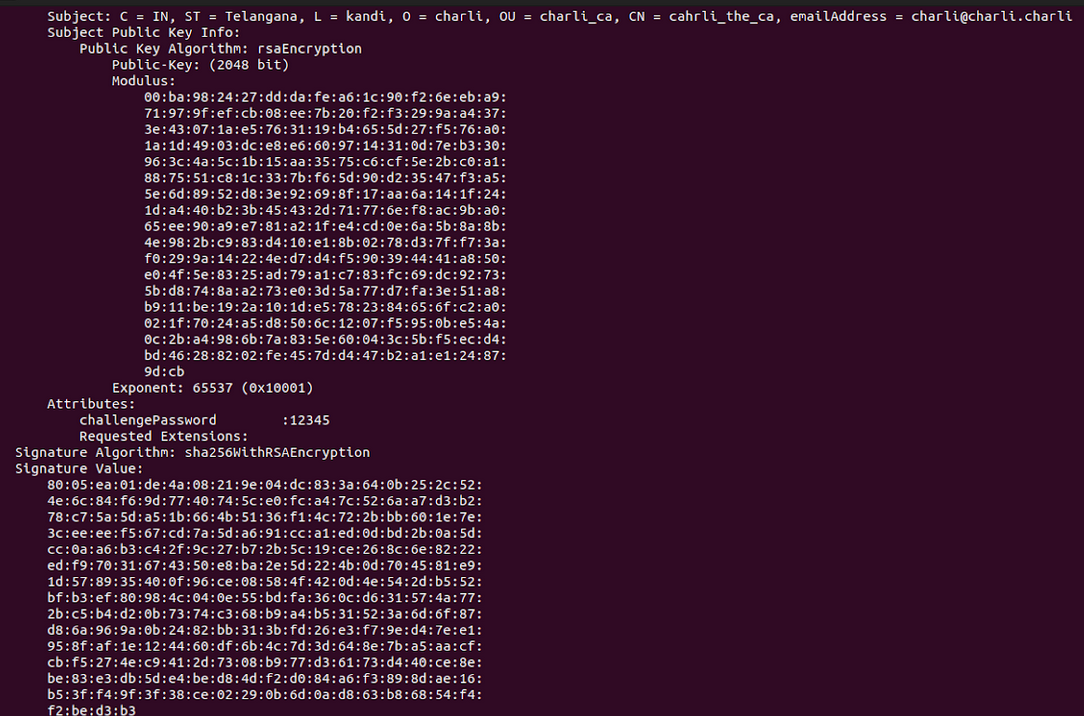


**Part B: Alice (Browser), Bob (web server) and Charlie (Root CA)**

1. Charlie (one of the students who is not part of this group of 2 students) generates a self-signed certificate named charlie-ca.pem or charlie-ca.crt as s/he also plays the role of the root CA.

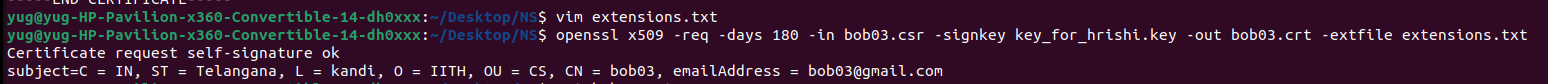
To generate a self signed certificate of root CA, first what I did is generated a csr request along with a certificate authority key i.e ca.key and signed the csr with the ca.key thus generating a self signed certificate. Also CA send its certificate through mail

*Generates csr names ca.csr and key ca.key*



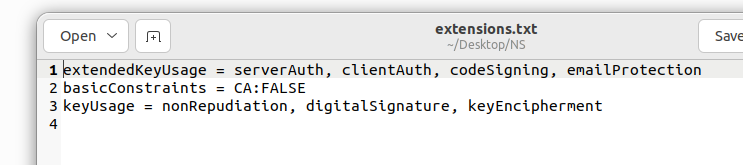
*Generates a self signed certificate*

**2. Bob generates a CSR named bob-domain.csr for getting X.509 V3 certificate and emails it to Charlie for getting the end-user cert named bob-domain.crt issued by the root CA (Charlie). Note that CSR should contain all relevant details of Bob (Student A/B) like Country, Organization, CN/SAN/etc. Bob verifies bob-domain.crt is valid and indeed signed by the root CA, Charlie.**

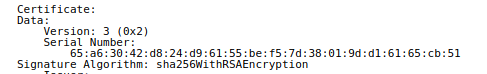
****

*Bob generated csr*

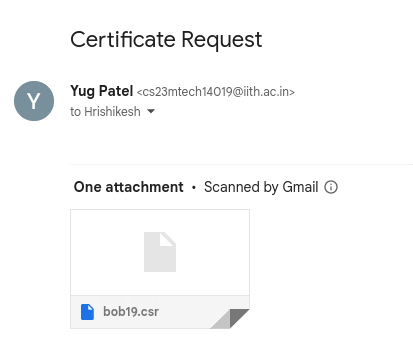


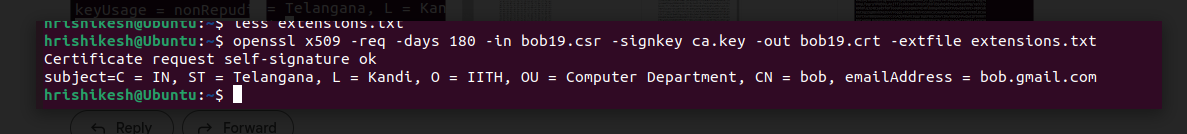


*Version 3 Extensions*



Then this step involves sending the CSR to Root CA over mail for signing



This step demonstrates how the CA signs the certificate using its secret key and bob securely receives that over an email.

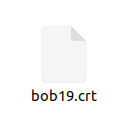
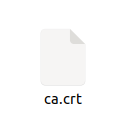
Lastly Bobs verifies the received certificate using the digital certificate of the root CA that he already had.

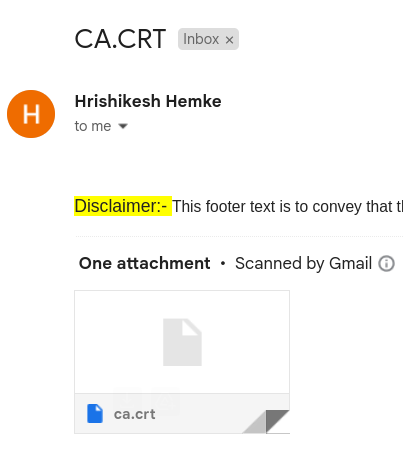
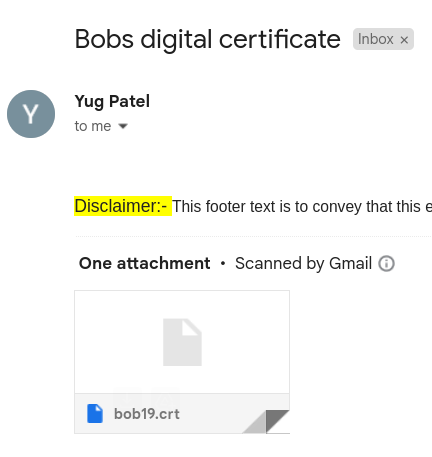




*Verification output*

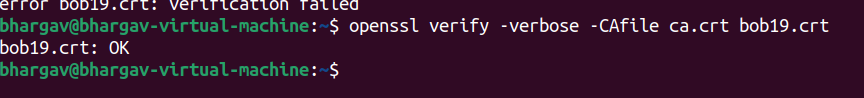
**3. Alice (Student A) gets charlie-ca.crt over email from Charlie and bob-domain.crt over email from Bob and verifies that these certificates are valid and signed by the root CA, Charlie. Comment on whether Bob's cert is of type X.509 V3, what is the serial no assigned, and what are the key usages/constraints associated with the cert.**

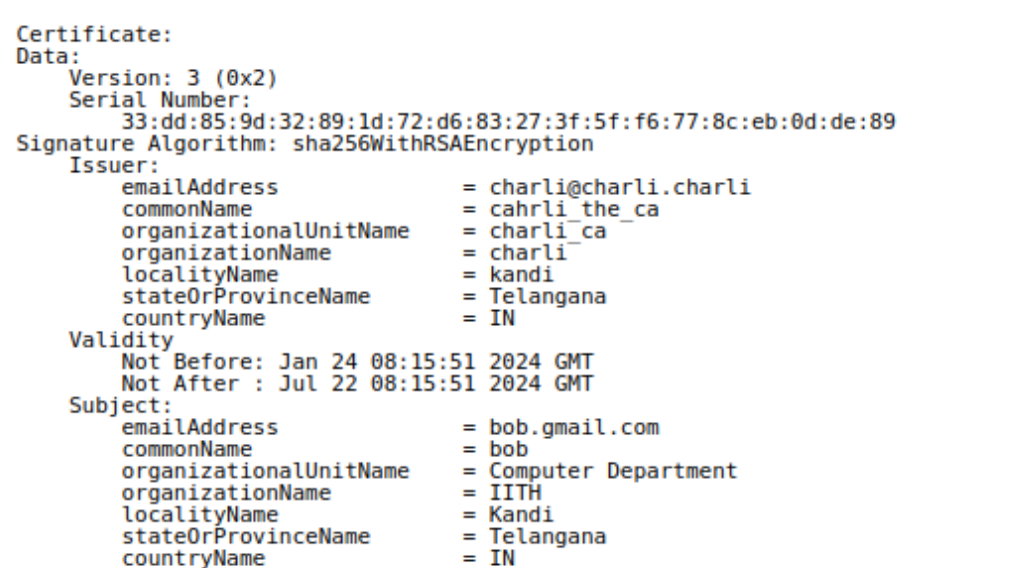




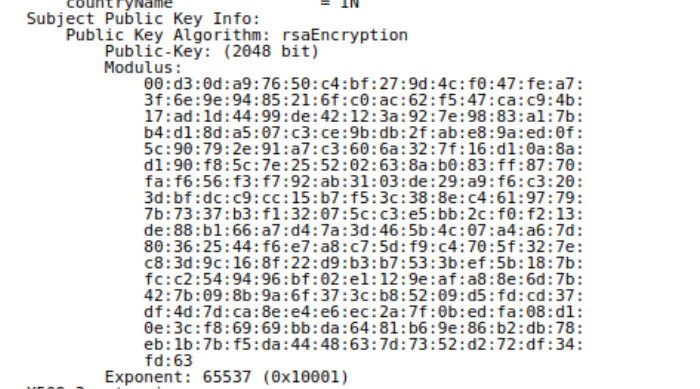


*Alice received certificate from bob Alice received CA certificate*

*Alice verifying the certificate*



*Certificate version, serial number and other details associated with it*



*key usages/constraints associated with the cert*