Cryptography (CS362)

Assignment - 2

**U19CS012**

**Aim**: To demonstrate working of **Diffie Hellman Key Agreement protocol**.

**Library Used**: *OpenSSL*

**To Show**: Same key is shared between two users i.e. User A and User B.

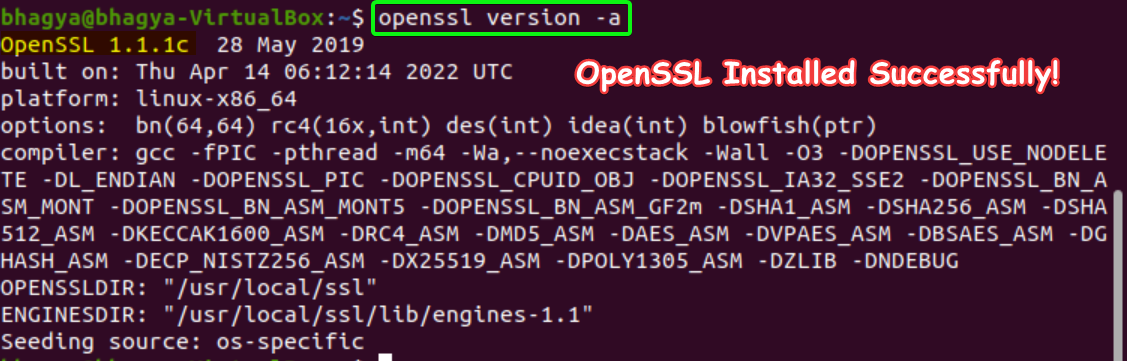
**To Generate Below Mentioned Files**:

1) Global Parameter file

2) Public key Private key files for User A and User B

3) Shared key file for User A and User B

1.) Check if **OpenSSL** is installed in your Linux system or not. If not, go to this link to install OpenSSL in your system.



Reference - [https://fedingo.com/how-to-install-OpenSSL-in-ubuntu/](https://fedingo.com/how-to-install-openssl-in-ubuntu/)

2.) Generate a Diffie-Hellman **Global** **Domain** Parameters and save it in a file DHparam.pem

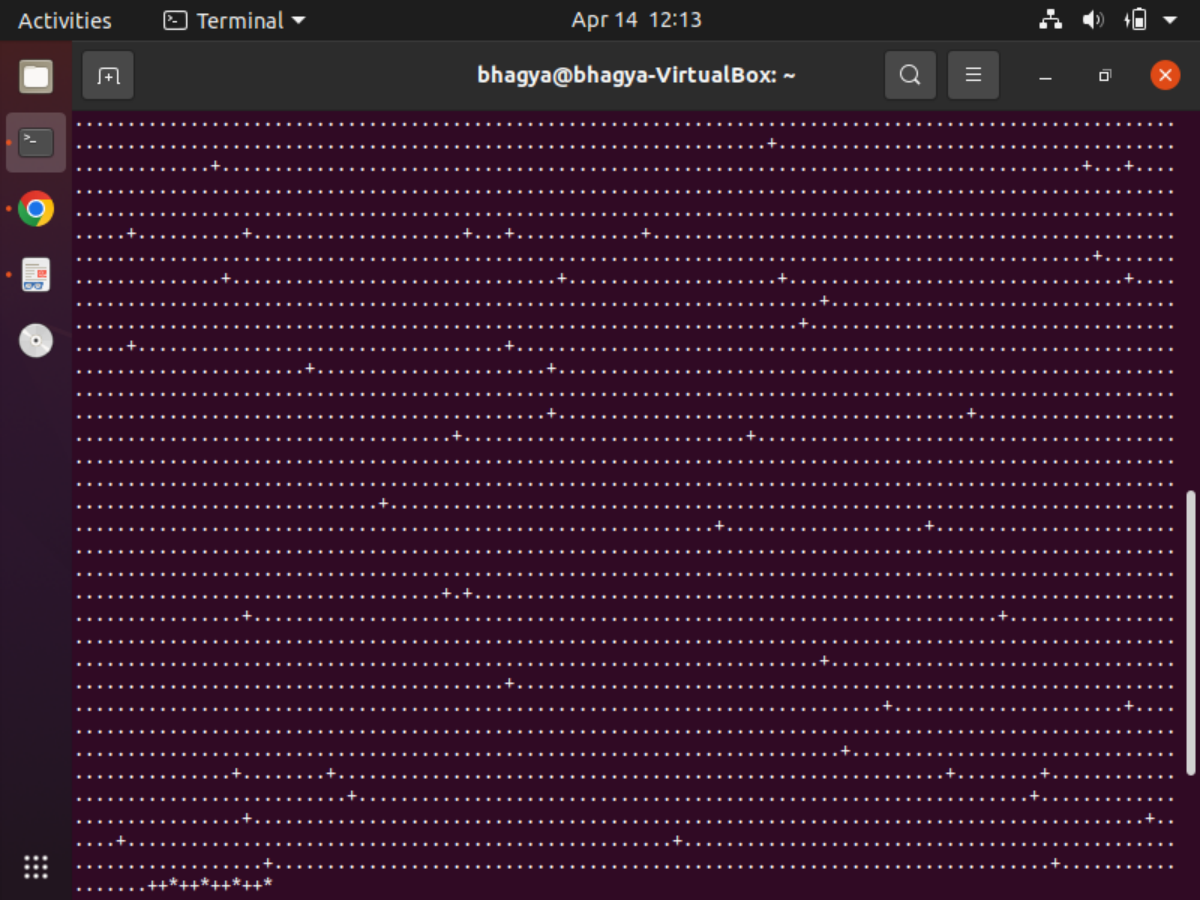
Use the command:

openssl genpkey -genparam -algorithm DH -out DHparam.pem

Meaning –

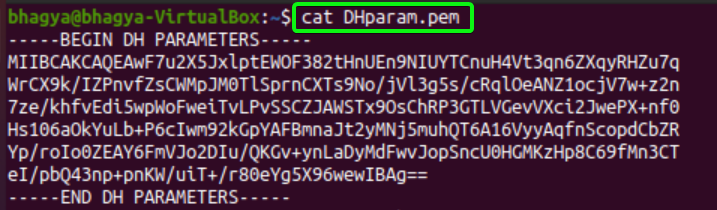
|  |  |
| --- | --- |
| Command | Meaning |
| openssl | OpenSSL command line tool |
| genpkey | Generates a Private key |
| -genparam | Generate a set of parameters instead of a private key. |
| -algorithm DH | Using Diffie Hellman Algorithm |
| -out DHparam.pem | Output saved to DHparam.pem |





3.) Display the generated global public parameters, using the following commands. See the difference between both the commands.

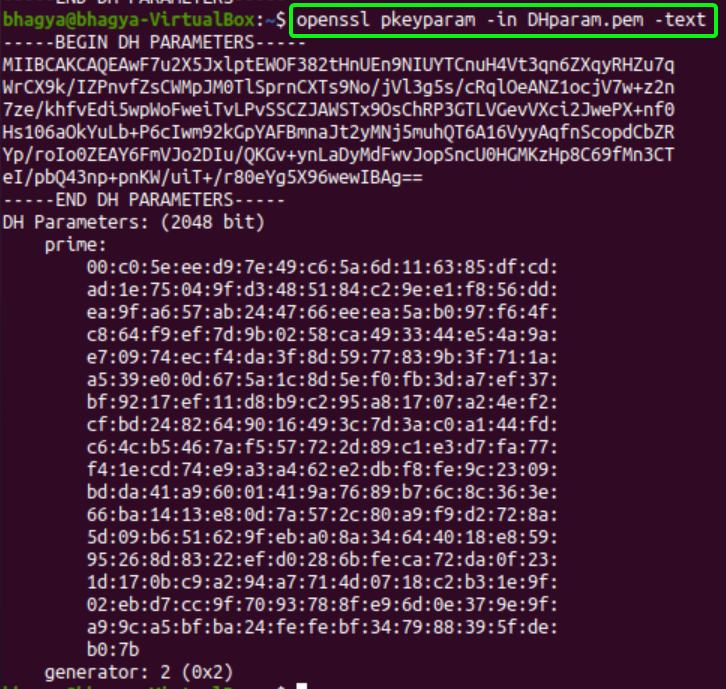
cat DHparam.pem



Print out text version of parameters

openssl pkeyparam -in DHparam.pem –text

|  |  |
| --- | --- |
| Command | Meaning |
| openssl | OpenSSL command line tool |
| pkeyparam | Public key algorithm parameter management |
| -in DHparam.pem | Input File to Read Parameters |
| –text | Print an (unencrypted) text representation of private and public keys and parameters along with the PEM or DER structure. {Certificates} |



We can observe the **Prime** and the **Generator** along with DH Parameters.

4.) The global public parameters generated in above steps can now be used by User A and User B in the protocol to generate their own **Public** and **Private** key.

Save the keys in files DHkeyA.pem and DHkeyB.pem for User A and B respectively.

Use the following **commands** for this step.

**For User A**:

OpenSSL genpkey -paramfile DHparam.pem -out DHkeyA.pem

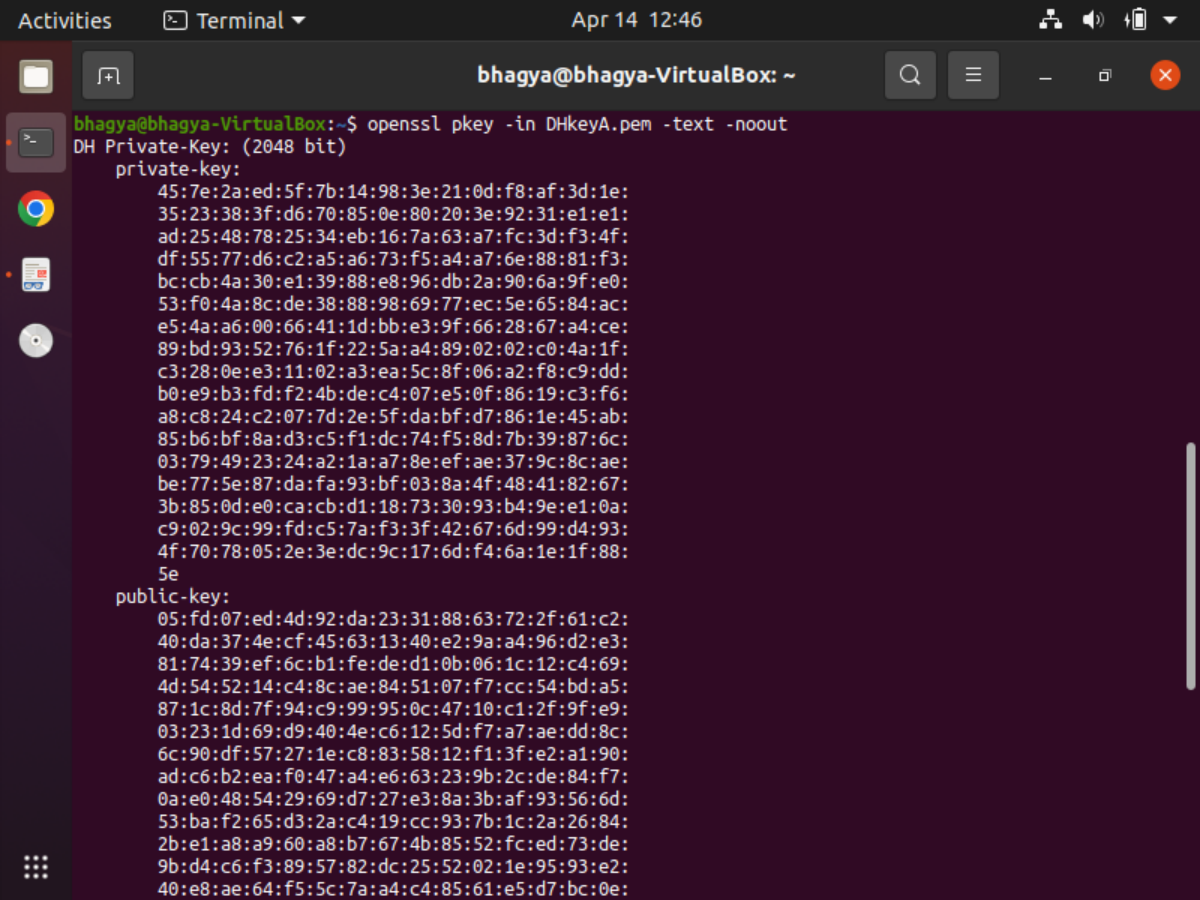
**For User B**:

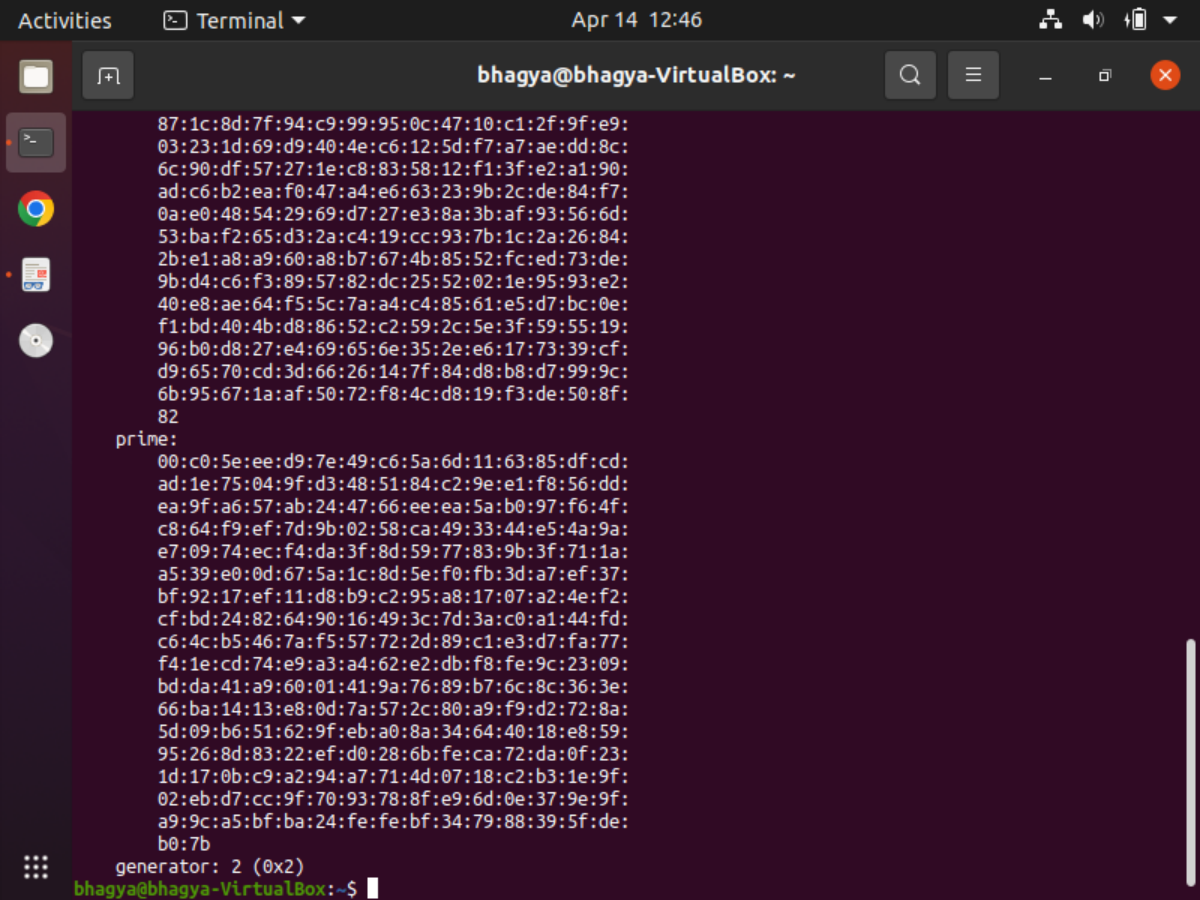
OpenSSL genpkey -paramfile DHparam.pem -out DHkeyB.pem



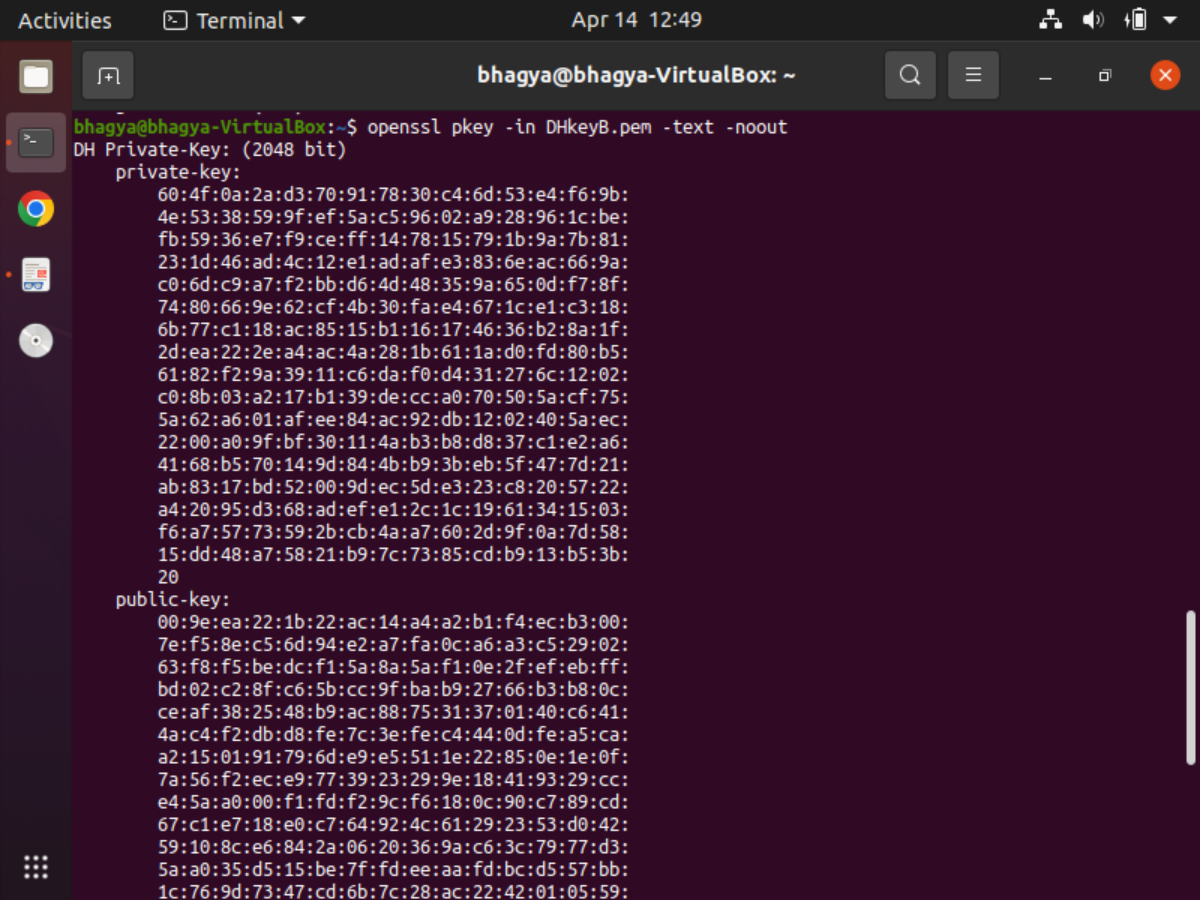
5.) Display the public and private key using following command

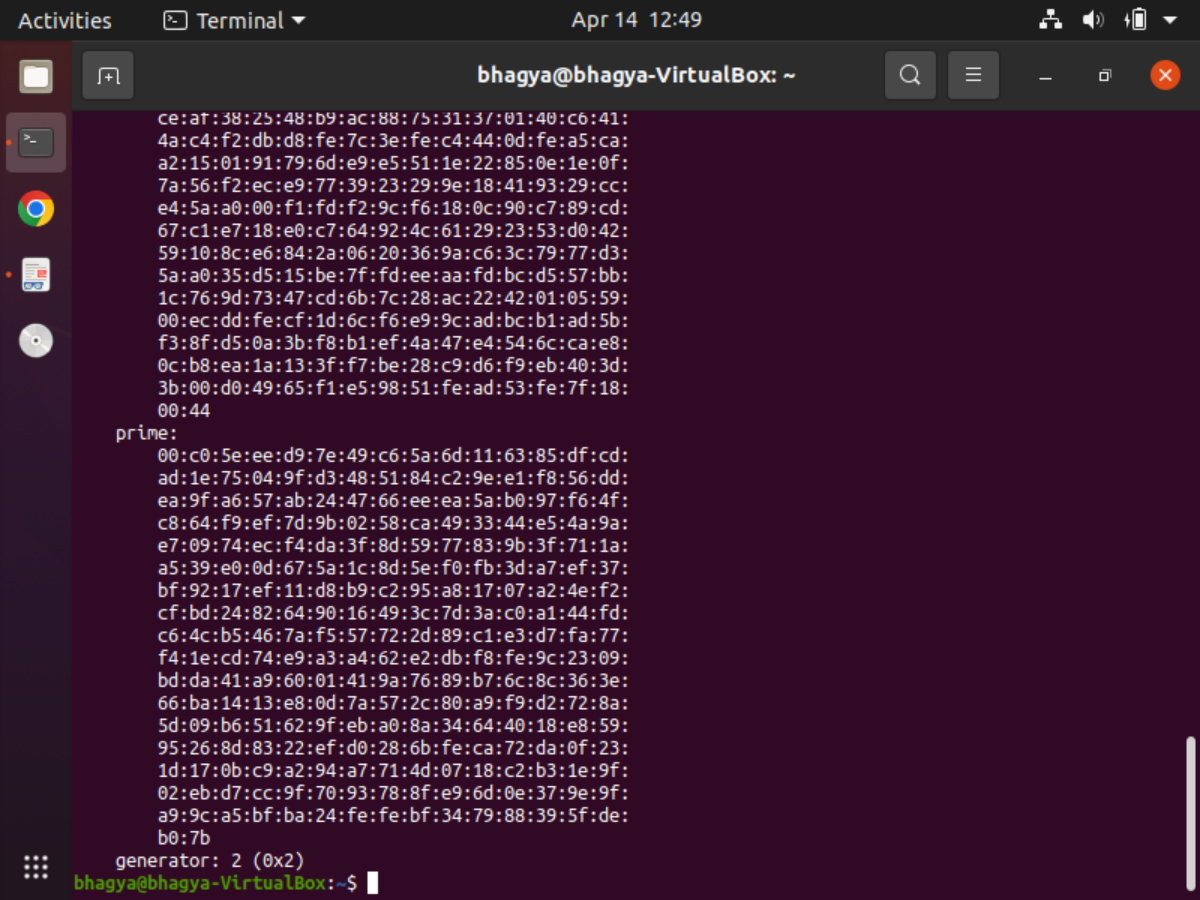
OpenSSL pkey -in DHkeyA.pem -text -noout





OpenSSL pkey -in DHkeyB.pem -text -noout





**Private Key**, **Public Key**, **Prime** and **Generator** can be Clearly Seen for Both A & B.

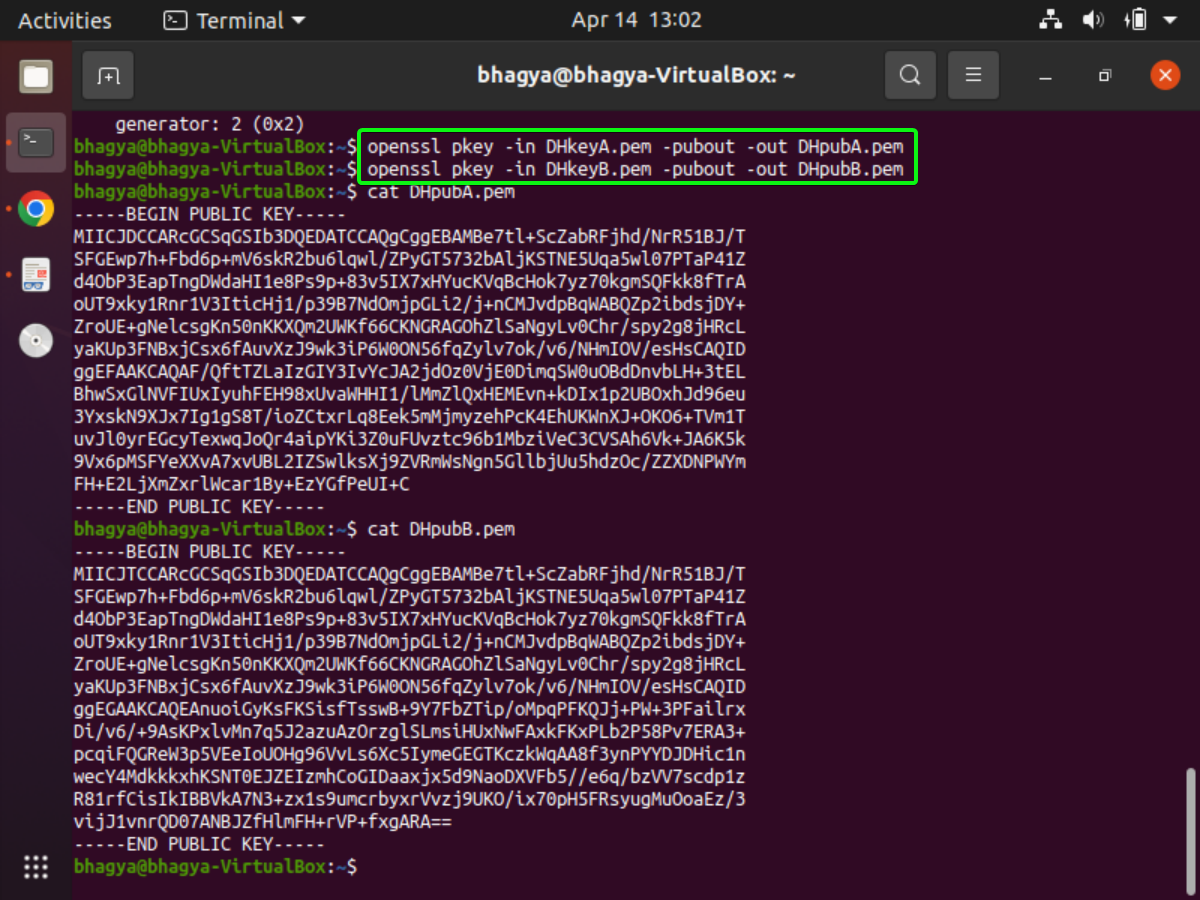
6.) Extract the public keys of user A and user B into separate file viz., DHpubA.pem and DHpubB.pem.

Command to Extract Public Key for A:

openssl pkey -in DHkeyA.pem -pubout -out DHpubA.pem

Command to Extract Public Key for B:

openssl pkey -in DHkeyB.pem -pubout -out DHpubB.pem



7.) Let us consider, both the users have exchanged their public keys with each other. That means, user A has DHpubB.pem and user B has DHpubA.pem.

Using this keys, generate a shared secret key (128 bit binary file) at both sides using following command.

OpenSSL pkeyutl -derive -inkey DHkeyA.pem -peerkey DHpubB.pem -out sharedkeyA.bin

OpenSSL pkeyutl -derive -inkey DHkeyB.pem -peerkey DHpubA.pem -out sharedkeyB.bin

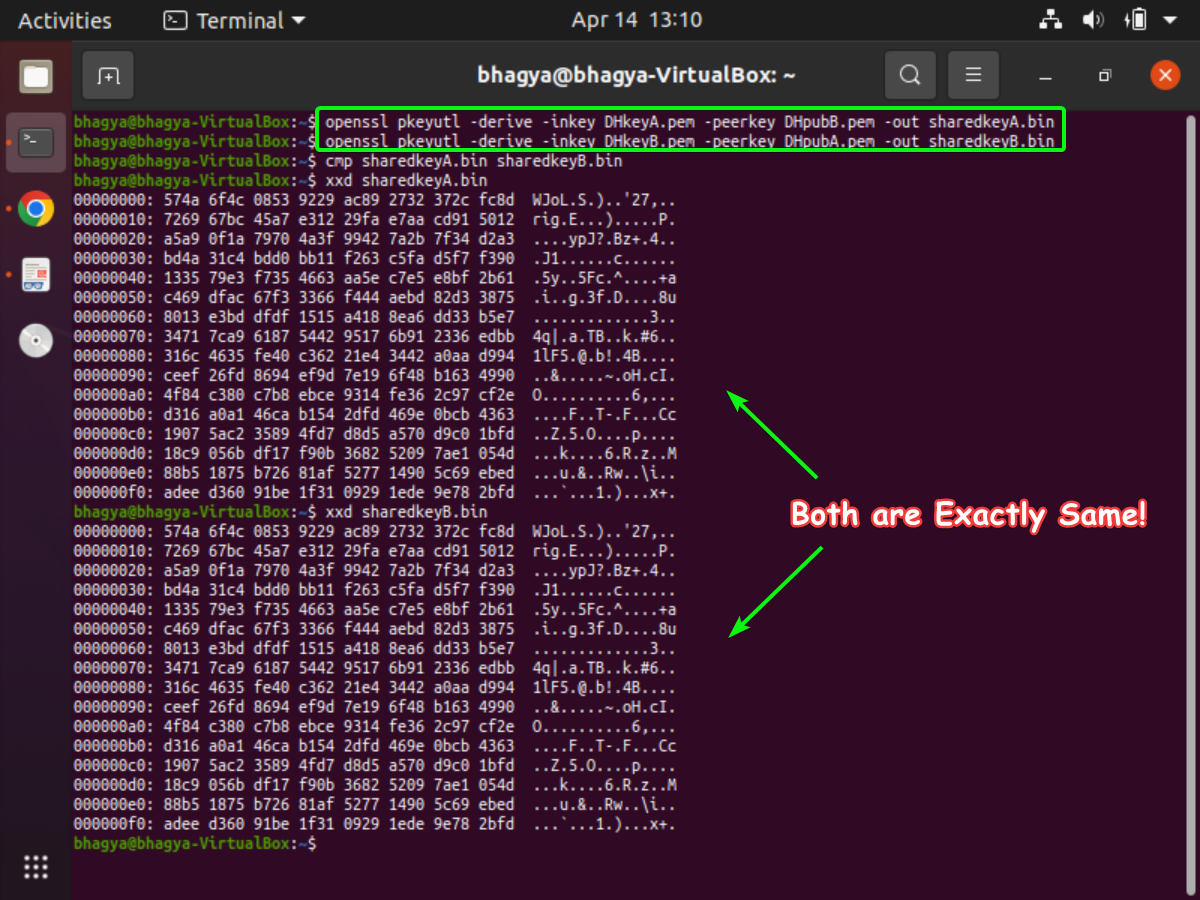


8.) Check if **same** **key** is generated at both sides.

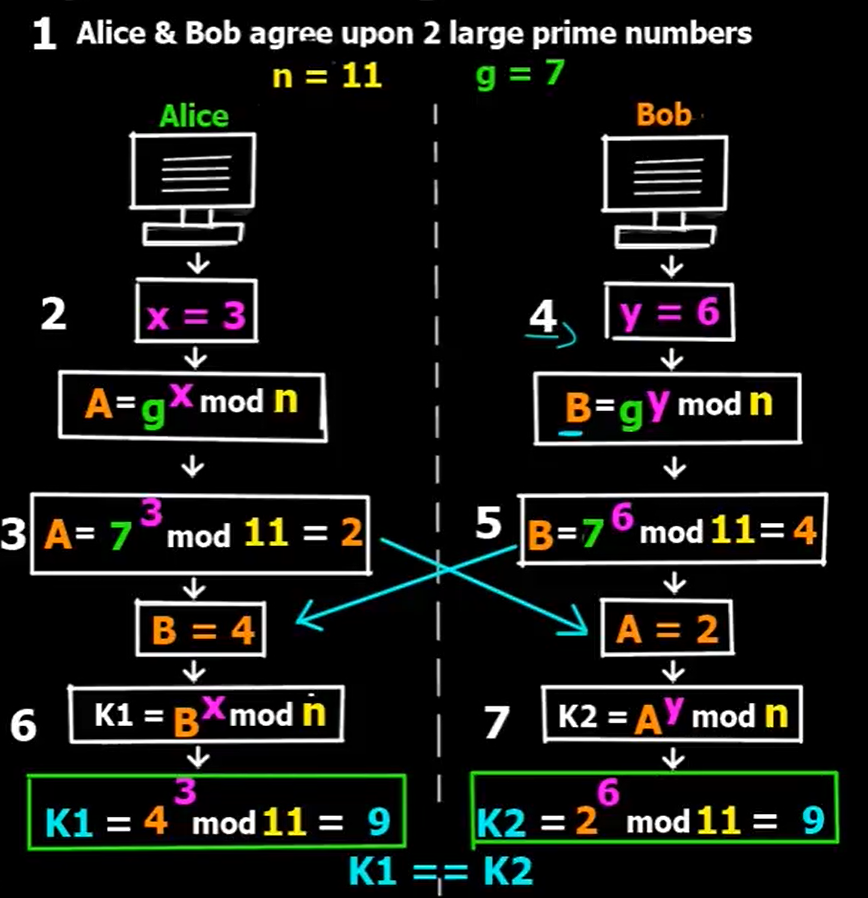
cmp sharedkeyA.bin sharedkeyB.bin

xxd sharedkeyA.bin

xxd sharedkeyB.bin



Both have the **Exact Same Shared Secret Key** (128 Bit Binary File).



Hence, Using **Diffie Hellman Key Protocol**, we have **Successfully Verified** that Same Key is Shared Between Two Users.

**SUBMITTED BY**: U19CS012

BHAGYA VINOD RANA