ST. XAVIER'S COLLEGE

**(Affiliated to Tribhuvan University)**

**Maitighar, Kathmandu**



**PROJECT REPORT**

**ON**

**Secured Sharing of Data Over Unsecure Channel by Using CUI of Operating System**

**Under the supervision of**

**Er. Anil Kumar Sah**

**Sr. Lecturer**

**Submitted By:**

**Priya Sri**

**Submitted To:**

ST. XAVIER'S COLLEGE

**Department of Computer Science**

**Maitighar, Kathmandu, Nepal**

**Summer 2019**

**Abstract**

CIA and non-repudiation are the most primary need for any enterprise network who wish to keep the information transmission secure. For any enterprise, networking is a must which is obviously needed for the day-to-day transactions. In this age, where Globalization is at the highest level of its entire history and also in the history of mankind, networking is thus important. With success, difficulties, opponents and threats come. This Security and Preparedness in needed.

Network and Security go hand on hand. While there are many facility services for a network with its own WAN, LANs, like VPN and so on, this project is a small effort on:

* Secured sharing of file between different Enterprises, organizations {small-scale}.
* Allowing an enterprise/company or even individual to do secured transactions from and to a place where it has none of its branch i.e. out of its own network , and that also just by collaboration.
* Simple command line interface usages to generate encryption/decryption key management and integrity techniques.
* Also, verifying authenticity of the sender.
* Simple conceptual procedures.
* Can be further extended in future to cover on more aspects of Security.
* Adds double –layer security.
* Guarantees security over a highly unsecured medium, the internet worldwide (wired/wireless).
* Practical and real-life usages in much inter-enterprise communication.

**PROBLEM STATEMENT**

Secure file sharing between different networks can be difficult and highly vulnerable. Especially when it’s not in the same WAN.

Even sharing of information in inter-enterprise communication over a secured medium like web with HTTPS protocol cannot be sometimes trustworthy and reliable, especially when the information is highly sensitive.

This is a small step on sharing of information based on the above scenario. Also, in this project best practices are to be considered so as to full real-life implementation of cryptography techniques and algorithm.

**PROJECT OBJECTIVES**

Cryptography techniques implementation right by the end-user from the end-system just by command-line interface of operating system.

Secure transmission of information over the unsecured medium like Internet.

Add double layer security.

Inter-enterprise data exchange with guaranteed security over any file sharing medium which may or may not have end-to-end encryption facilities.

Close relation to real world techniques as far as possible and practical.

Summary implementation of complete SSL package.

**METHODOLOGY**

This full project will be initially constructed using a rough draft with paper and pencil. Once the format and step is determined we will move towards the next step. Also, platform to be used are:

* Linux (since it’s CUI is more user-friendly, being open-source, etc.).
* Any mailing or file-sharing platform (e-mail, Whatsapp, Viber, etc.).

Some other algorithms and techniques to be used are:

* RSA (for key management/ key length: 2048 bits).
* Hashing Algorithm: sha256.
* Hashing based Message Authentication Code (HMAC)
* AES (Symmetric Algorithm, for actual sharing of encrypted data files).
* CBC

Others:

* Private key and public key of receiver in .pem format.
* 256-bit random number.
* Tool : Openssl

**SAMPLE SCENARIO:**

A money-transfer company in Kathmandu wants to do its daily transaction from Rajbiraj as well. Currently, it’s not able to own a branch of its own at Rajbiraj due to lack of resource. Now, the company plans to collaborate with any IT shop or even any another company who could handle all of its transactions from Rajbiraj and after every business hour give the company at Kathmandu its updates.

The company is concern about the secure transmission of information as its mishandling can result in moderate to high loss and even can create conflicts and hamper the trust.

**CONSIDERATION AND FRAMEWORK:**

The Kathmandu-based company be the receiver, for ease.

The one at Rajbiraj needs to send transaction updates, i.e. it be the sender.

**CONCEPTUAL PROCEDURES:**

1. The receiver sends the public key of its server using any file-sharing platform.

(In this case, we generate private key and public key using CUI of Linux)

1. Sender gets the public key.
2. Sender generate a file to be encrypted.
3. Sender generate a random number (secret key of 256-bit for encrypting file using aes-256.
4. Sender encrypt the file using the the secret key.
5. Sender encrypt the secret key using public key of receiver by RSA.
6. Sender generate HMAC using the file and secret key by SHA-1
7. Sender sends the encrypted file, encrypted secret key and HMAC to the receiver.
8. Receiver first decrypt the encrypted secret key using its private key.
9. Using the secret key, receiver now decrypt the encrypted file by aes-256.
10. Receiver generate HMAC using the file and secret key by SHA-1.
11. Receiver now compare the HMAC it has created with the HMAC from the sender.

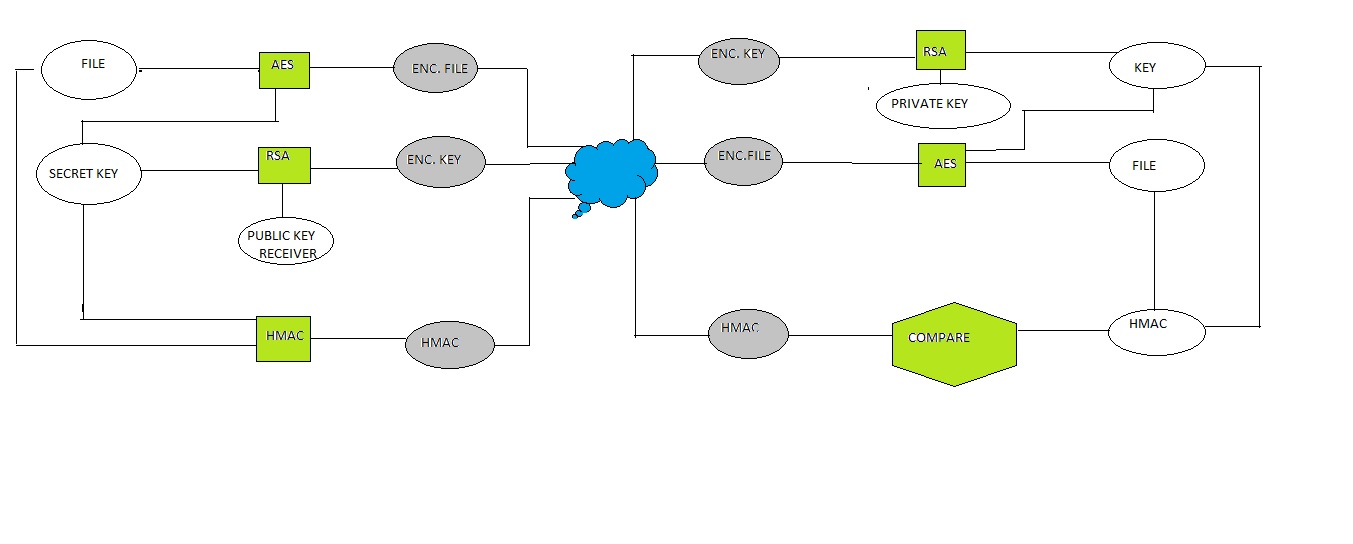


FIG. : MODEL DIAGRAM

**NOMENCLATURE:**

Receiver:

Folder name : receiver

Private key : RecPrivate.key

Public key : RecPublic.key

Sender:

Folder name : sender

Secret key : secret.key

File : file.txt

Encrypted file : file.txt.enc

Encrypted key : secret.key.enc

HMAC: file.txt.hmac.sender

Receiver:

Decrpted key : secret.key

Decrypted file : file.txt

HMAC: file.txt.hmac.receiver

**COMMANDS:**

To make folder:

mkdir receiver

mkdir sender

To generate private key:

openssl genrsa –out RecPrivate.key 2048

To generate public key using private key:

openssl rsa –in RecPrivate.key –out RecPublic.key –outform pem –pubout

To generate 256 bit random number:

openssl rand –hex 64 –out secret.key

To generate file:

vi file.txt

To encrypt file:

openssl enc –aes-256-cbc –salt –in file.txt –out file.txt.enc –pass file:secret.key

To encrypt secret key:

openssl rsautl –encrypt –inkey RecPublic.key –pubin –in secret.key –out secret.key.enc

HMAC:

openssl dgst –sha256 –hmac secret.key –hex file.txt >> file.txt.hmac.sender

To decrypt secret key:

openssl rsautl –decrypt –inkey RecPrivate key –in secret.key.enc –out secret.key

To decrypt file:

openssl enc –d –aes-256-cbc –in file.txt.enc –out file.txt pass file:secret.key

HMAC:

openssl dgst –sha256 –hmac secret.key –hex file.txt.hmac.receiver

HMAC Comparison:

diff –q file.txt.hmac.receiver file.txt.hmac.sender

To view file content:

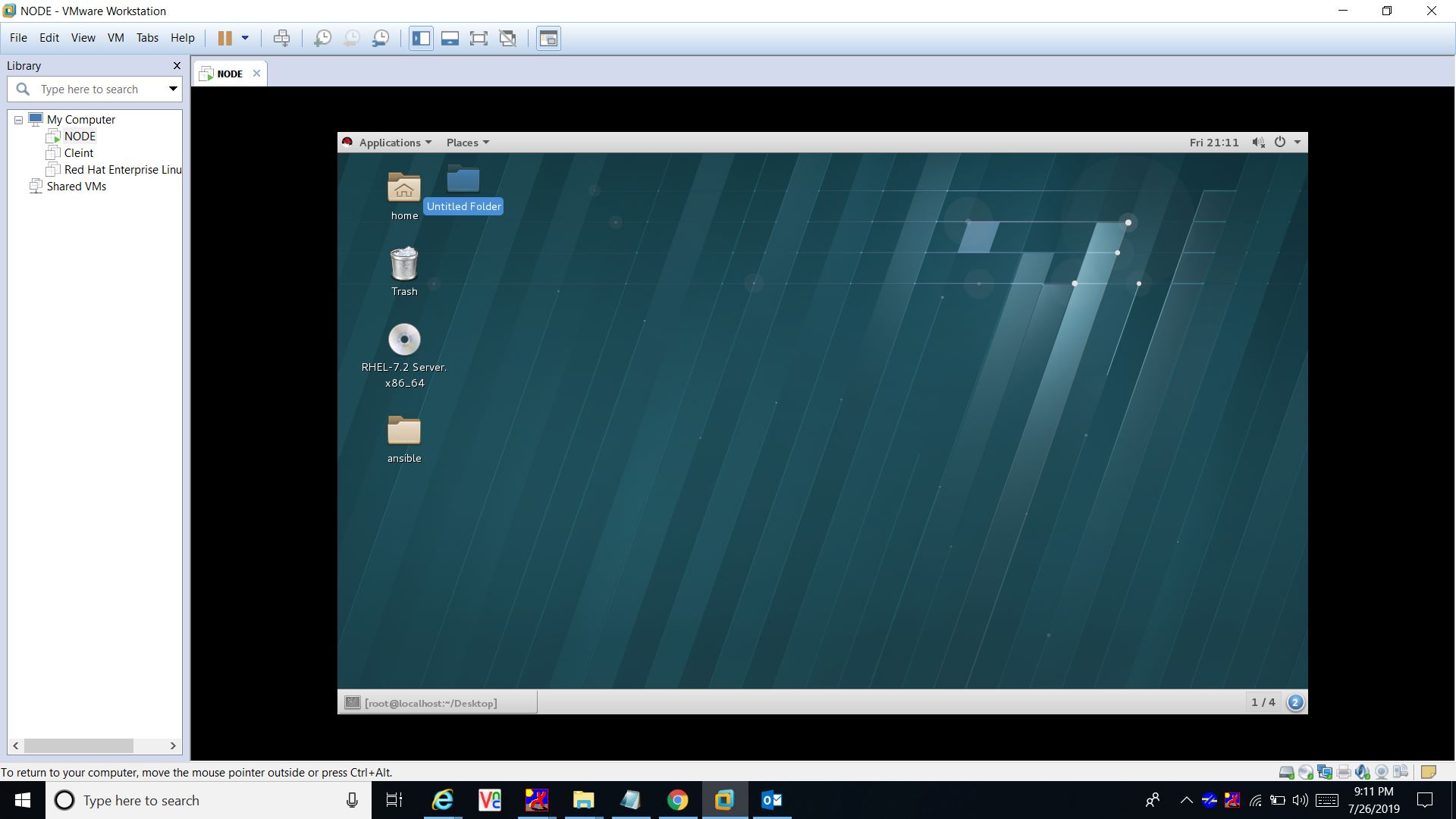
cat filename

To view all files in a folder:

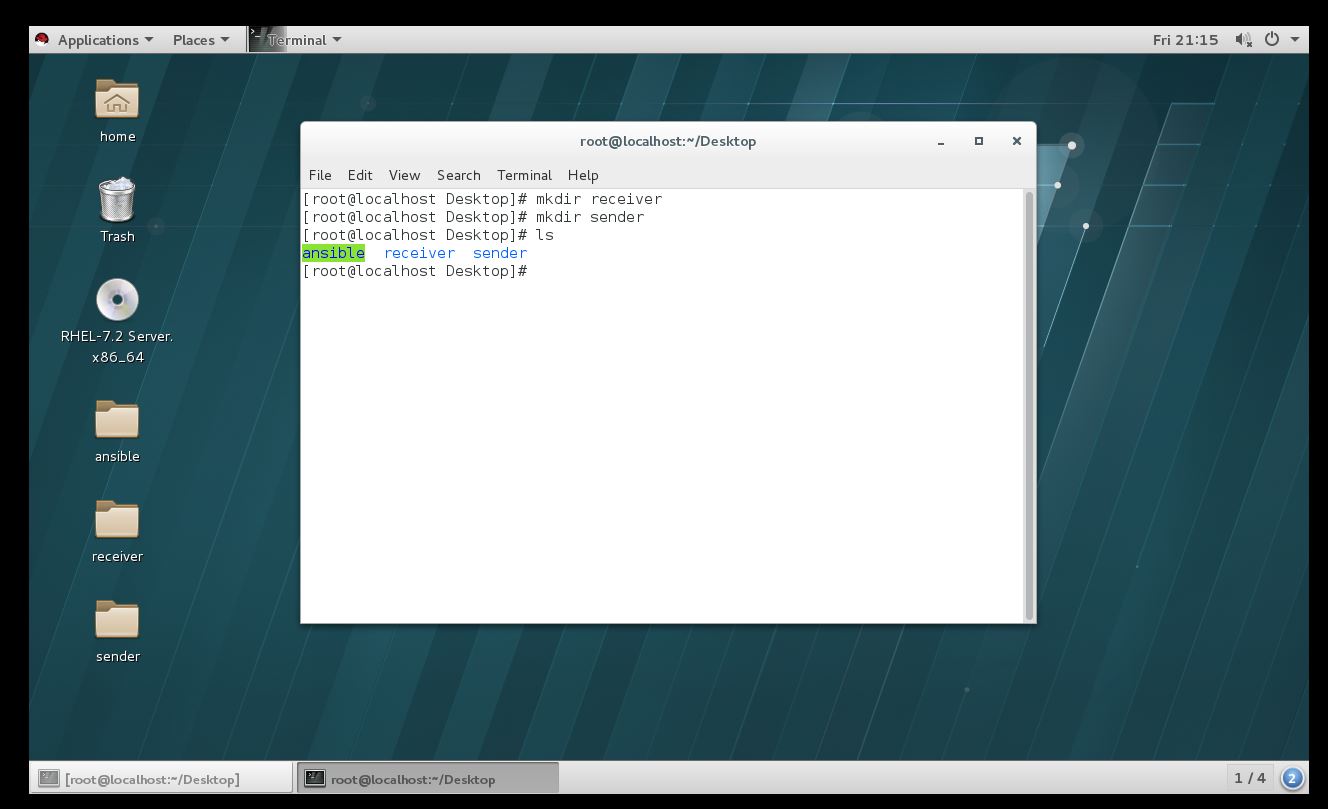
ls

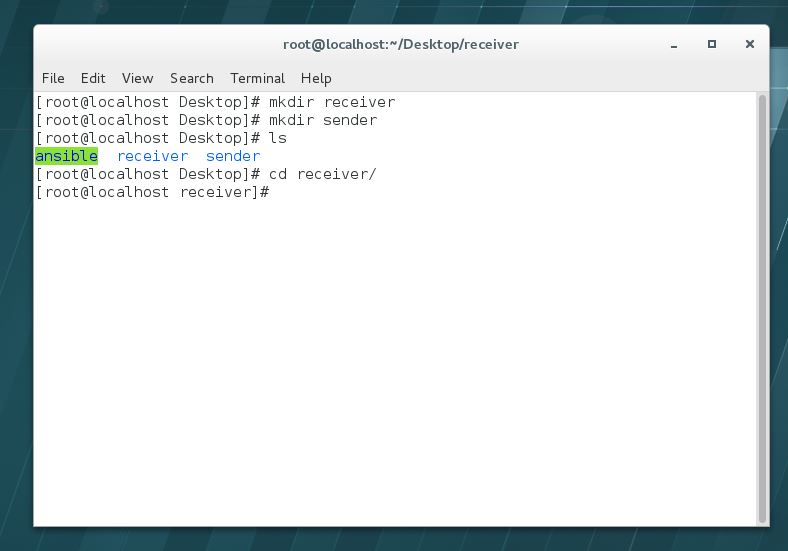
**SCREENSHOTS:**

Linux OS:

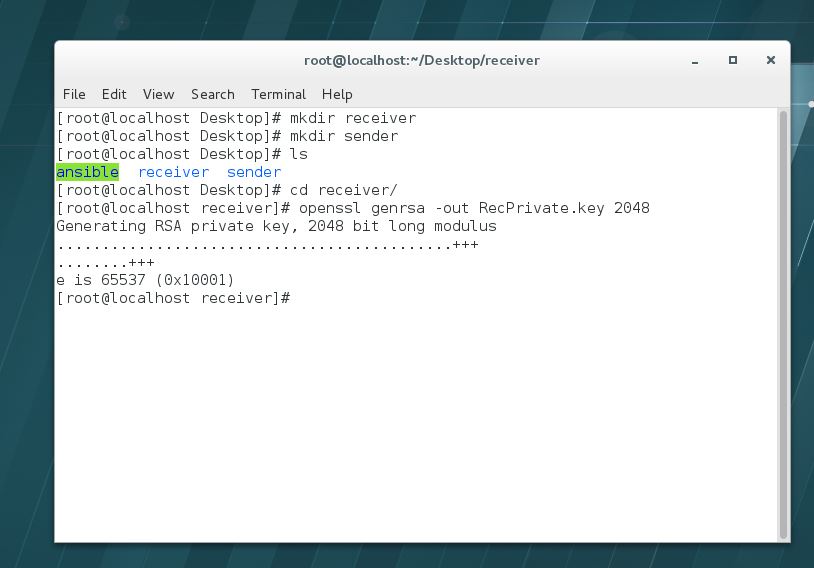


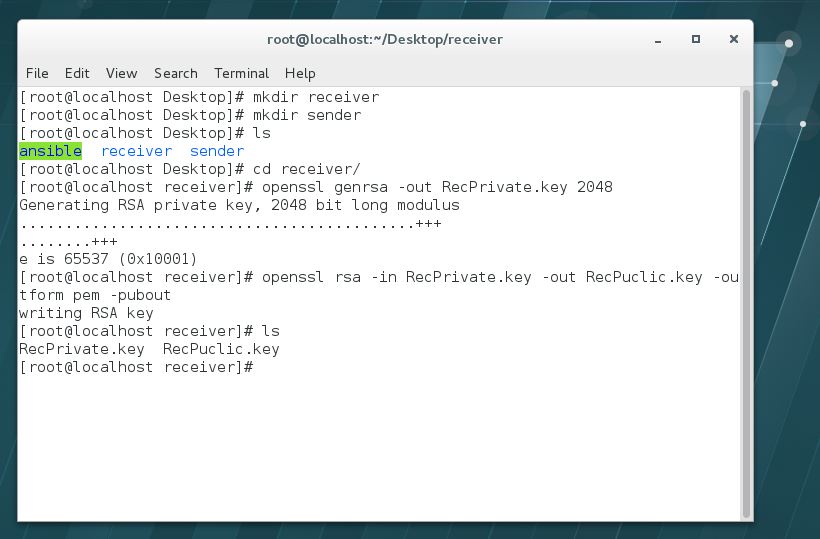
Sender and receiver:

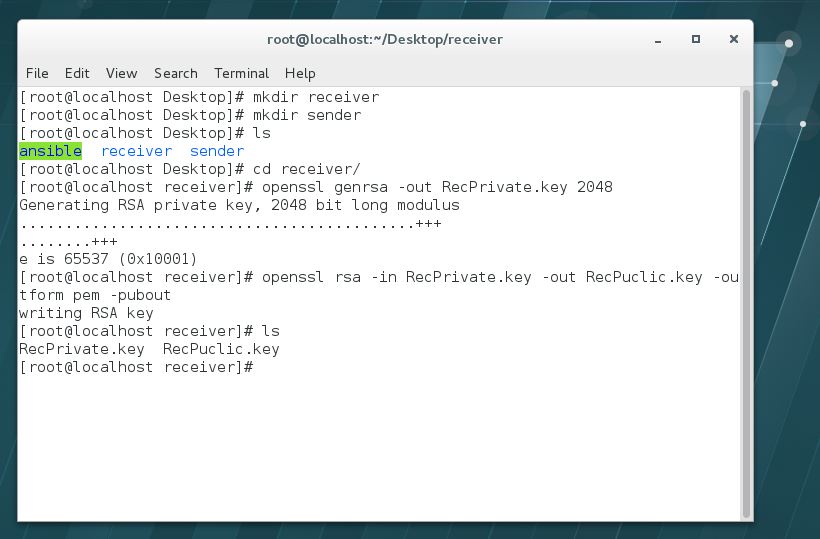




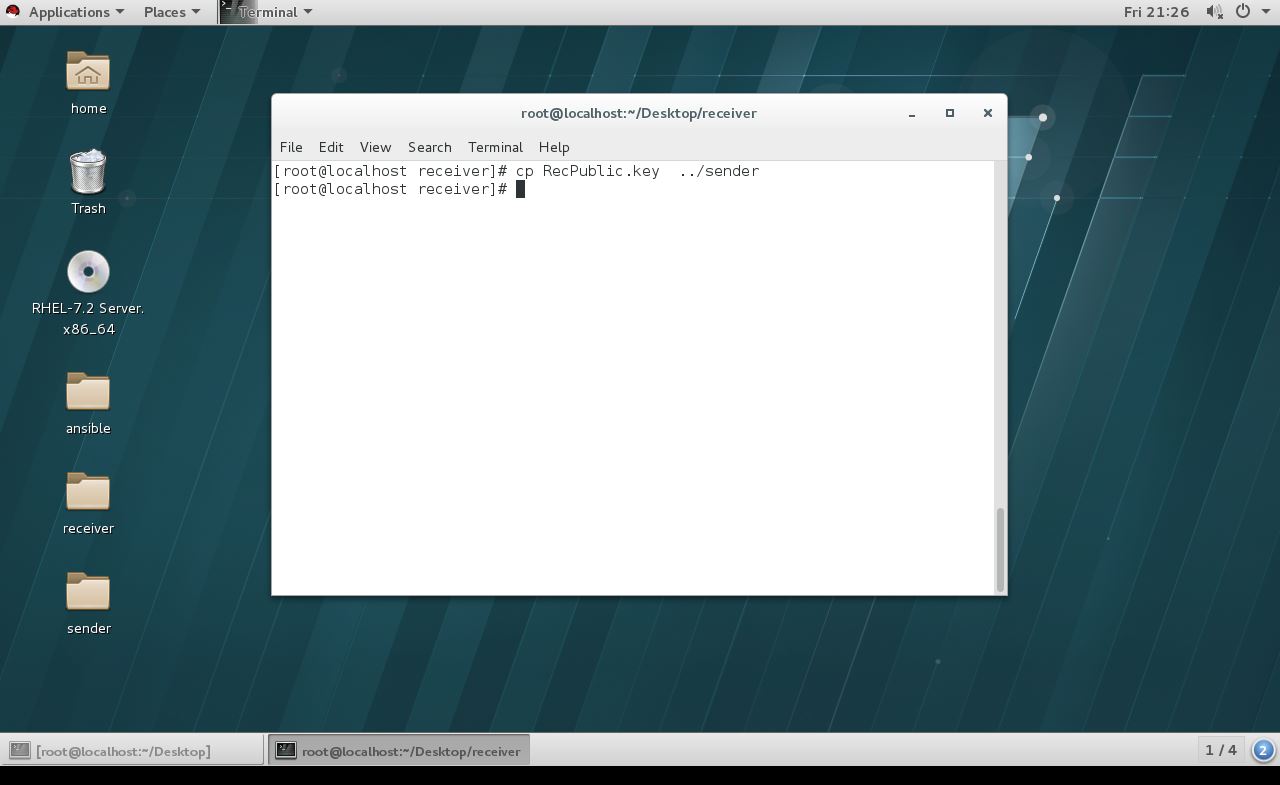
Private key generation:



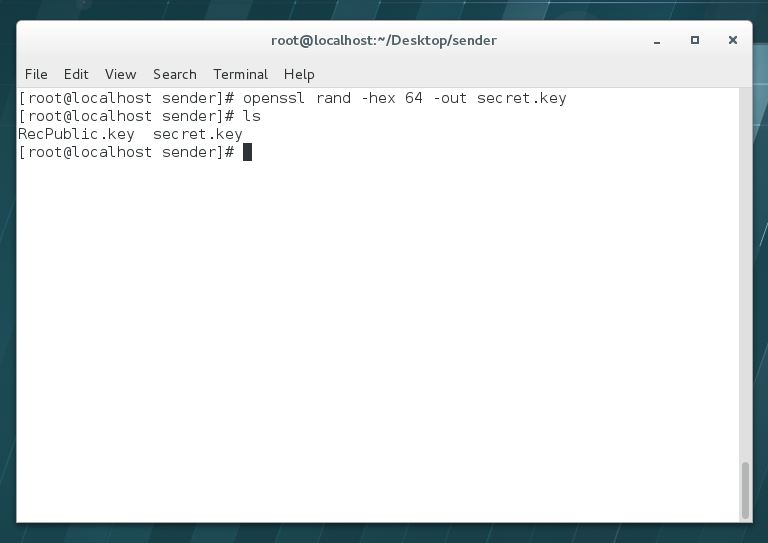
**Public key generation** ****



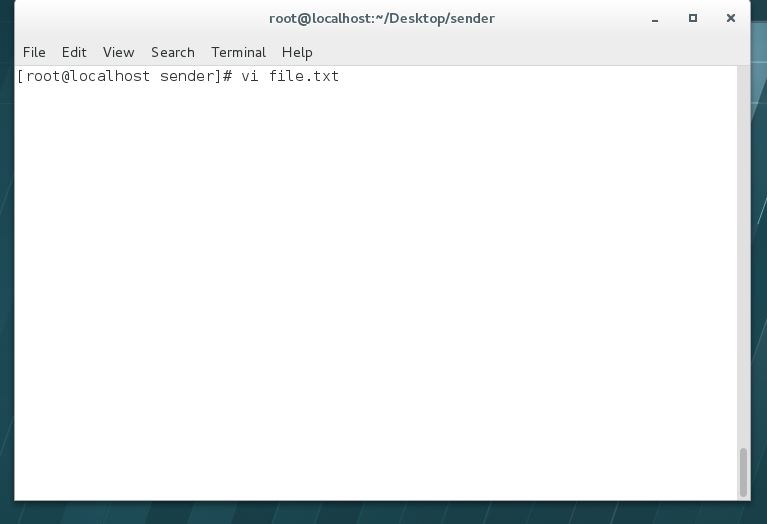
Sending Public key to sender:

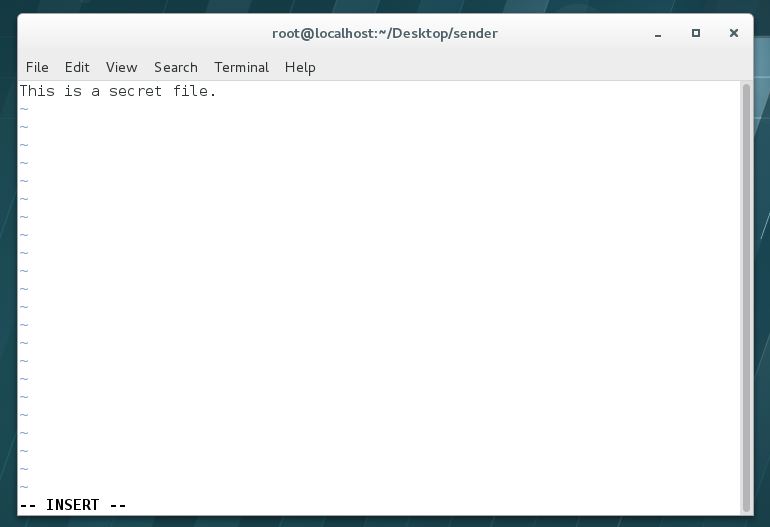


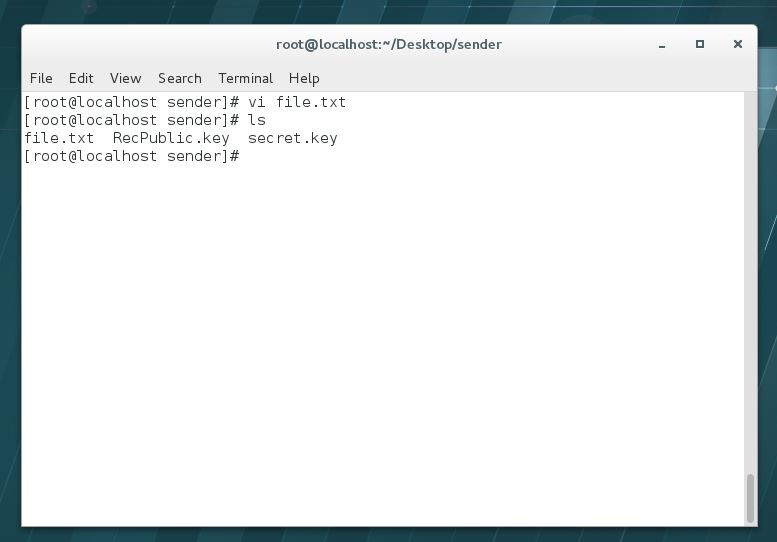
Random Number generation



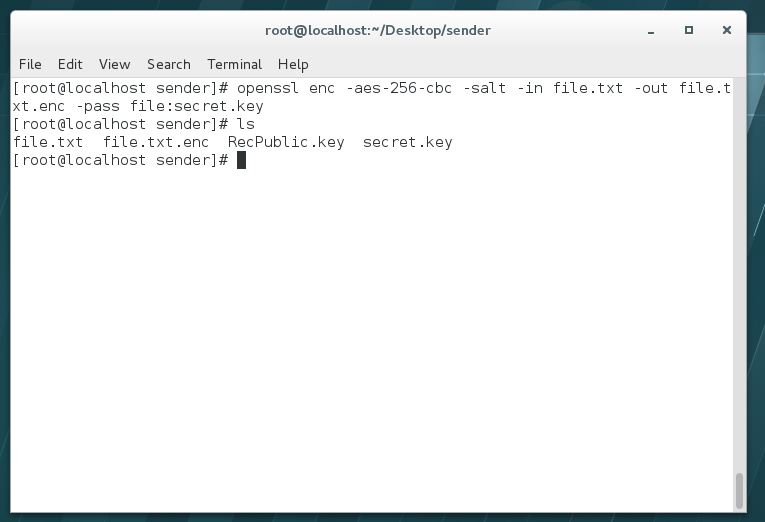
File generation:

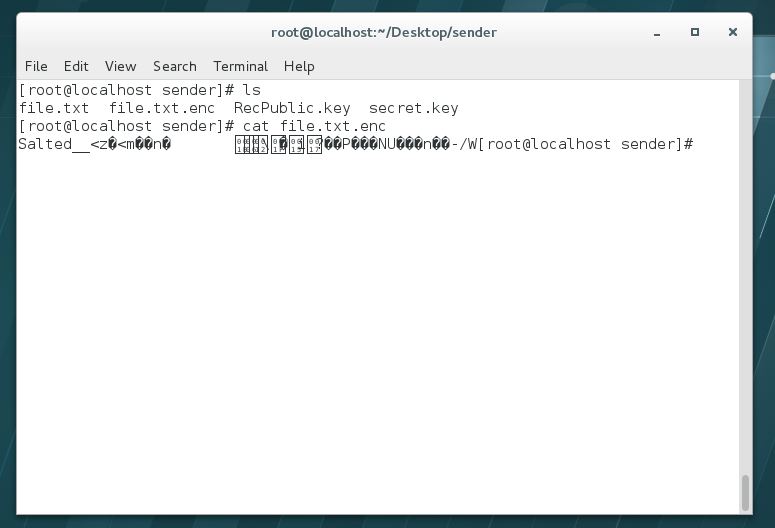




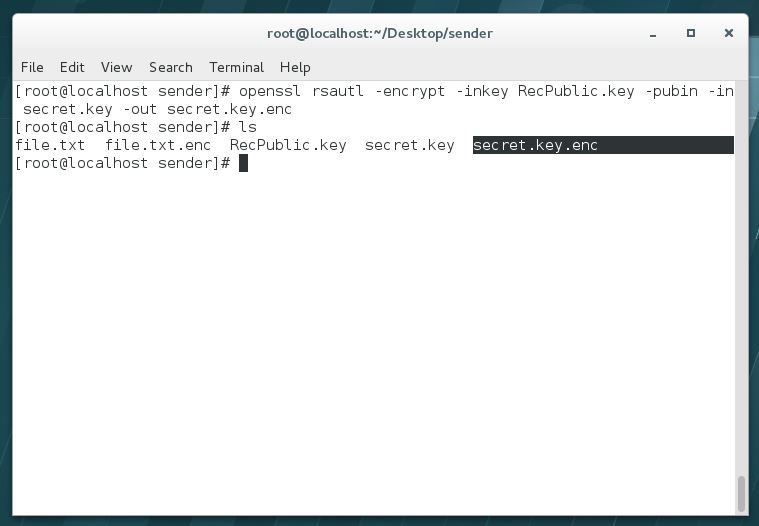


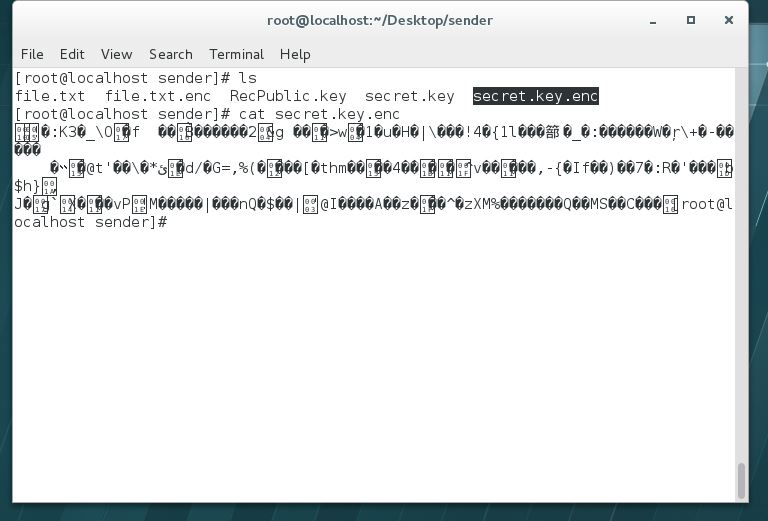
Encrypting file:



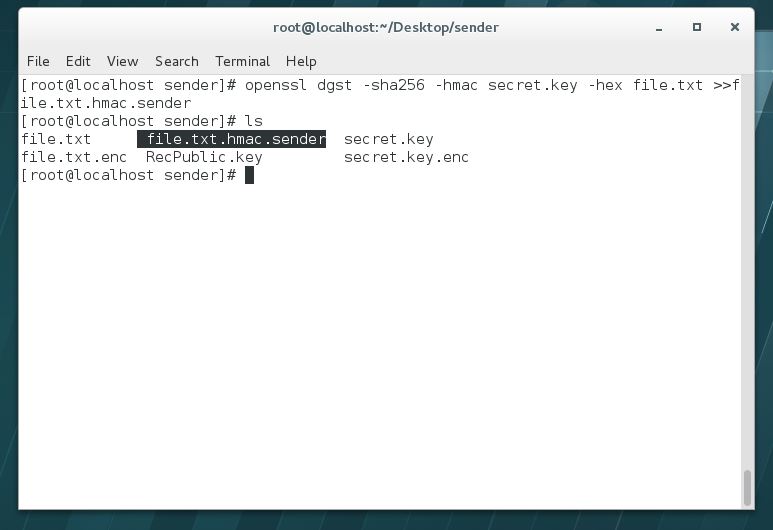


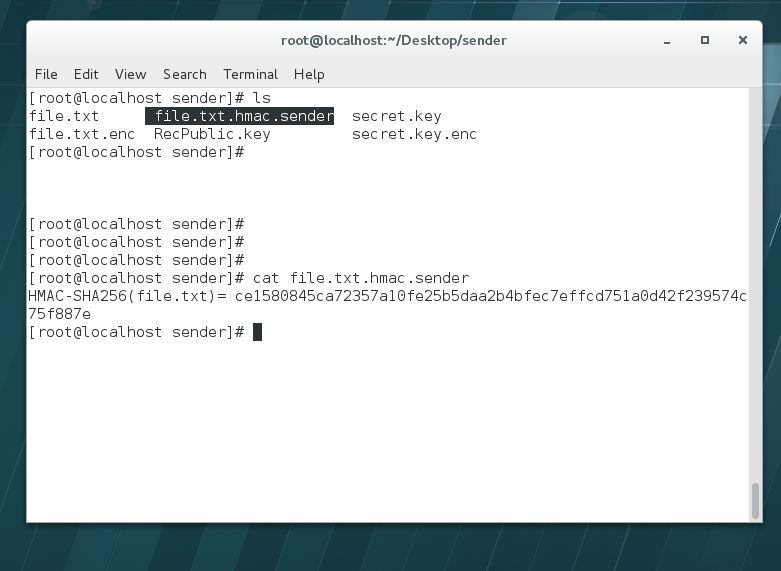
Encrypting Key:



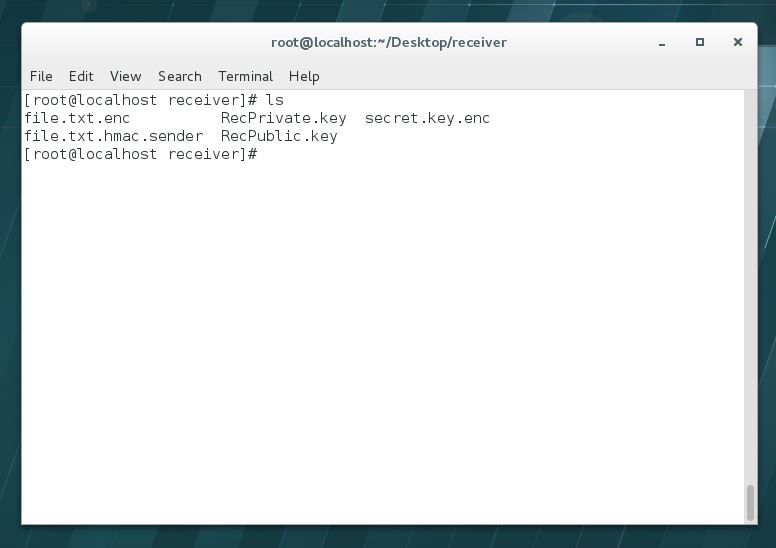


HMAC at sender:

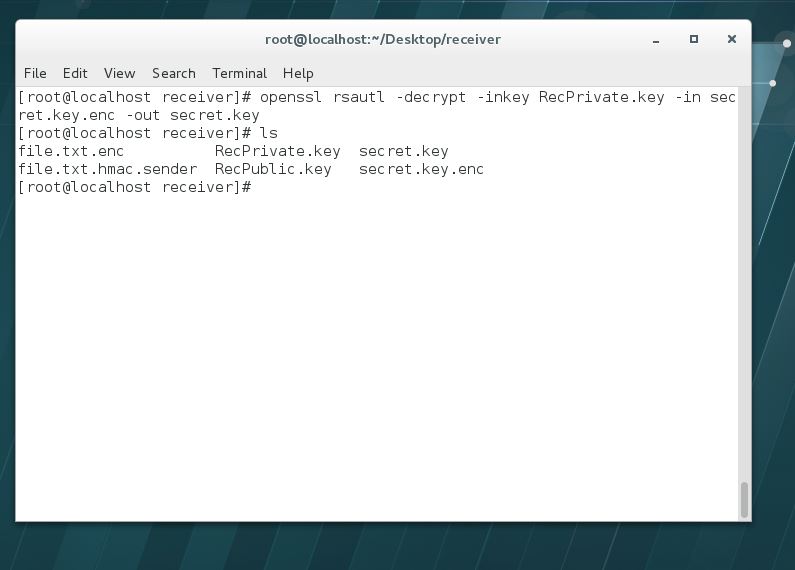




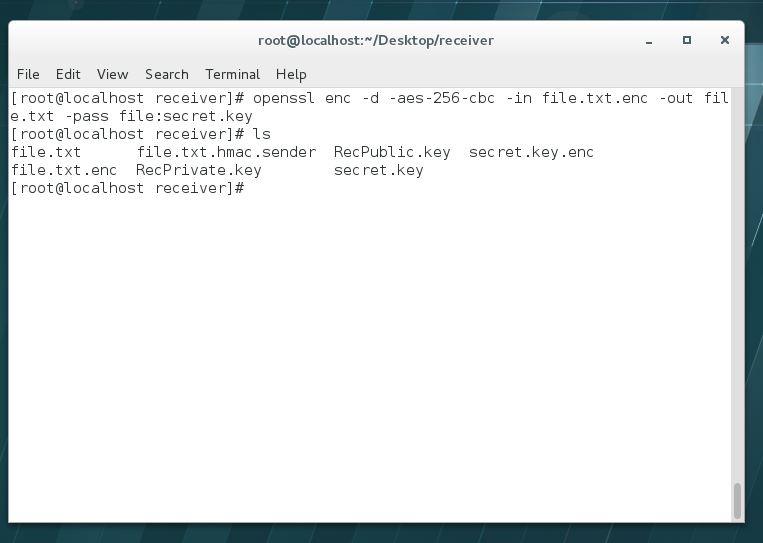
At receiver now:

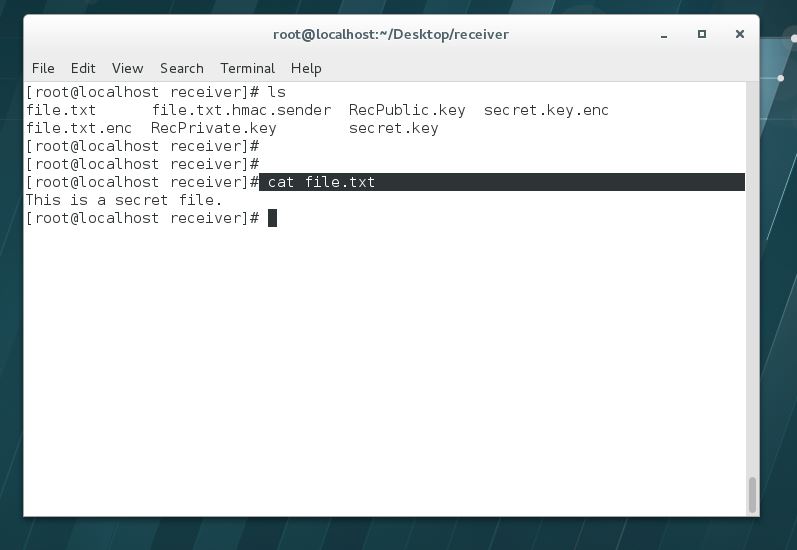


Decrypting Key:

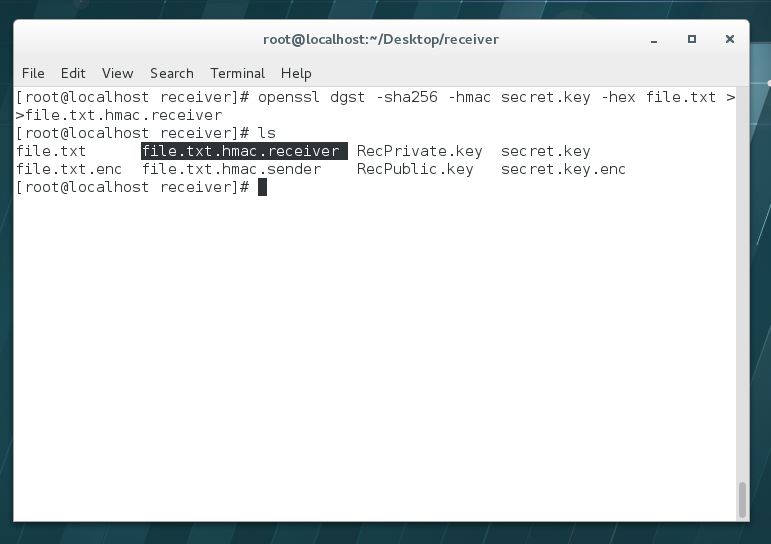


Decrypting File:

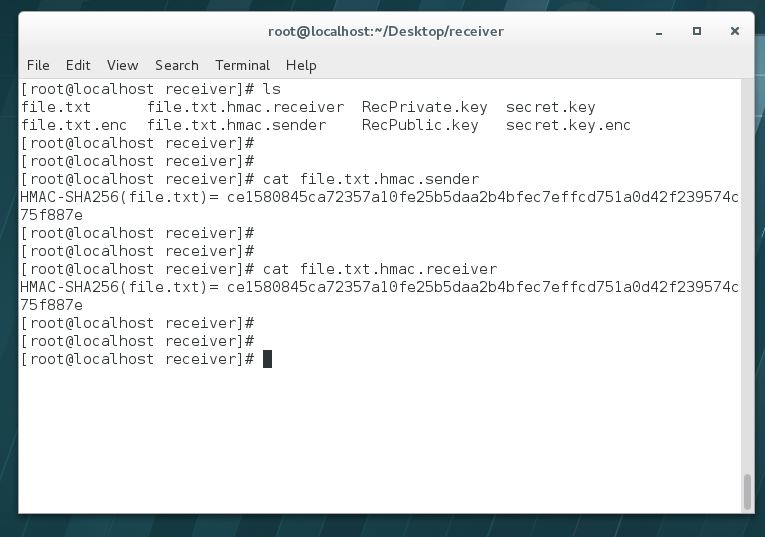




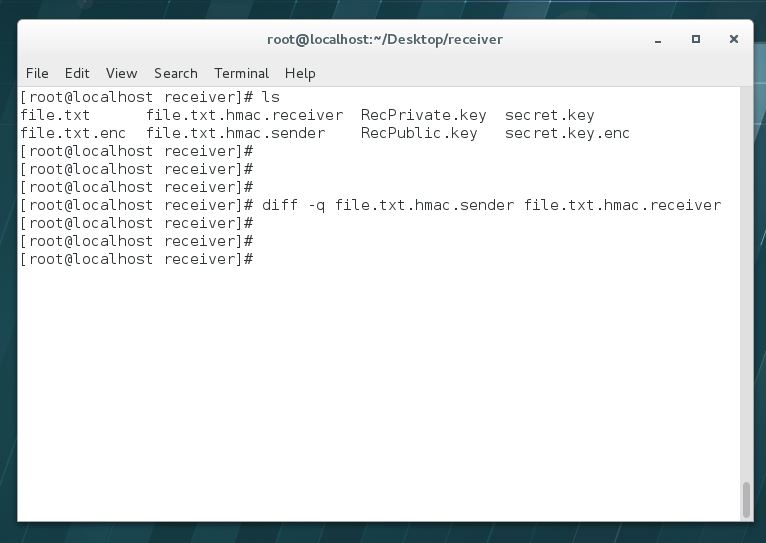
HMAC at receiver:



Comparison of HMAC:



Alternative Way:



**CASE STUDY NO. 1:**

Foreign Account Tax Compliance Act (FATCA)

The Foreign Account Tax Compliance Act (FATCA) is a tax law that compels U.S. citizens at home and abroad to file annual reports on any foreign account holdings.

Nepal and Fatca

Under policy, banks in Nepal are to report information about U.S. citizen account holders to Bank of America.

All banks are required to use the processes mentioned in this report for security purposes to send the report.

Public key is fetched from their website.

**CASE STUDY NO. 2:**

NEPAL SBI BANK

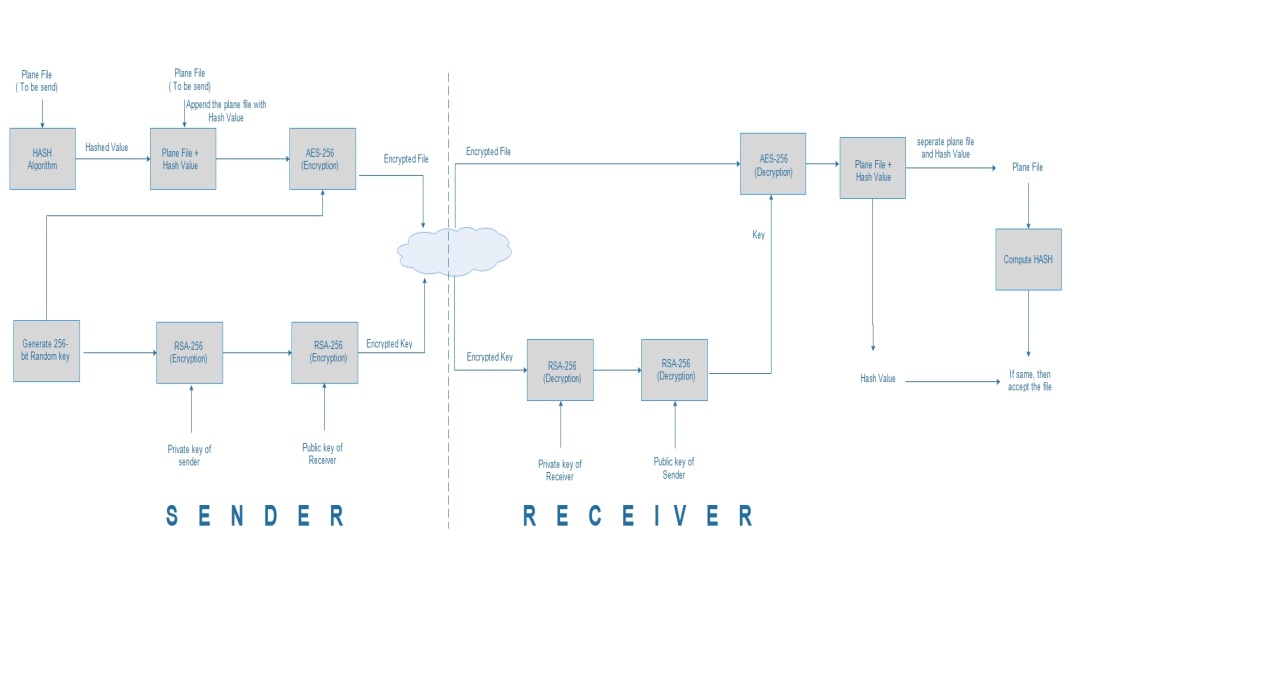
Nepal SBI Bank Ltd. is the first Indo-Nepal joint venture in the financial sector sponsored by three institutional promoters, namely State Bank of India, Employees Provident Fund and Agricultural Development Bank of Nepal through a Memorandum of Understanding signed on 17 July 1992.

The bank has many of its branches all over the world.

The bank also supports transactions from places worldwide out of its branch network for the ease of clients. Information so collected are also transmitted securely by the use of process mentioned in this project.

Example: In Dubai, there are many outlets out of the bank’s branch network where transactions are supported.

**Alternate Sample:**



**REFERENCES:**

Understanding Cryptography, Christof Paar, Jan Pelzl

Springer Publication

**SUGGESTIONS AND FEEDBACK:**

**SUGGESTIONS AND FEEDBACK:**