7481 Take Home Exam

Design & Documentation

Eric Wu, A00961904

Hong Kit, Wu A00968591

BTech 2020 Winter

[**Overview**](#_j7obtqiwk7rt) **2**

[**Propose system**](#_jogx2ipzgx8w) **2**

[**Example scenario**](#_i1c8h16d14tj) **4**

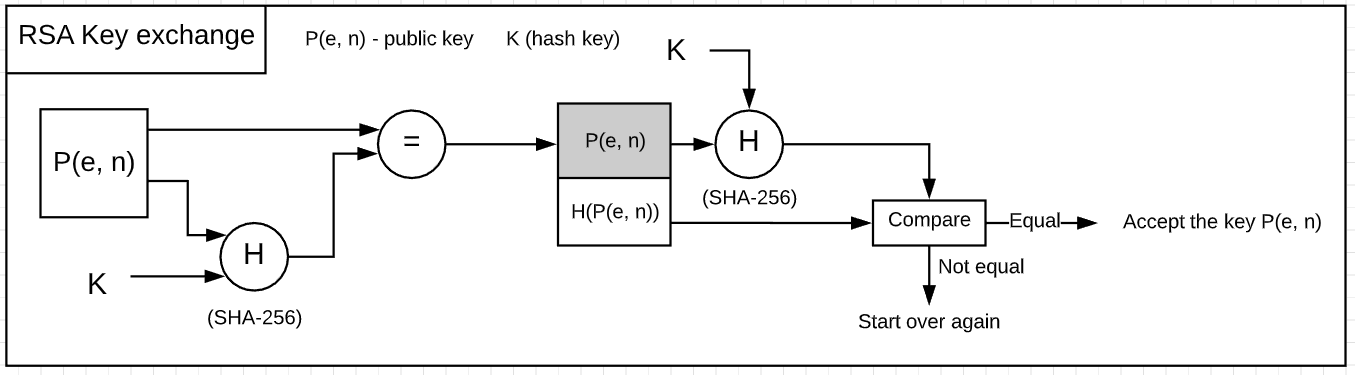
[**Captures and Run-through**](#_4skckzfpt2hm) **5**

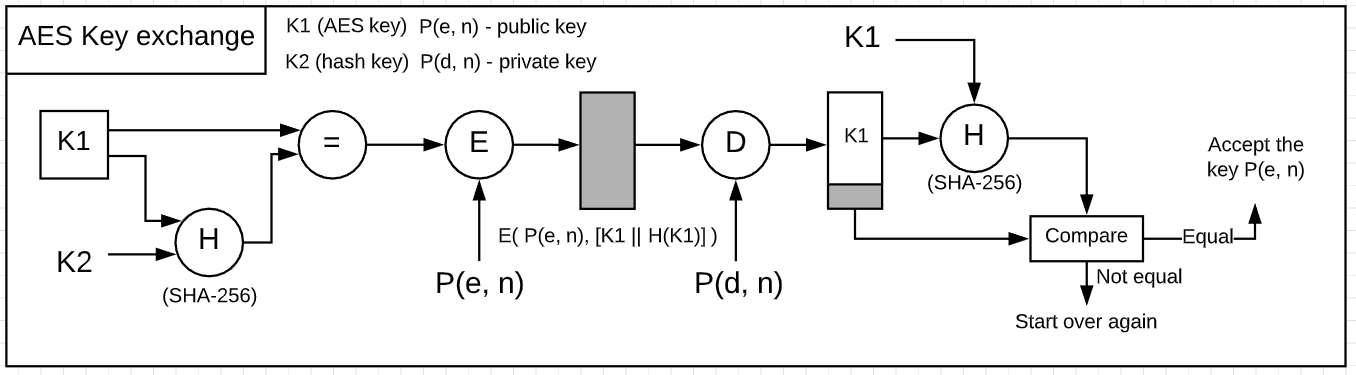
# **Overview**

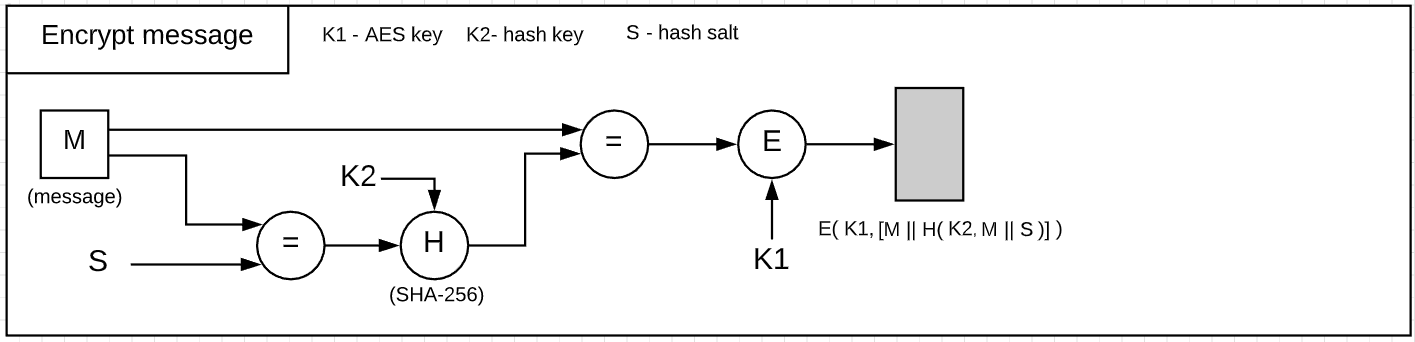
Propose an experimental system, using the algorithms and techniques we have learned in this course. In this system, there are two parties, Bob and Alice, who live far away but wish to communicate with each other. Below are some possible elements of the system (\* bold text are elements used)

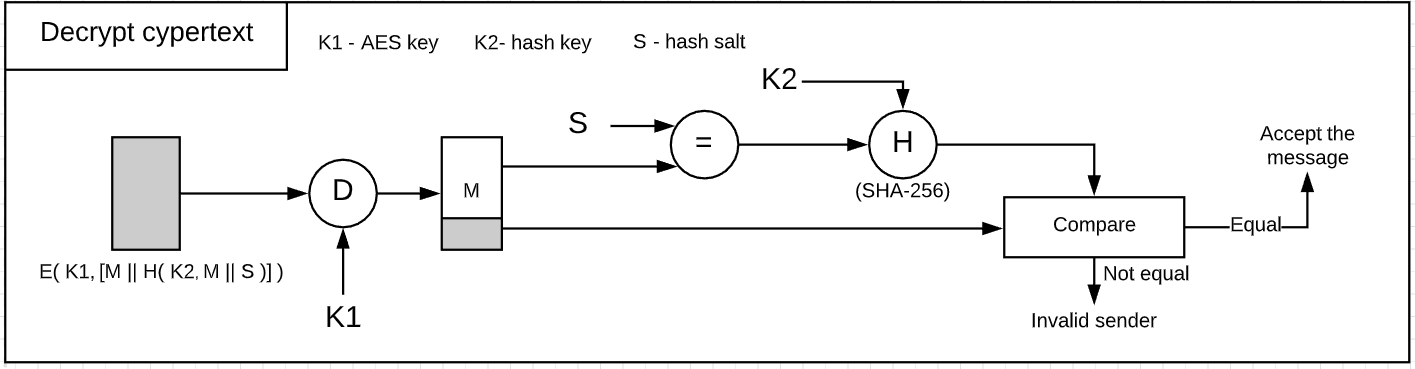
1. **Symmetric key**
2. **Asymmetric keys**
3. **Passphrase (for private key)**
4. **Keyed hash**
5. **Salted hash**
6. Some enemy (Eve)
7. Other

# **Propose system**









In this system, we are assuming that the private key generation involves adding passphrase. The sample system captures all three aspects of cryptography - **confidentiality**, **integrity** and **authentication**. The hash functions are used to ensure that the communication happens only between the intended sender and receiver; It also ensures that the received data has not been altered when transmitted over the channel. The “Salt” added to the hashing procedure strengthens the “authentication” property even more.

Both symmetric and asymmetric are used in this system. Before encrypting the plaintext, the plaintext is first hashed with SHA-256 algorithm along with “Salt”. The symmetric key (AES) is used to encrypt the hashed output to produce ciphertext. The asymmetric key is only used when exchanging the AES key.

# 

# **Example scenario**

Assumption

* Alice is the one who initiates the communication
* Alice and Bob both agree in advance on the hash algorithm (SHA-256), hash salt, and hash key.

**Key exchange**

1. Bob use the passphrase to generate an RSA **P(e, n)** (public key) and **P(d, n)** (private key)
2. Alice use the passphrase to generates an RSA **P(e, n)** and **P(d, n)**
3. Bob hash his public key with hash key “**H( K, P(e, n) )**” and mail it to Alice
4. Alice hash her public key with hash key “**H( K, P(e, n) )**” and mail it to Bob
5. Alice generates a AES key “**K1**”, hash it with hash key “**K2**”, encrypt it with Bob’s public key “**P(e, n)**”, and mail it to Bob “**E( P(e, n), [K1 || H( K2, K1)] )**”
6. Bob receives encrypted AES key, decrypt it with his private key “**P(d, n)**”, check if hash valid, and retrieve the decrypted AES key “**K1**”

**Begin transmission (one side)**

1. Alice append “**salt**” to her secret message “**S || M**”
2. Alice hash the output with hash key “**K2**” and combine it with original message to produce “**M || H( K2, S || M)**”
3. Alice then encrypt the output with both Bob’s public key and passphrase to produce “**E( P(e, n), [M || H( K2, S || M)] )**”, and mail it to Bob
4. Bob received the ciphertext, decrypt with the AES key “**K1**”
5. Bob then run the message through the hash algorithm along with “**salt**” and hash key “**K2**” to check if the message is indeed from Alice
6. If the hash check pass, then Bob has received the actual secret message from Alice
7. Transmission completed

# 

# 

# 

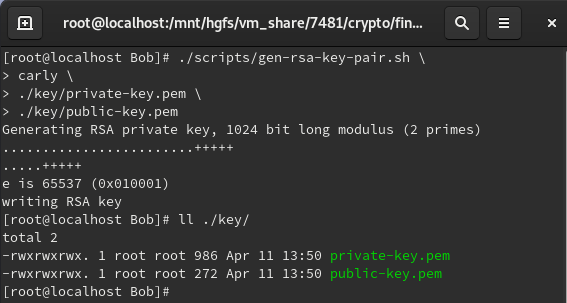
# **Captures and Run-through**

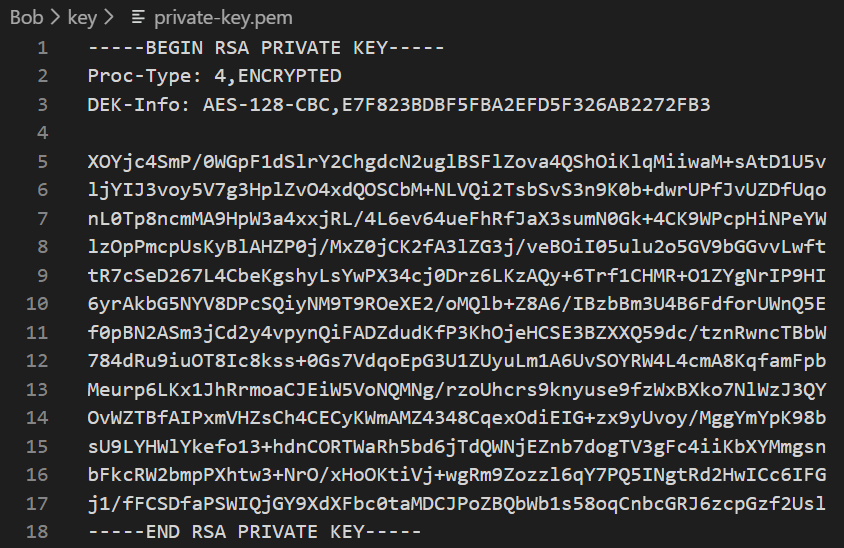
Alice and Bob both agree in advance on the

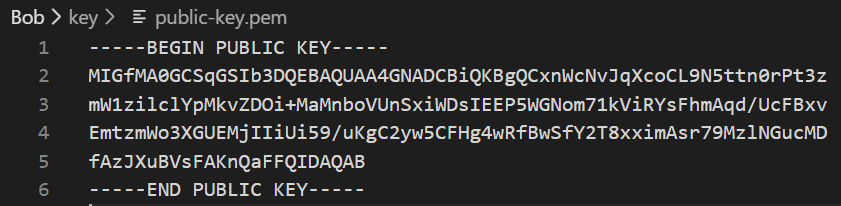
* hash algorithm (SHA-256)
* hash salt and the hash key (refer to README.md).

We are using linux command “**cp**” to simulate mailing behaviour

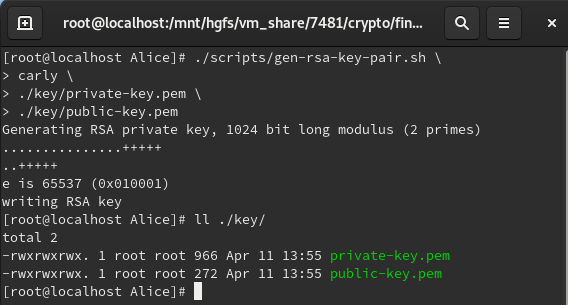
**1)** Bob use the passphrase “**carly**” to generate an RSA **P(e, n)** (public key) and **P(d, n)** (private key)

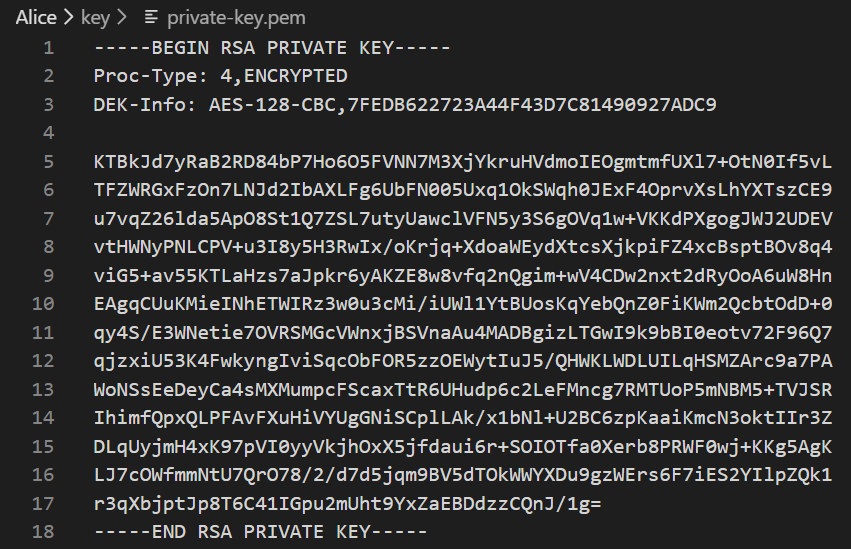


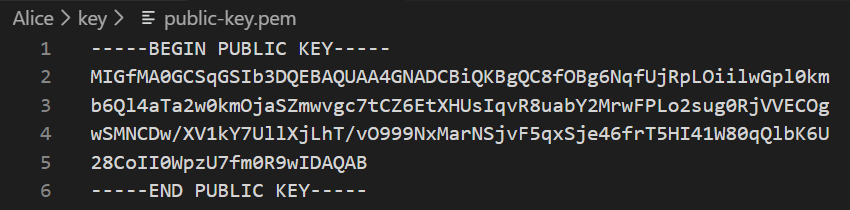




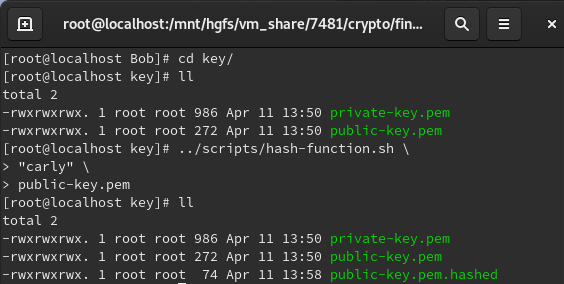
**2)** Alice use the passphrase “**carly**” to generate an RSA **P(e, n)** (public key) and **P(d, n)** (private key)

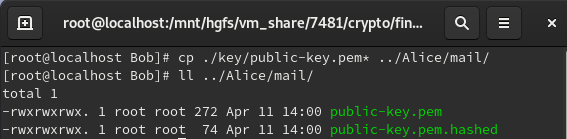




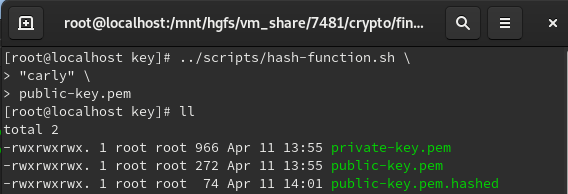


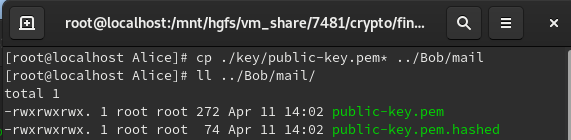
**3)** Bob hash his public key with hash key “**carly**” and mail both the original and hash public key to Alice



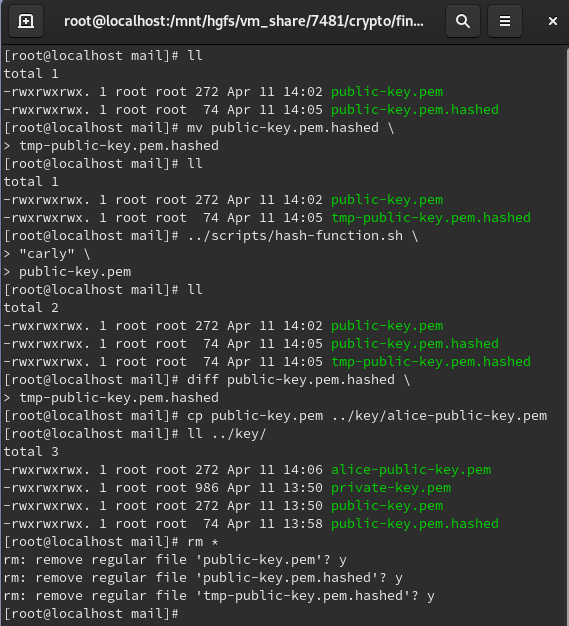


**4)** Alice hash her public key with hash key “**carly**” and mail both the original and hash public key to Bob

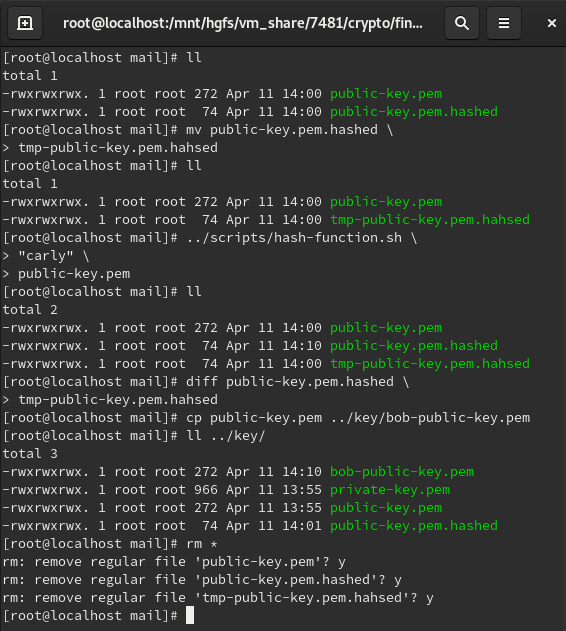




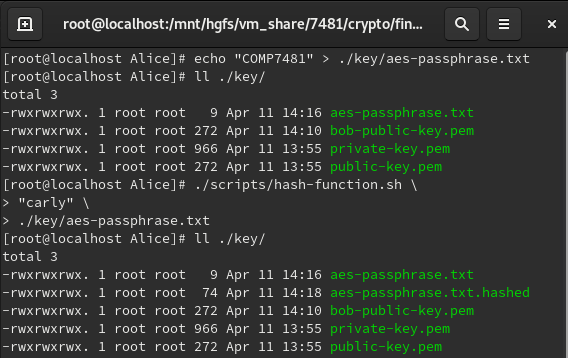
**5)** Bob retrieved both the original and hashed public key from his mailbox, and did a hash check to confirm that the public key is indeed from Alice. He then saved that public key, and cleared the mailbox.

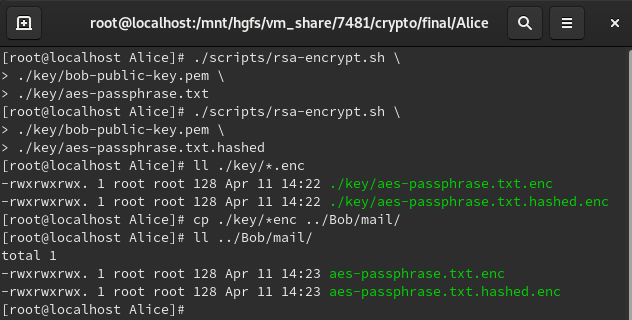


**6)** Alice retrieved both the original and hashed public key from her mailbox, and did a hash check to confirm that the public key is indeed from Bob. She then saved that public key, and cleared the mailbox.

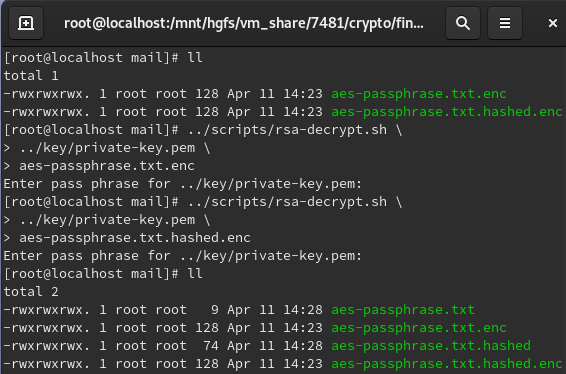


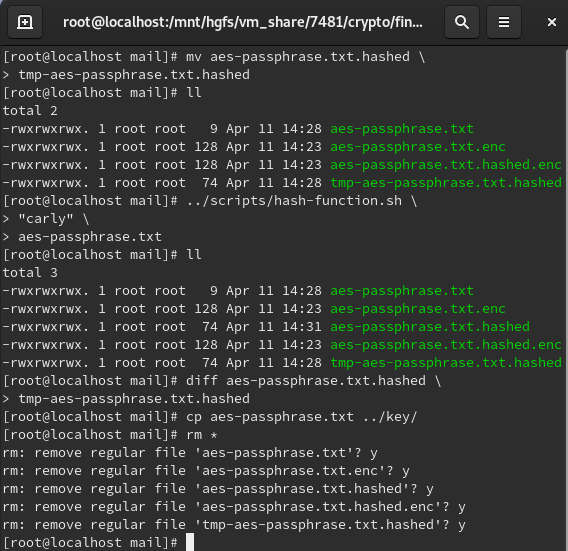
**7)** Alice decided to use the passphrase “**COMP7481**” for AES encryption. To securely pass this passphrase to Bob, she hash the passphrase with hash key “**carly**”, encrypt both the original and hashed passphrase with Bob’s public key, and mail all to Bob

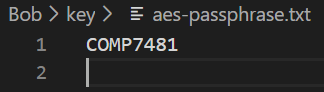




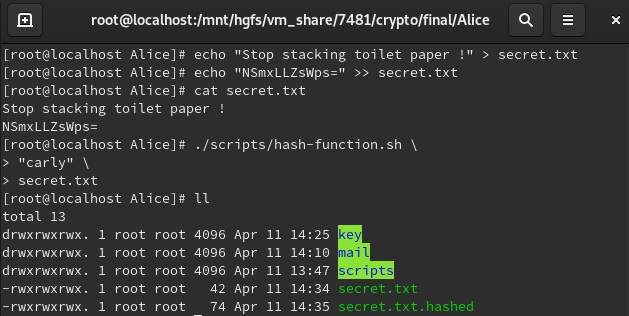
**8)** Bob retrieved both the original and hashed AES key from his mailbox. He first decrypted both files with his private RSA key. Next, he did a hash check to confirm that the AES key is indeed from Alice. He then saved that AES key, and cleared the mailbox.

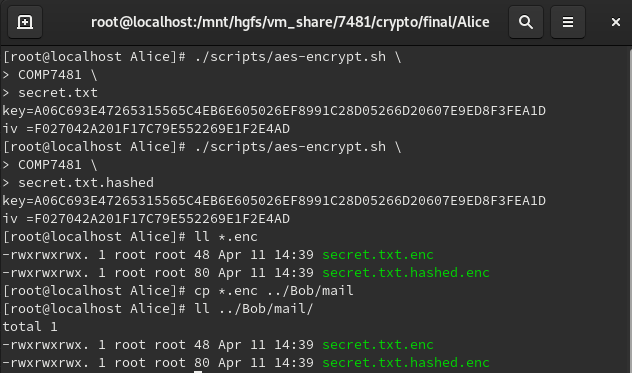






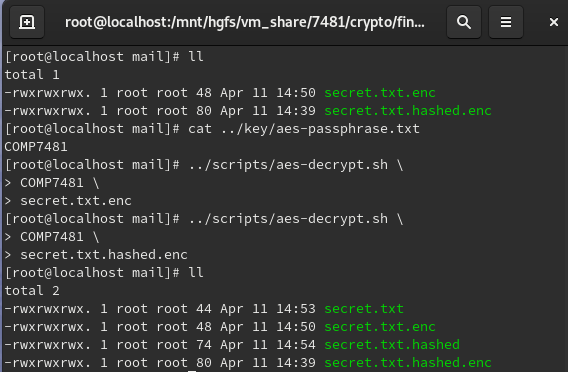
**9)** Bob let Alice know he had successfully retrieved the AES key. (a simple echo would do). Next, Alice created a secret file named “secret.txt”. She appended the hash salt to the text file and hash the file with a hash key. After, she encrypted both the original and hashed file with AES using passphrase “COMP7481”, and directed both files to Bob’s mailbox.

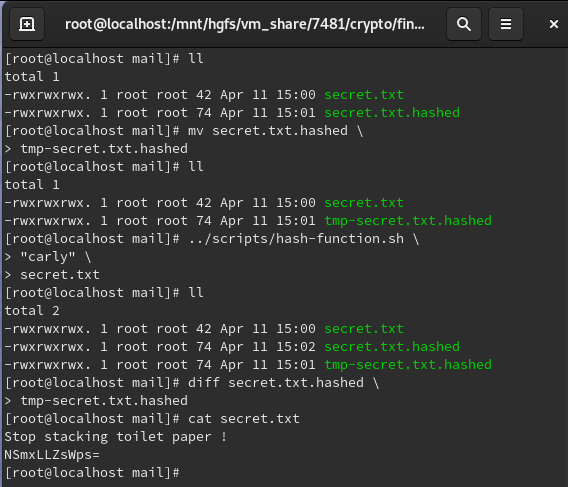




**10)** Bob retrieved the encrypted files from mail. He first decrypted both .enc files with AES using the passphrase received earlier (COMP7481). To confirm that the received secret message has not been altered, he checked the salt and hash. The checking passed.

Bob had successfully receive Alice’s secret message





# **Conclusion**

The system seems fairly robust in terms of security. The RSA passphrase provides an extra layer of protection. For instance, if a third party (Alex) somehow gains access to Bob’s private key, he would have to know the passphrase in order to decrypt the message. The RSA keys are only used for AES key exchange, this decreases the cost for encryption while keeping the security strength at an acceptable level.

To increase the security strength of this system even more, we could choose to use dynamic salt (different salt for every encryption process), different hash algorithm for both key exchange and data encryption, and so on. However, for the scope of this experiment, this system is more than efficient for simple private message exchange.