HACETTEPE UNIVERSITY
ENGINEERING DEPARTMENT
COMPUTER ENGINEERING

LESSON
BBM 465 INFORMATION SECURITY LAB.

ADVISOR
Yasin Şahin, Işıl Karabey

EXPERIMENT 3
PGP and SMIME Usage for Data Communication

GROUP NO : 30

GROUP MEMBERS

Hasan Hüseyin TOPÇU	21228764
Taha BAŞKAK	21228104

PGP

PGP is a program developed by Phil Zimmermann in 1991 that provides data privacy and identity authentication, used to encrypt data, decrypt encrypted data, sign data **[1].** Generally PGP program is used for encrypting, decrypting, signing and verifying documents such as text spoofs, e-mails, files, disk partitions.

PGP uses different encryption algorithms. RSA has a hybrid structure formed by using the asymmetric encryption method and the Bass-O-Matic encryption algorithm, which is also a symmetric encryption method developed by itself. When you want to encrypt a text with PGP, PGP compresses the text. If you want to send an e-mail, the contents of the e-mail will be encrypted with the public key of the sender. In file signing, the file is summed with hashing and encrypted with the sender's private key. The recipient decrypts the hash code encrypted with the public key of the opposite party.

Encryption in PGP provide the file to be moved securely. Signing ensures that the integrity of the message, that is, it changes on the road, and confirms the identity of the sender.

The general functioning of PGP;

- The file to be sent by the sender is signed with the sender's secret key.
- The signed file is then encrypted with the recipient's open key and sent to the recipient.
- The signed and encrypted file is delivered to the recipient.
- The encrypted file, which was signed before it was received by the recipient, is initially decrypted with the recipient's secret key. Decryption is performed because the file receiver is encrypted with the open key and the secret key is known to the public key.
- The signed message is then decrypted with the public key of the sender and the actual text is reached.

Sign-then-encrypt

In this method, the file is signed first, then encrypted. We used this method in our experience because anyone with the sender's public key can do the validation. If it is signed first and then encrypted, it must be decrypted before it can be verified. This requires the recipient's public anchor. So the signed file can not be obtained from the receiver. It prevents some attacks such as "against existential forgery attack".

Encrypt-then-sign

In this method, the file is first encrypted and then signed. This method is generally not recommended because it is vulnerable to some attacks. Anyone can do verification because everyone can access the sender's public key. This leads to some kind of devastation.

S/MIME

S/MIME is a standard that uses public key cryptography and data signing. The first version was not as powerful as its competence. It was developed by the IETF to be very strong.

S/MIME digital signing and message encryption services. Data security, authentication, data integrity, privacy, and denial are provided by these two services.

Digital signing has the same authority as the signature of the person who is legally seen. Digital signing provides authentication, decryption and data integrity. But it can not provide privacy and data security. Authentication verifies who is sending the message and by whom. This also prevents denial. Since the message is intended to be sent, the signing process only contains information that can be provided by the sender. It will capture the e-mail that will be sent with this information and perform the signing process and produce the digital signature. But here too, signing can not provide security. The information sent can be used by others. But this solution can also be resolved by the S / MIME standard. In verifying the sent e-mail, the message and signing information is compared with the signature information to be generated and digital signature control is provided.

Encryption will change the signed message until it can not be read or understood. Although not used as much as signing, it is very important for privacy and data security.

Signing and encryption are different from each other. However, the use of these two provides security and authentication and ensures that data is transmitted accurately and reliably, forming the message security strategy. These two complement each other. The processing logic of the message security certificate:

Signing and encryption of the message; The message is captured, the sender's information is received, the recipient's information is received, the signing operation is carried out with the sender's information and a signature is added to the message, the message is encrypted with the recipient's information and the message is sent.

Deciphering and verifying the message; The encrypted message is received and decrypted with the receiver's information, the decrypted message is sent to the receiver, the signature is taken from the message, the sample signature is generated with the sender's information and compared with the signature received from the message, the validity and integrity of the message is confirmed.

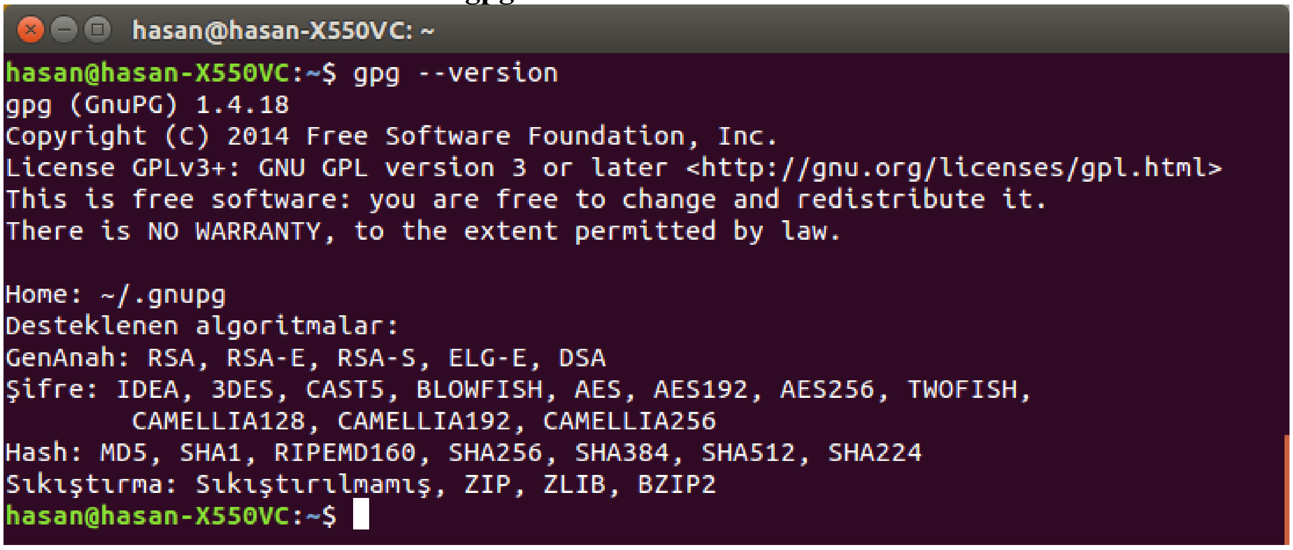
EXPERIMENT PHASES

PGP

There are two peers in this experiment. These can be called receiver peer and sender peer. The phases to be performed below are specified separately for the receiver and sender peer. Group members change role and phases are done again. Because of applied steps are the same, due to the repetition of sender and receiver commands, it was not written separately for each group member. Explanations and screenshot of commands for sender and receiver peer are given.

/ Sender Peer */*
Sender name : hasan huseyin topcu
Sender email : hasanhuseyintopcuu@gmail.com

sudo apt-get install gnupg
#Install pgp application for ubuntu operating system.
#pgp application is automatically installed for ubuntu 12.04 and above. Thus we don't installed.
#Version control can be done with "**gpg --version**" command if desired.

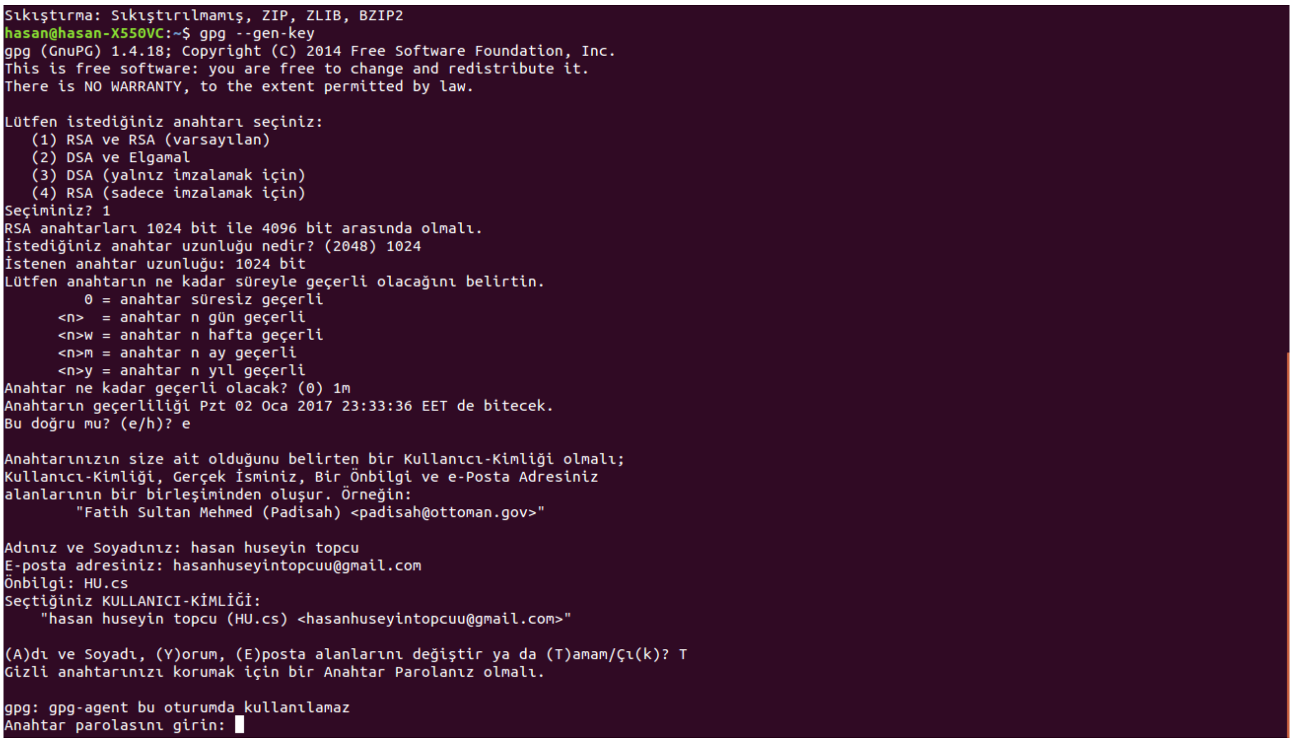


```
hasan@hasan-X550VC: ~
hasan@hasan-X550VC:~$ gpg --version
gpg (GnuPG) 1.4.18
Copyright (C) 2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: ~/.gnupg
Desteklenen algoritmalar:
GenAnah: RSA, RSA-E, RSA-S, ELG-E, DSA
Şifre: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
        CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Sıkıştırma: Sıkıştırılmamış, ZIP, ZLIB, BZIP2
hasan@hasan-X550VC:~$
```

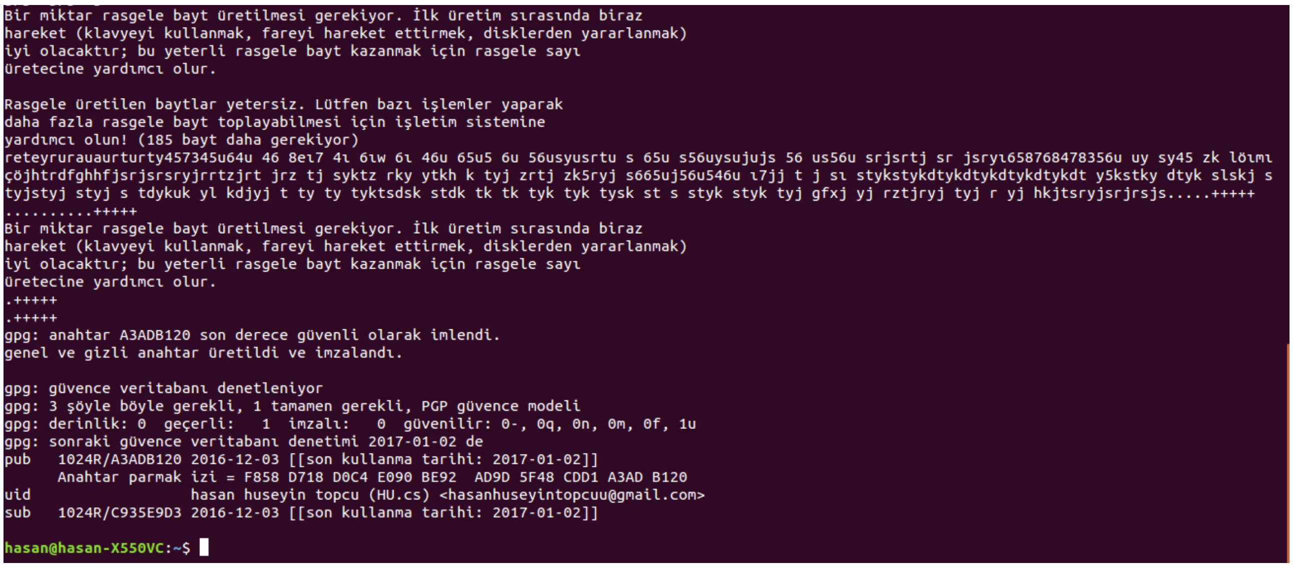
gpg --gen-key
#Generate a new key pair. Public and private key.
#There are options for the key pair to be created.
#These :

The type of algorithm to use to generate the keys : default option is 1
Key size : default size is 2048
Lifetime of the keys : default lifetime is indefinite
Verification question of lifetime of the keys : e or h
Name and Last Name
E-mail Address
Comment : Optional

#After all options if you approve then enter "T"
#And we must enter a passphrase to protect the secret key.
#Finally generate random byte to generate a new key pair but this operation done randomly therefore we can help using mouse or keyboard.

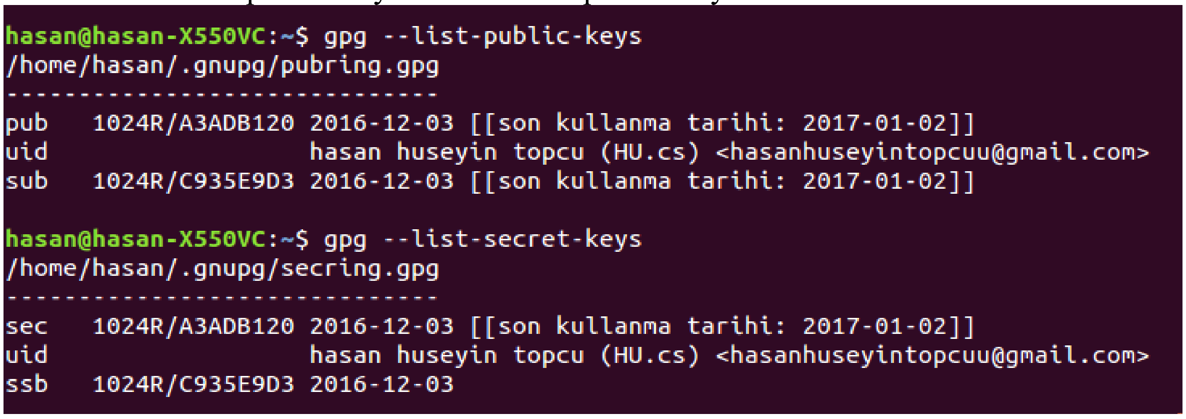


```
Sıkıştırma: Sıkıştırılmamış, ZIP, ZLIB, BZIP2
hasan@hasan-X550VC:~$ gpg --gen-key
gpg (GnuPG) 1.4.18; Copyright (C) 2014 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Lütfen istediğiniz anahtarı seçiniz:
   (1) RSA ve RSA (varsayılan)
   (2) DSA ve Elgamal
   (3) DSA (yalnız imzalamak için)
   (4) RSA (sadece imzalamak için)
Seçiminiz? 1
RSA anahtarları 1024 bit ile 4096 bit arasında olmalı.
İstediğiniz anahtar uzunluğu nedir? (2048) 1024
İstenen anahtar uzunluğu: 1024 bit
Lütfen anahtarın ne kadar süreyle geçerli olacağını belirtin.
         0 = anahtar süresiz geçerli
      <n>  = anahtar n gün geçerli
      <n>w = anahtar n hafta geçerli
      <n>m = anahtar n ay geçerli
      <n>y = anahtar n yıl geçerli
Anahtar ne kadar geçerli olacak? (0) 1m
Anahtarın geçerliliği Pzt 02 Oca 2017 23:33:36 EET de bitecek.
Bu doğru mu? (e/h)? e

Anahtarınızın size ait olduğunu belirten bir Kullanıcı-Kimliği olmalı;
Kullanıcı-Kimliği, Gerçek İsminiz, Bir Önbilgi ve e-Posta Adresiniz
alanlarının bir birleşiminden oluşur. Örneğin:
        "Fatih Sultan Mehmed (Padisah) <padisah@ottoman.gov>"

Adınız ve Soyadınız: hasan huseyin topcu
E-posta adresiniz: hasanhuseyintopcuu@gmail.com
Önbilgi: HU.cs
Seçtiğiniz KULLANICI-KİMLİĞİ:
    "hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>"

(A)dı ve Soyadı, (Y)orum, (E)posta alanlarını değiştir ya da (T)amam/Çı(k)? T
Gizli anahtarınızı korumak için bir Anahtar Parolanız olmalı.

gpg: gpg-agent bu oturumda kullanılamaz
Anahtar parolasını girin: 
```



```
Bir miktar rasgele bayt üretilmesi gerekiyor. İlk üretim sırasında biraz
hareket (klavyeyi kullanmak, fareyi hareket ettirmek, disklerden yararlanmak)
iyi olacaktır; bu yeterli rasgele bayt kazanmak için rasgele sayı
üretecine yardımcı olur.

Rasgele üretilen baytlar yetersiz. Lütfen bazı işlemler yaparak
daha fazla rasgele bayt toplayabilmesi için işletim sistemine
yardımcı olun! (185 bayt daha gerekiyor)
reteyrurauaurturty457345u64u 46 8eı7 4ı 6ıw 6ı 46u 65u5 6u 56usyusrtu s 65u s56uysujujs 56 us56u srjsrtj sr jsryı658768478356u uy sy45 zk löımı
çöjhtrdfghhfjsrjsrsryjrrtzjrt jrz tj syktz rky ytkh k tyj zrtj zk5ryj s665uj56u546u ı7jj t j sı stykstykdtykdtykdtykdtykdt y5kstky dtyk slskj s
tyjstyj styj s tdykuk yl kdjyj t ty ty tyktsdsk stdk tk tk tyk tyk tysk st s styk styk tyj gfxj yj rztjryj tyj r yj hkjtsryjsrjrsjs.....+++++
..........+++++
Bir miktar rasgele bayt üretilmesi gerekiyor. İlk üretim sırasında biraz
hareket (klavyeyi kullanmak, fareyi hareket ettirmek, disklerden yararlanmak)
iyi olacaktır; bu yeterli rasgele bayt kazanmak için rasgele sayı
üretecine yardımcı olur.
.+++++
.+++++
gpg: anahtar A3ADB120 son derece güvenli olarak imlendi.
genel ve gizli anahtar üretildi ve imzalandı.

gpg: güvence veritabanı denetleniyor
gpg: 3 şöyle böyle gerekli, 1 tamamen gerekli, PGP güvence modeli
gpg: derinlik: 0  geçerli:   1  imzalı:   0  güvenilir: 0-, 0q, 0n, 0m, 0f, 1u
gpg: sonraki güvence veritabanı denetimi 2017-01-02 de
pub   1024R/A3ADB120 2016-12-03 [[son kullanma tarihi: 2017-01-02]]
      Anahtar parmak izi = F858 D718 D0C4 E090 BE92  AD9D 5F48 CDD1 A3AD B120
uid                  hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>
sub   1024R/C935E9D3 2016-12-03 [[son kullanma tarihi: 2017-01-02]]

hasan@hasan-X550VC:~$ 
```

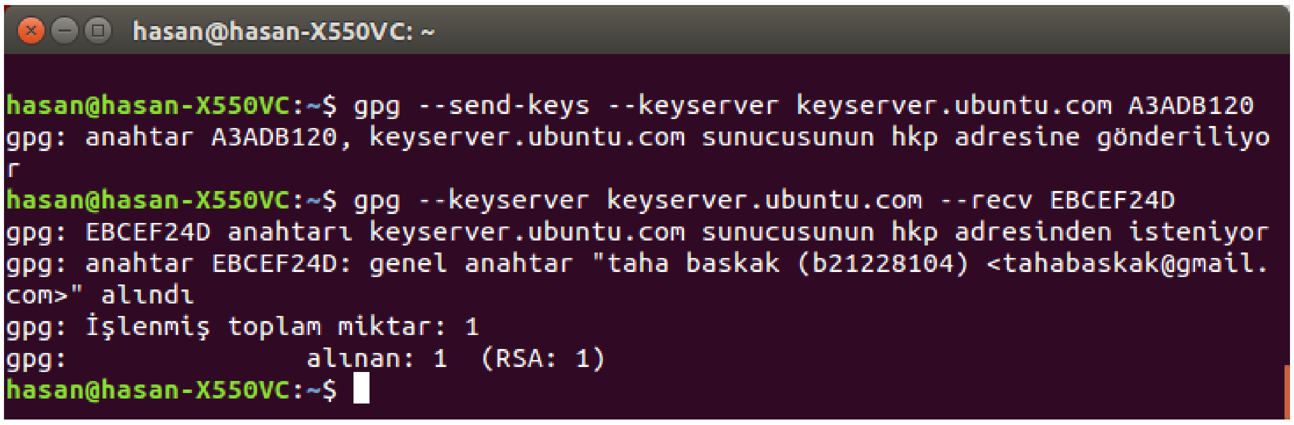
gpg --list-public-keys
#We can see the list of public keys. There is just own public key.
gpg –list-secret-keys
#We can see the list of private keys. There is one private key.



```
hasan@hasan-X550VC:~$ gpg --list-public-keys
/home/hasan/.gnupg/pubring.gpg
-------------------------------
pub   1024R/A3ADB120 2016-12-03 [[son kullanma tarihi: 2017-01-02]]
uid                  hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>
sub   1024R/C935E9D3 2016-12-03 [[son kullanma tarihi: 2017-01-02]]

hasan@hasan-X550VC:~$ gpg --list-secret-keys
/home/hasan/.gnupg/secring.gpg
-------------------------------
sec   1024R/A3ADB120 2016-12-03 [[son kullanma tarihi: 2017-01-02]]
uid                  hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>
ssb   1024R/C935E9D3 2016-12-03
```

gpg --send-keys --keyserver keyserver.ubuntu.com keyid
#Send public key with the xxxxxx id number with the last parameter to the keyserver.ubuntu.com
#Sender peer must export own public key to receiver peer.

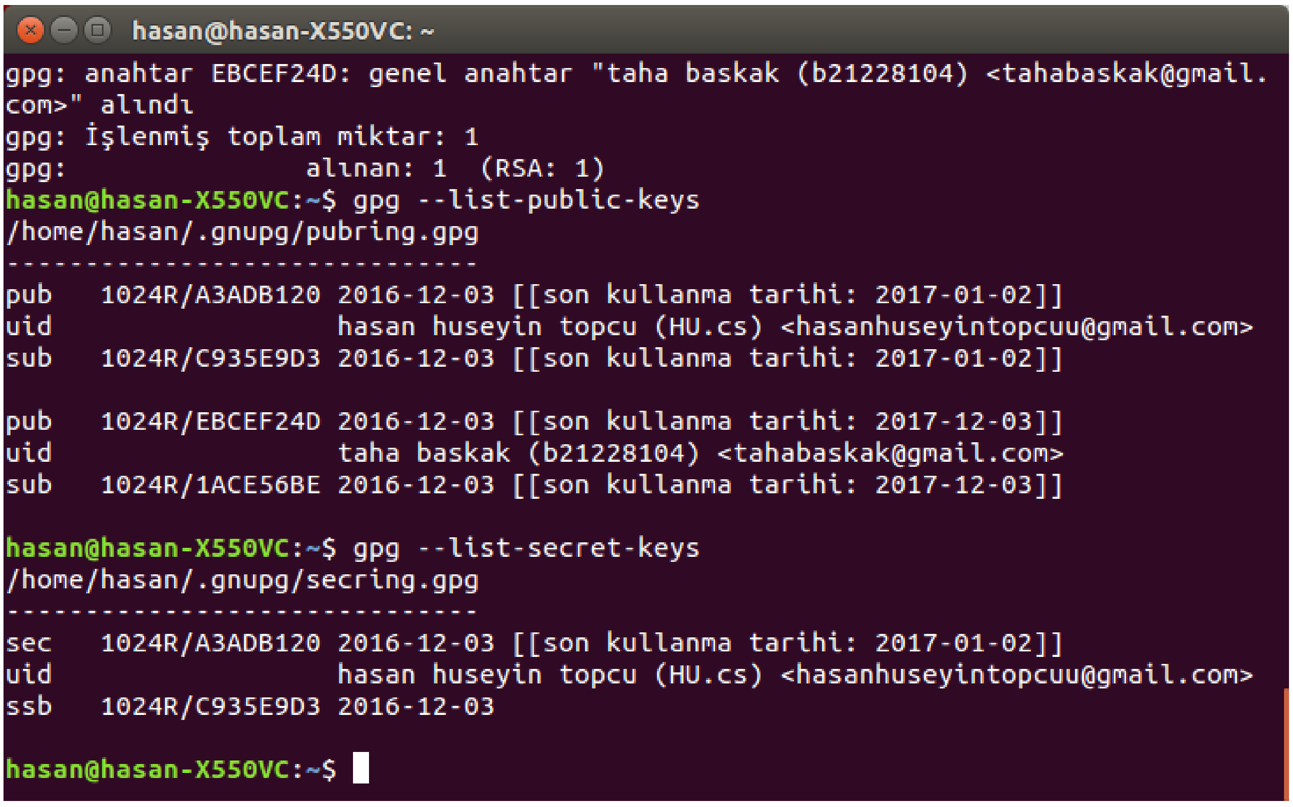
gpg --keyserver keyserver.ubuntu.com --recv keyid
#Import public key with the xxxxxx id number with the last parameter from keyserver.ubuntu.com
#Sender peer import public key from receiver peer.
#To find the key id number of the receiver's public key, we can go to any key server web site and find it by search email address or comment or date vs.



```
hasan@hasan-X550VC:~$ gpg --send-keys --keyserver keyserver.ubuntu.com A3ADB120
gpg: anahtar A3ADB120, keyserver.ubuntu.com sunucusunun hkp adresine gönderiliyo
r
hasan@hasan-X550VC:~$ gpg --keyserver keyserver.ubuntu.com --recv EBCEF24D
gpg: EBCEF24D anahtarı keyserver.ubuntu.com sunucusunun hkp adresinden isteniyor
gpg: anahtar EBCEF24D: genel anahtar "taha baskak (b21228104) <tahabaskak@gmail.
com>" alındı
gpg: İşlenmiş toplam miktar: 1
gpg:               alınan: 1  (RSA: 1)
hasan@hasan-X550VC:~$ 
```

gpg --list-public-keys
#Now, we can see two public keys. Sender's public key and receiver's public key.

gpg –list-secret-keys
#We can only see own secret key again.



```
gpg: anahtar EBCEF24D: genel anahtar "taha baskak (b21228104) <tahabaskak@gmail.
com>" alındı
gpg: İşlenmiş toplam miktar: 1
gpg:               alınan: 1  (RSA: 1)
hasan@hasan-X550VC:~$ gpg --list-public-keys
/home/hasan/.gnupg/pubring.gpg
------------------------------
pub   1024R/A3ADB120 2016-12-03 [[son kullanma tarihi: 2017-01-02]]
uid                  hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>
sub   1024R/C935E9D3 2016-12-03 [[son kullanma tarihi: 2017-01-02]]

pub   1024R/EBCEF24D 2016-12-03 [[son kullanma tarihi: 2017-12-03]]
uid                  taha baskak (b21228104) <tahabaskak@gmail.com>
sub   1024R/1ACE56BE 2016-12-03 [[son kullanma tarihi: 2017-12-03]]

hasan@hasan-X550VC:~$ gpg --list-secret-keys
/home/hasan/.gnupg/secring.gpg
------------------------------
sec   1024R/A3ADB120 2016-12-03 [[son kullanma tarihi: 2017-01-02]]
uid                  hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>
ssb   1024R/C935E9D3 2016-12-03

hasan@hasan-X550VC:~$ 
```

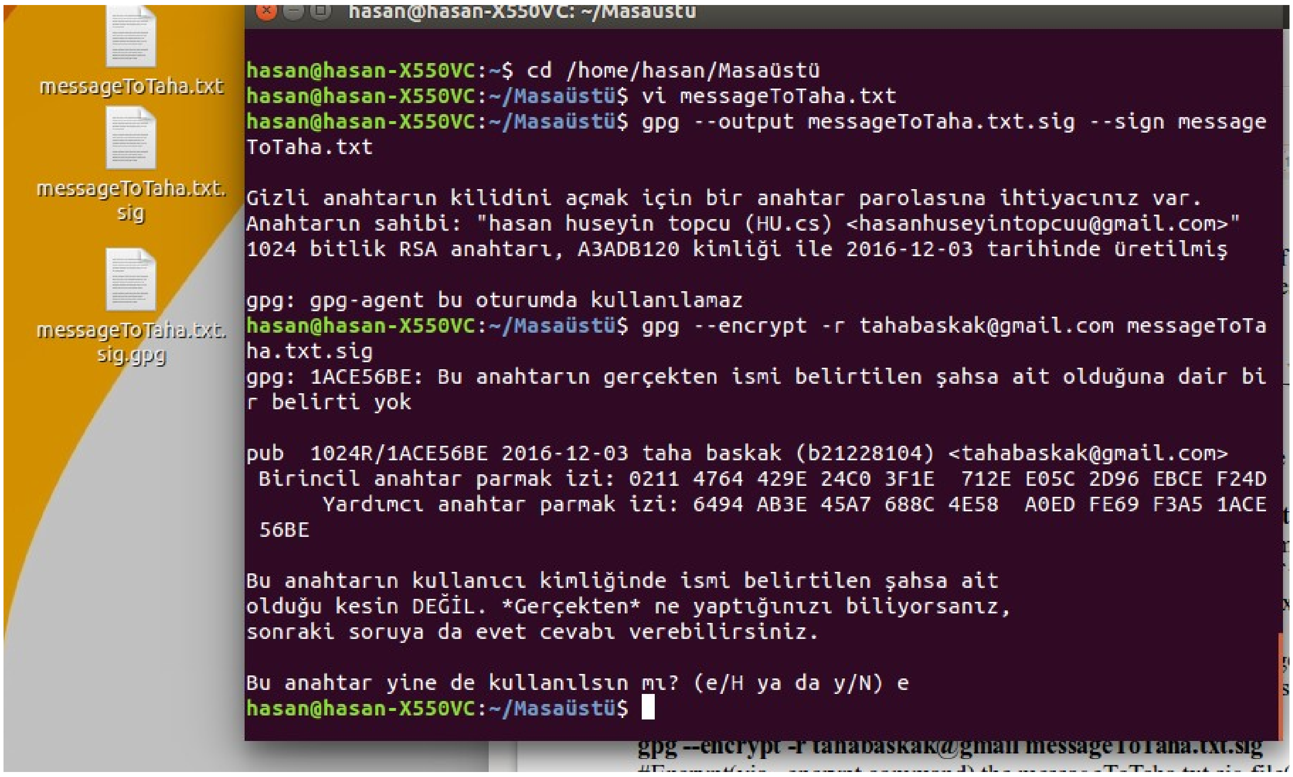
cd /home/hasan/Masaüstü
#We change directory to just be easy. There is not necessary. If not change directory you must be careful for the files to be created and the directories of the files given as parameters for the next commands.

<u>Encrypt and Sign Unique Message Part(messageToTaha.txt File)</u>

vi messageToTaha.txt
#Create or enter messageToTaha.txt file using vi editor and write secret message.

gpg --output messageToTaha.txt.sig --sign messageToTaha.txt
#Sign(via –sign command) the messageToTaha.txt file(last parameter)
#”--ouput” command is optional. If not use then the extension of the created file is gpg
#If use –output command then we can determine the name and extension of the file to be created
#Use sender's private key to signing
#When this command is executed, ask for the passphrase while generating the secret key
#The operation of this command creates the messageToTaha.txt.sig file in the current directory

gpg --encrypt -r tahabaskak@gmail messageToTaha.txt.sig
#Encrypt(via –encrypt command) the messageToTaha.txt.sig file(last parameter)
#Use receiver's public key to encrypting
#With the parameter given to the -r command, we specify who uses the public key
#If “--output” command not use, then file name is same and extension is default so gpg.
#The operation of this command creates the messageToTaha.txt.sig.gpg file in the current directory



```
hasan@hasan-X550VC:~$ cd /home/hasan/Masaüstü
hasan@hasan-X550VC:~/Masaüstü$ vi messageToTaha.txt
hasan@hasan-X550VC:~/Masaüstü$ gpg --output messageToTaha.txt.sig --sign message
ToTaha.txt

Gizli anahtarın kilidini açmak için bir anahtar parolasına ihtiyacınız var.
Anahtarın sahibi: "hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>"
1024 bitlik RSA anahtarı, A3ADB120 kimliği ile 2016-12-03 tarihinde üretilmiş

gpg: gpg-agent bu oturumda kullanılamaz
hasan@hasan-X550VC:~/Masaüstü$ gpg --encrypt -r tahabaskak@gmail.com messageToTa
ha.txt.sig
gpg: 1ACE56BE: Bu anahtarın gerçekten ismi belirtilen şahsa ait olduğuna dair bi
r belirti yok

pub  1024R/1ACE56BE 2016-12-03 taha baskak (b21228104) <tahabaskak@gmail.com>
 Birincil anahtar parmak izi: 0211 4764 429E 24C0 3F1E  712E E05C 2D96 EBCE F24D
      Yardımcı anahtar parmak izi: 6494 AB3E 45A7 688C 4E58  A0ED FE69 F3A5 1ACE
 56BE

Bu anahtarın kullanıcı kimliğinde ismi belirtilen şahsa ait
olduğu kesin DEĞİL. *Gerçekten* ne yaptığınızı biliyorsanız,
sonraki soruya da evet cevabı verebilirsiniz.

Bu anahtar yine de kullanılsın mı? (e/H ya da y/N) e
hasan@hasan-X550VC:~/Masaüstü$
```

Encrypt and Sign Unique Zip File Part(fileToTaha.zip File)

mkdir fileToTaha
#Creat fileToTaha file.

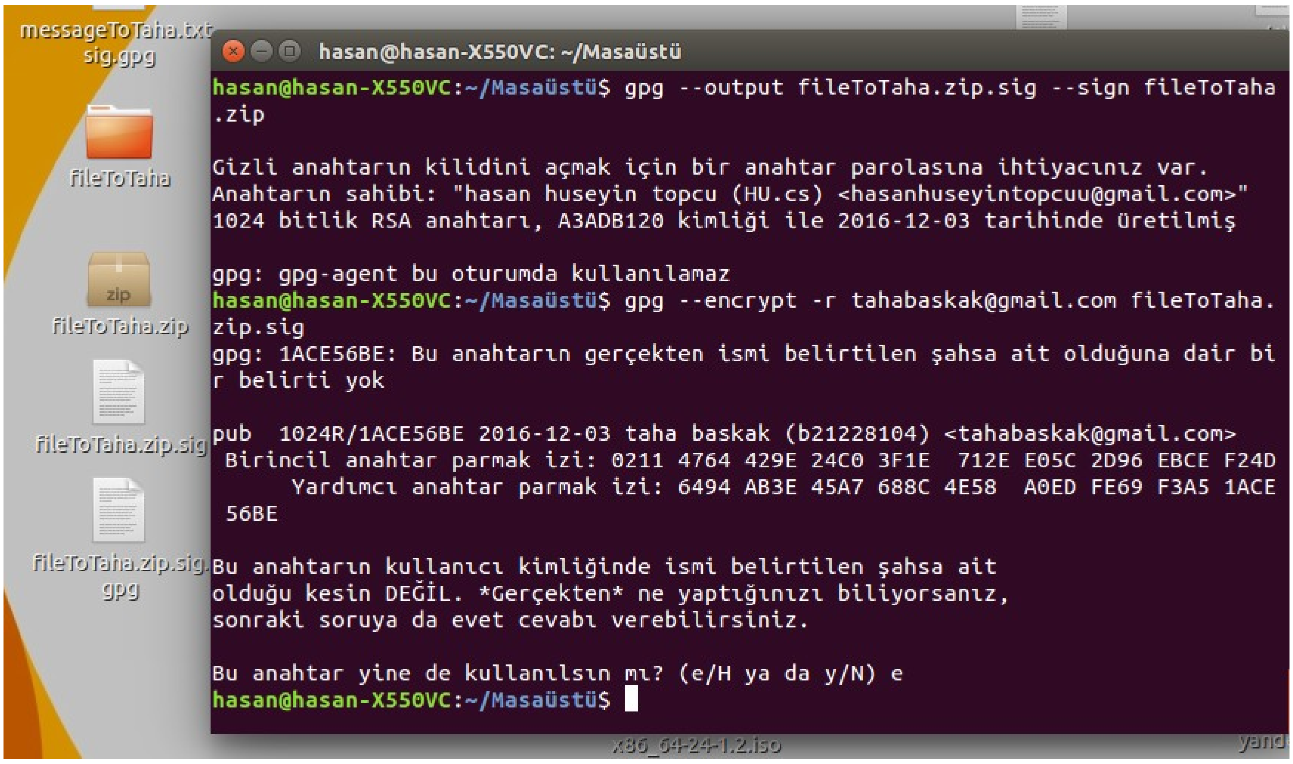
cd fileToTaha
#Enter fileToTaha file.

mkdir photo message
#Create sub file and copy photo in photo file, write yout message in message.txt in message file

zip -r fileToTaha.zip fileToTaha
#Zip fileToTaha file as name fileToTaha.zip

gpg --output fileToTaha.zip.sig --sign fileToTaha.zip
#Sign zip file via –sign command and create fileToTaha.zip.sig file.
#Sign file using sender's private key.
#When this command is executed, ask for the passphrase while generating the secret key
#The operation of this command creates the fileToTaha.zip.sig file in the current directory

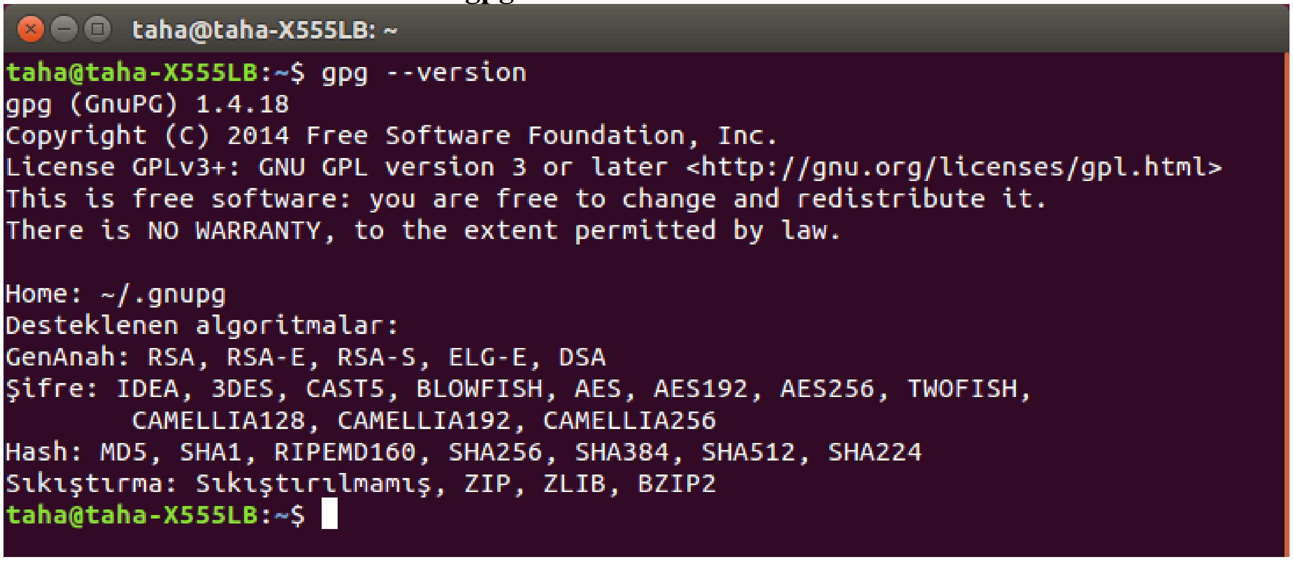
gpg --encrypt -r tahabaskak@gmail fileToTaha.zip.sig
#Encrypt(via –encrypt command) the fileToTaha.zip.sig file(last parameter)
#Use receiver's public key to encrypting
#With the parameter given to the -r command, we specify who uses the public key
#If "--output" command not use, then file name is same and extension is default so gpg.
#The operation of this command creates the fileToTaha.zip.sig.gpg file in the current directory



```
hasan@hasan-X550VC:~/Masaüstü$ gpg --output fileToTaha.zip.sig --sign fileToTaha
.zip

Gizli anahtarın kilidini açmak için bir anahtar parolasına ihtiyacınız var.
Anahtarın sahibi: "hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>"
1024 bitlik RSA anahtarı, A3ADB120 kimliği ile 2016-12-03 tarihinde üretilmiş

gpg: gpg-agent bu oturumda kullanılamaz
hasan@hasan-X550VC:~/Masaüstü$ gpg --encrypt -r tahabaskak@gmail.com fileToTaha.
zip.sig
gpg: 1ACE56BE: Bu anahtarın gerçekten ismi belirtilen şahsa ait olduğuna dair bi
r belirti yok

pub  1024R/1ACE56BE 2016-12-03 taha baskak (b21228104) <tahabaskak@gmail.com>
 Birincil anahtar parmak izi: 0211 4764 429E 24C0 3F1E  712E E05C 2D96 EBCE F24D
      Yardımcı anahtar parmak izi: 6494 AB3E 45A7 688C 4E58  A0ED FE69 F3A5 1ACE
 56BE

Bu anahtarın kullanıcı kimliğinde ismi belirtilen şahsa ait
olduğu kesin DEĞİL. *Gerçekten* ne yaptığınızı biliyorsanız,
sonraki soruya da evet cevabı verebilirsiniz.

Bu anahtar yine de kullanılsın mı? (e/H ya da y/N) e
hasan@hasan-X550VC:~/Masaüstü$
```

/ Receiver Peer */*
receiver name : taha baskak
receiver email : tahabaskak@gmail

sudo apt-get install gnupg
#Install pgp application for ubuntu operating system.
#pgp application is automatically installed for ubuntu 12.04 and above. Thus we don't installed.
#Version control can be done with “**gpg --version**” command if desired.

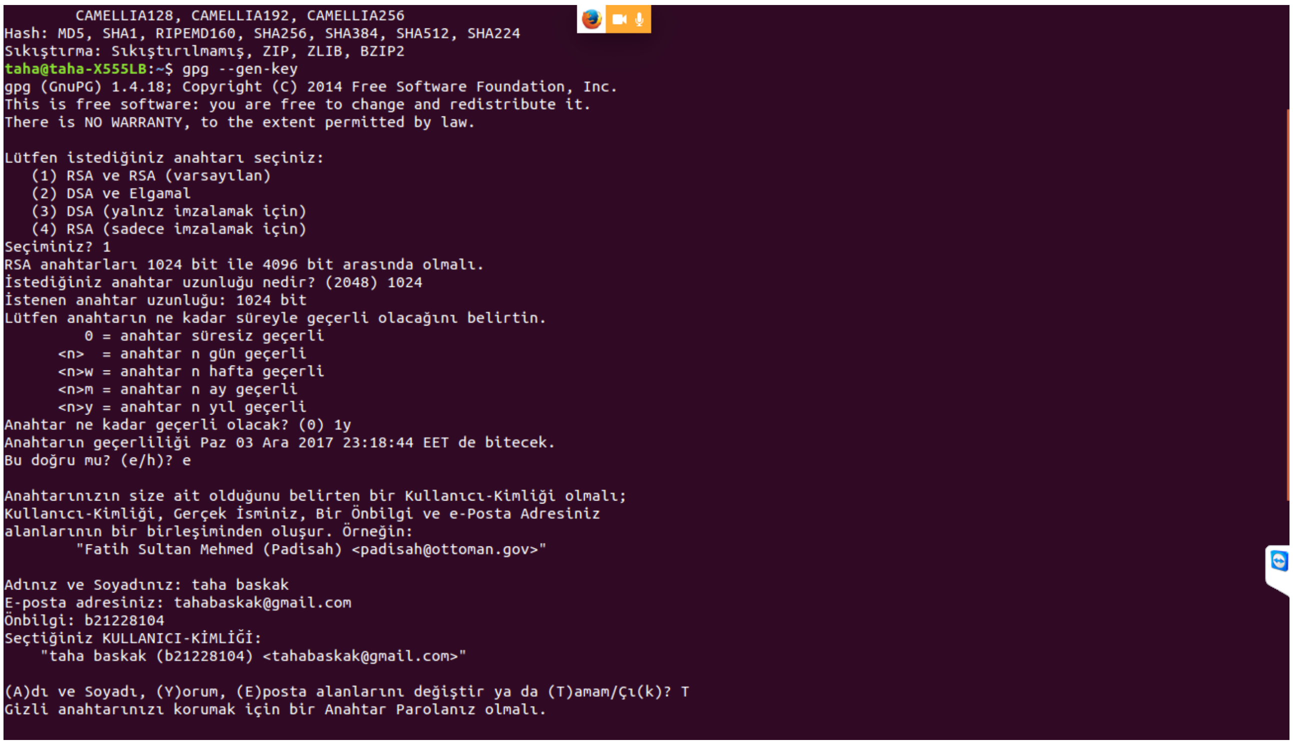


```
taha@taha-X555LB: ~
taha@taha-X555LB:~$ gpg --version
gpg (GnuPG) 1.4.18
Copyright (C) 2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: ~/.gnupg
Desteklenen algoritmalar:
GenAnah: RSA, RSA-E, RSA-S, ELG-E, DSA
Şifre: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
        CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Sıkıştırma: Sıkıştırılmamış, ZIP, ZLIB, BZIP2
taha@taha-X555LB:~$
```

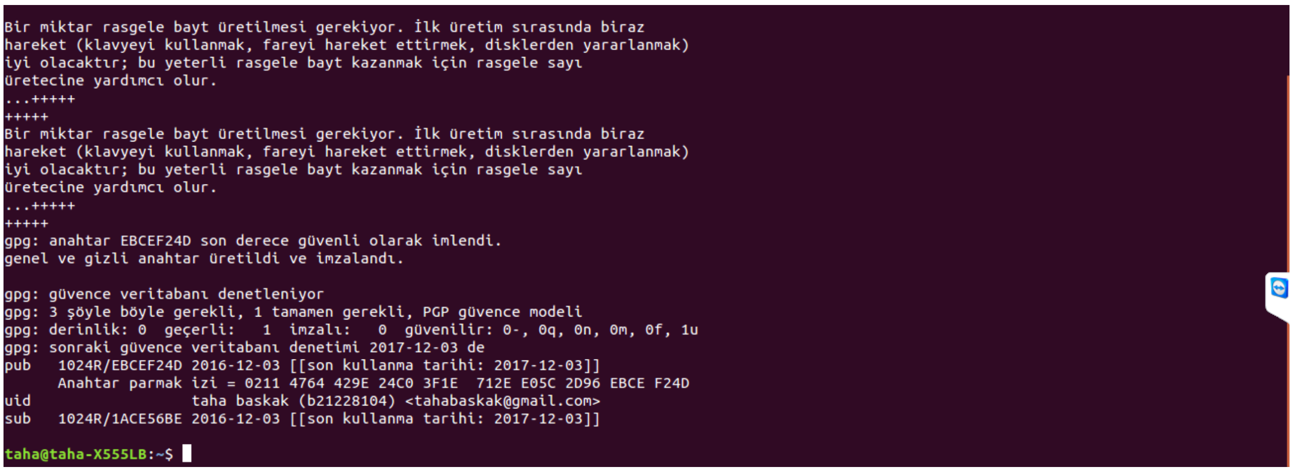
gpg --gen-key
#Generate a new key pair. Public and private key.
#There are options for the key pair to be created.
#These :

The type of algorithm to use to generate the keys : default option is 1
Key size : default size is 2048
Lifetime of the keys : default lifetime is indefinite
Verification question of lifetime of the keys : e or h
Name and Last Name
E-mail Address
Comment : Optional

#After all options if you approve then enter “T”
#And we must enter a passphrase to protect the secret key.
#Finally generate random byte to generate a new key pair but this operation done randomly therefore we can help using mouse or keyboard.

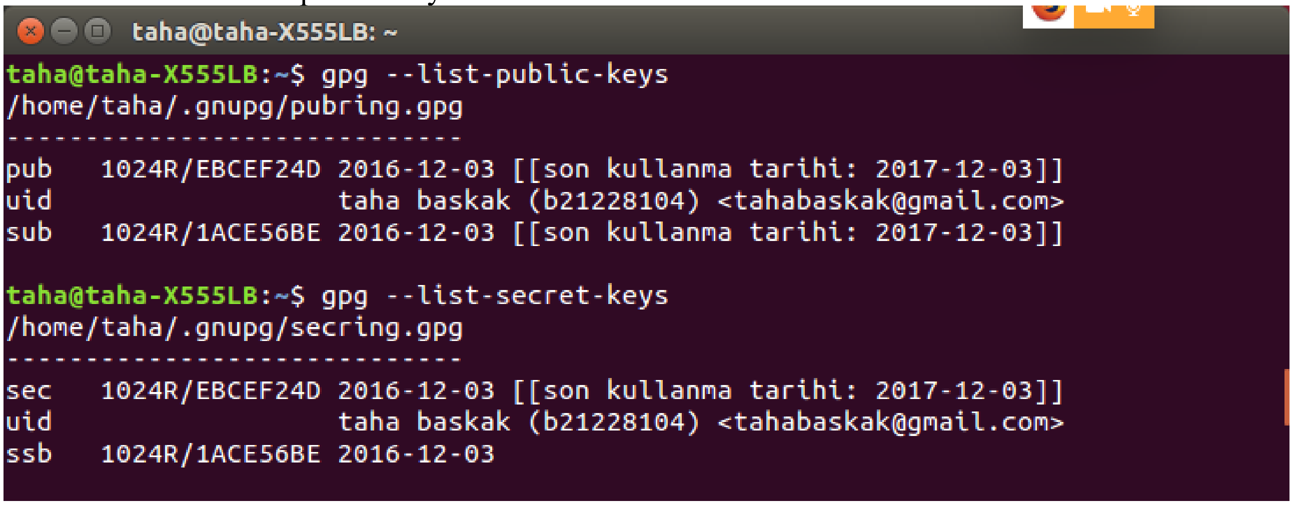


```
        CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Sıkıştırma: Sıkıştırılmamış, ZIP, ZLIB, BZIP2
taha@taha-X555LB:~$ gpg --gen-key
gpg (GnuPG) 1.4.18; Copyright (C) 2014 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Lütfen istediğiniz anahtarı seçiniz:
   (1) RSA ve RSA (varsayılan)
   (2) DSA ve Elgamal
   (3) DSA (yalnız imzalamak için)
   (4) RSA (sadece imzalamak için)
Seçiminiz? 1
RSA anahtarları 1024 bit ile 4096 bit arasında olmalı.
İstediğiniz anahtar uzunluğu nedir? (2048) 1024
İstenen anahtar uzunluğu: 1024 bit
Lütfen anahtarın ne kadar süreyle geçerli olacağını belirtin.
         0 = anahtar süresiz geçerli
      <n>  = anahtar n gün geçerli
      <n>w = anahtar n hafta geçerli
      <n>m = anahtar n ay geçerli
      <n>y = anahtar n yıl geçerli
Anahtar ne kadar geçerli olacak? (0) 1y
Anahtarın geçerliliği Paz 03 Ara 2017 23:18:44 EET de bitecek.
Bu doğru mu? (e/h)? e

Anahtarınızın size ait olduğunu belirten bir Kullanıcı-Kimliği olmalı;
Kullanıcı-Kimliği, Gerçek İsminiz, Bir Önbilgi ve e-Posta Adresiniz
alanlarının bir birleşiminden oluşur. Örneğin:
        "Fatih Sultan Mehmed (Padisah) <padisah@ottoman.gov>"

Adınız ve Soyadınız: taha baskak
E-posta adresiniz: tahabaskak@gmail.com
Önbilgi: b21228104
Seçtiğiniz KULLANICI-KİMLİĞİ:
    "taha baskak (b21228104) <tahabaskak@gmail.com>"

(A)dı ve Soyadı, (Y)orum, (E)posta alanlarını değiştir ya da (T)amam/Çı(k)? T
Gizli anahtarınızı korumak için bir Anahtar Parolanız olmalı.
```



```
Bir miktar rasgele bayt üretilmesi gerekiyor. İlk üretim sırasında biraz
hareket (klavyeyi kullanmak, fareyi hareket ettirmek, disklerden yararlanmak)
iyi olacaktır; bu yeterli rasgele bayt kazanmak için rasgele sayı
üretecine yardımcı olur.
...+++++
+++++
Bir miktar rasgele bayt üretilmesi gerekiyor. İlk üretim sırasında biraz
hareket (klavyeyi kullanmak, fareyi hareket ettirmek, disklerden yararlanmak)
iyi olacaktır; bu yeterli rasgele bayt kazanmak için rasgele sayı
üretecine yardımcı olur.
...+++++
+++++
gpg: anahtar EBCEF24D son derece güvenli olarak imlendi.
genel ve gizli anahtar üretildi ve imzalandı.

gpg: güvence veritabanı denetleniyor
gpg: 3 şöyle böyle gerekli, 1 tamamen gerekli, PGP güvence modeli
gpg: derinlik: 0  geçerli:   1  imzalı:   0  güvenilir: 0-, 0q, 0n, 0m, 0f, 1u
gpg: sonraki güvence veritabanı denetimi 2017-12-03 de
pub   1024R/EBCEF24D 2016-12-03 [[son kullanma tarihi: 2017-12-03]]
      Anahtar parmak izi = 0211 4764 429E 24C0 3F1E  712E E05C 2D96 EBCE F24D
uid                  taha baskak (b21228104) <tahabaskak@gmail.com>
sub   1024R/1ACE56BE 2016-12-03 [[son kullanma tarihi: 2017-12-03]]

taha@taha-X555LB:~$ 
```

gpg --list-public-keys
#We can see the list of public keys
gpg –list-secret-keys
#We can see the list of private keys



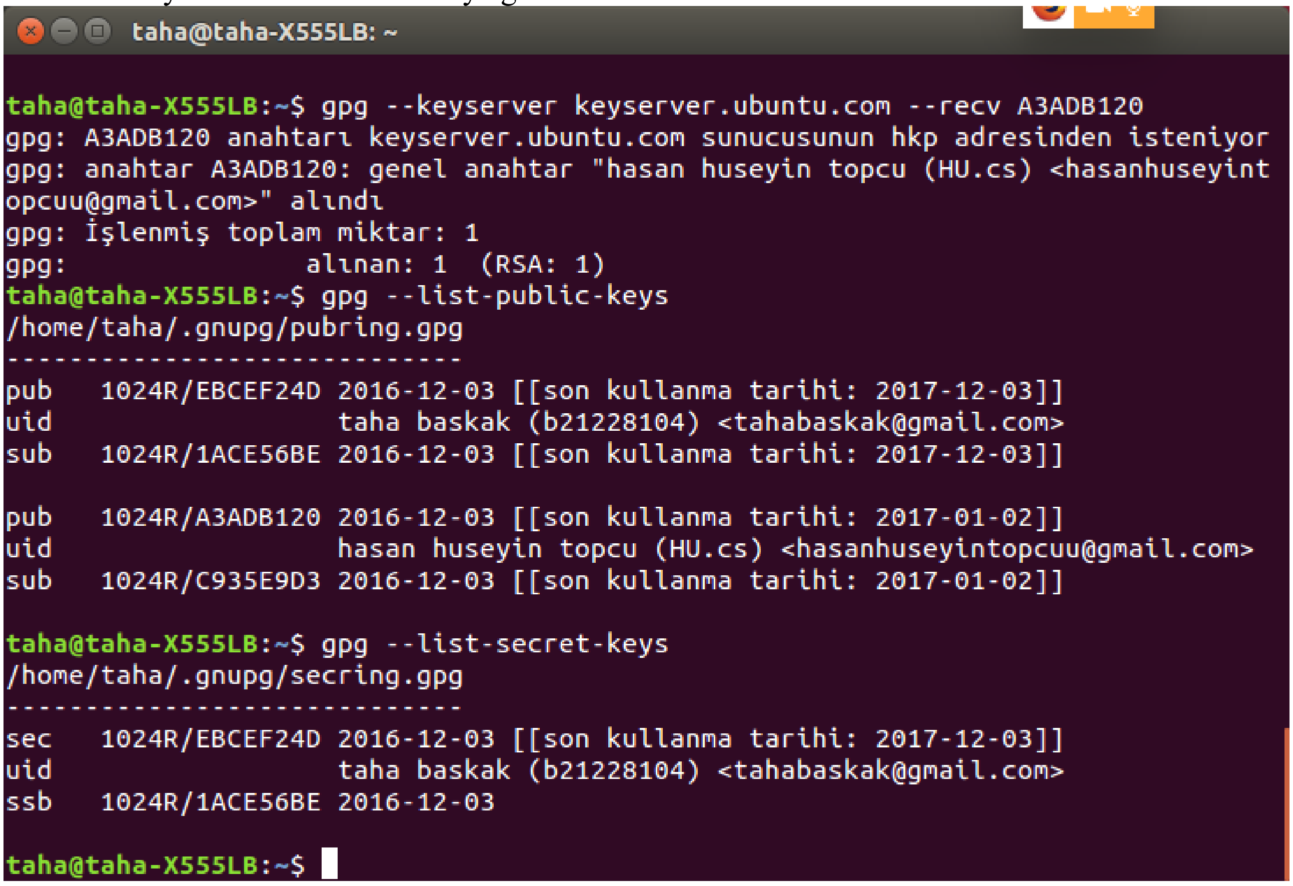
taha@taha-X555LB: ~

```
taha@taha-X555LB:~$ gpg --list-public-keys
/home/taha/.gnupg/pubring.gpg
-----------------------------
pub   1024R/EBCEF24D 2016-12-03 [[son kullanma tarihi: 2017-12-03]]
uid                  taha baskak (b21228104) <tahabaskak@gmail.com>
sub   1024R/1ACE56BE 2016-12-03 [[son kullanma tarihi: 2017-12-03]]

taha@taha-X555LB:~$ gpg --list-secret-keys
/home/taha/.gnupg/secring.gpg
-----------------------------
sec   1024R/EBCEF24D 2016-12-03 [[son kullanma tarihi: 2017-12-03]]
uid                  taha baskak (b21228104) <tahabaskak@gmail.com>
ssb   1024R/1ACE56BE 2016-12-03
```

gpg --send-keys --keyserver keyserver.ubuntu.com keyid
#Send public key with the xxxxxx id number with the last parameter to the keyserver.ubuntu.com
#Receiver peer must export own public key to sender peer.

gpg --keyserver keyserver.ubuntu.com --recv keyid
#Import public key with the xxxxxx id number with the last parameter from keyserver.ubuntu.com
#Receiver peer import public key from sender peer.
#To find the key id number of the sender's public key, we can go to any key server web site and find it by search email address or comment or date vs.

gpg --list-public-keys
#Now, we can see two public keys. Sender's public key and receiver's public key.
gpg –list-secret-keys
#We can only see our own secret key again.



```
taha@taha-X555LB: ~
taha@taha-X555LB:~$ gpg --keyserver keyserver.ubuntu.com --recv A3ADB120
gpg: A3ADB120 anahtarı keyserver.ubuntu.com sunucusunun hkp adresinden isteniyor
gpg: anahtar A3ADB120: genel anahtar "hasan huseyin topcu (HU.cs) <hasanhuseyint
opcuu@gmail.com>" alındı
gpg: İşlenmiş toplam miktar: 1
gpg:               alınan: 1  (RSA: 1)
taha@taha-X555LB:~$ gpg --list-public-keys
/home/taha/.gnupg/pubring.gpg
-----------------------------
pub   1024R/EBCEF24D 2016-12-03 [[son kullanma tarihi: 2017-12-03]]
uid                  taha baskak (b21228104) <tahabaskak@gmail.com>
sub   1024R/1ACE56BE 2016-12-03 [[son kullanma tarihi: 2017-12-03]]

pub   1024R/A3ADB120 2016-12-03 [[son kullanma tarihi: 2017-01-02]]
uid                  hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>
sub   1024R/C935E9D3 2016-12-03 [[son kullanma tarihi: 2017-01-02]]

taha@taha-X555LB:~$ gpg --list-secret-keys
/home/taha/.gnupg/secring.gpg
-----------------------------
sec   1024R/EBCEF24D 2016-12-03 [[son kullanma tarihi: 2017-12-03]]
uid                  taha baskak (b21228104) <tahabaskak@gmail.com>
ssb   1024R/1ACE56BE 2016-12-03

taha@taha-X555LB:~$ 
```

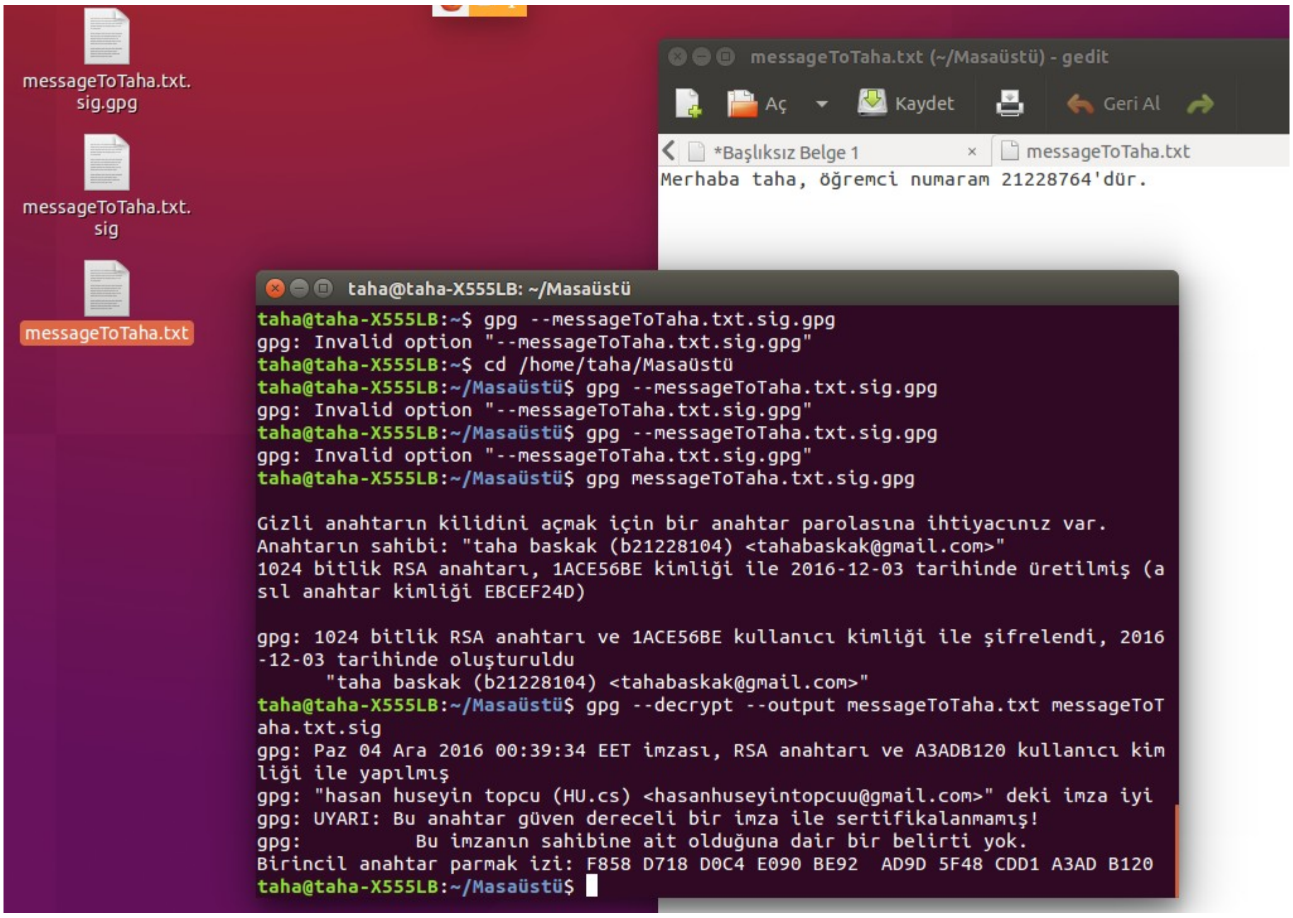
cd /home/taha/Masaüstü
#we change directory to just be easy. There is not necessary. If not change directory you must be careful for the files to be created and the directories of the files given as parameters for the next commands.

Decrypt and Verify Unique Message Part(messageToTaha.txt File)

Note : Receiver peer take encrypted file and copy in /home/taha/Masaüstü folder.

gpg messageToTaha.txt.sig.gpg
#Decrypt file with gpg comand.
#Sender encrypted using receiver's public key so receiver decrypt via own private key.
#When receiver enter right passphrase then created messageToTaha.txt.sig file in current directory.

gpg –decrypt –output messageToTaha.txt message.txt.sig
#Finally, receiver must verify message. Thats mean are message integrity and sender authentication.
#Sender sign using own private key so receiver verify using sender's public key.
#If verify is succesful then created messageToTaha.txt file in current dictory.


messageToTaha.txt.sig.gpg
messageToTaha.txt.sig
messageToTaha.txt
messageToTaha.txt (~/Masaüstü) - gedit
Aç
Kaydet
Geri Al
*Başlıksız Belge 1
messageToTaha.txt
taha@taha-X555LB: ~/Masaüstü

```
Merhaba taha, öğremci numaram 21228764'dür.
```

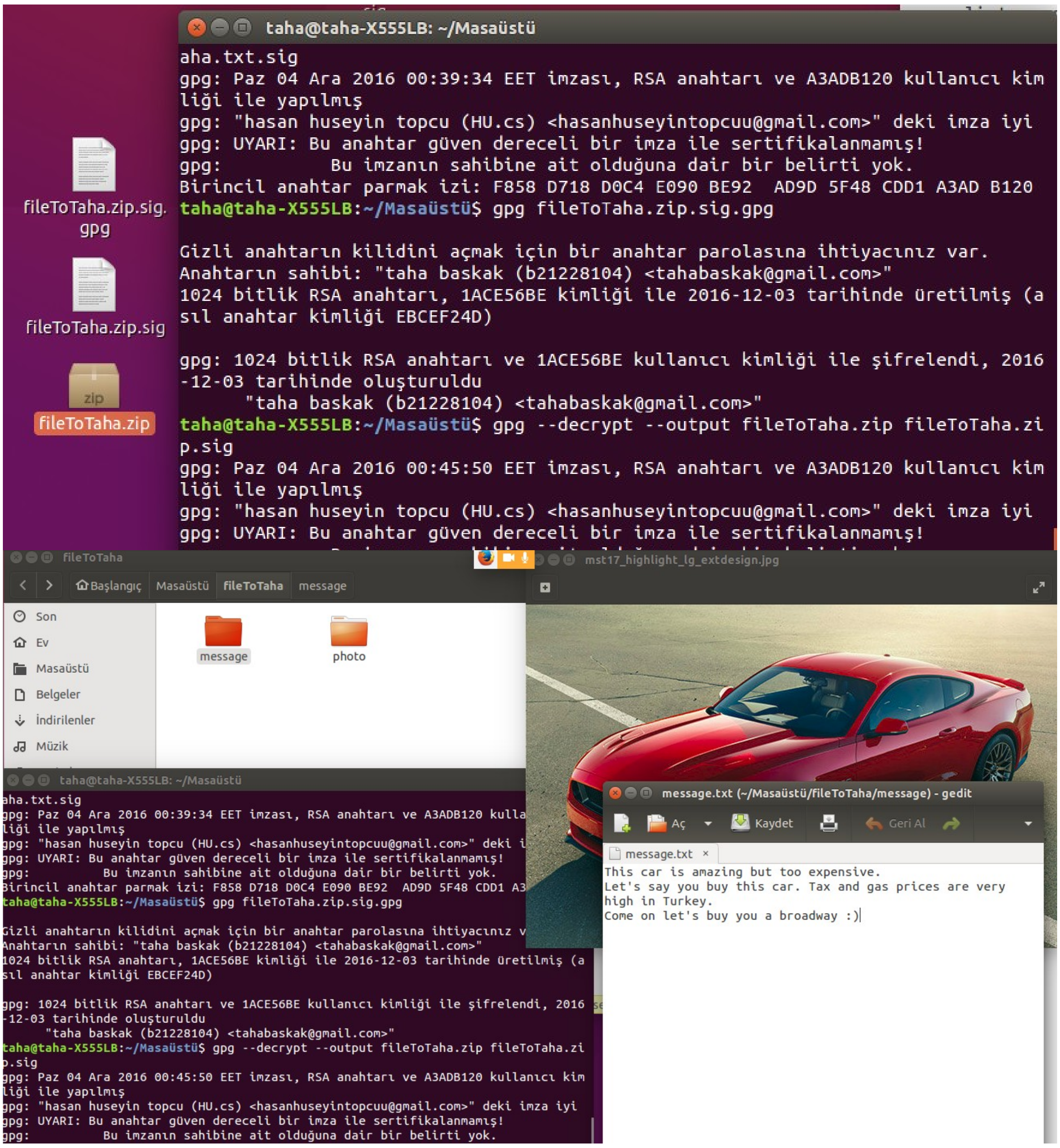
```
taha@taha-X555LB:~$ gpg --messageToTaha.txt.sig.gpg
gpg: Invalid option "--messageToTaha.txt.sig.gpg"
taha@taha-X555LB:~$ cd /home/taha/Masaüstü
taha@taha-X555LB:~/Masaüstü$ gpg --messageToTaha.txt.sig.gpg
gpg: Invalid option "--messageToTaha.txt.sig.gpg"
taha@taha-X555LB:~/Masaüstü$ gpg --messageToTaha.txt.sig.gpg
gpg: Invalid option "--messageToTaha.txt.sig.gpg"
taha@taha-X555LB:~/Masaüstü$ gpg messageToTaha.txt.sig.gpg

Gizli anahtarın kilidini açmak için bir anahtar parolasına ihtiyacınız var.
Anahtarın sahibi: "taha baskak (b21228104) <tahabaskak@gmail.com>"
1024 bitlik RSA anahtarı, 1ACE56BE kimliği ile 2016-12-03 tarihinde üretilmiş (a
sıl anahtar kimliği EBCEF24D)

gpg: 1024 bitlik RSA anahtarı ve 1ACE56BE kullanıcı kimliği ile şifrelendi, 2016
-12-03 tarihinde oluşturuldu
      "taha baskak (b21228104) <tahabaskak@gmail.com>"
taha@taha-X555LB:~/Masaüstü$ gpg --decrypt --output messageToTaha.txt messageToT
aha.txt.sig
gpg: Paz 04 Ara 2016 00:39:34 EET imzası, RSA anahtarı ve A3ADB120 kullanıcı kim
liği ile yapılmış
gpg: "hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>" deki imza iyi
gpg: UYARI: Bu anahtar güven dereceli bir imza ile sertifikalanmamış!
gpg:          Bu imzanın sahibine ait olduğuna dair bir belirti yok.
Birincil anahtar parmak izi: F858 D718 D0C4 E090 BE92  AD9D 5F48 CDD1 A3AD B120
taha@taha-X555LB:~/Masaüstü$
```

Decrypt and Verify Unique Zip File Part(fileToTaha.zip File)

Note : Receiver peer take encrypted file and copy in /home/taha/Masaüstü folder.

gpg fileToTaha.zip.sig.gpg
#Decrypt zip file with gpg comand.
#Sender encrypted using receiver's public key so receiver decrypt via own private key.
#When receiver enter right passphrase then created fileToTaha.zip.sig file in current directory.

gpg –decrypt –output fileToTaha.zip fileToTaha.zip.sig
#Finally, receiver must verify zip file. Thats mean are file integrity and sender authentication.
#Sender sign using own private key so receiver verify using sender's public key.
#If verify is succesful then created fileToTaha.zip file in current dictory.



taha@taha-X555LB: ~/Masaüstü

```
aha.txt.sig
gpg: Paz 04 Ara 2016 00:39:34 EET imzası, RSA anahtarı ve A3ADB120 kullanıcı kim
liği ile yapılmış
gpg: "hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>" deki imza iyi
gpg: UYARI: Bu anahtar güven dereceli bir imza ile sertifikalanmamış!
gpg:          Bu imzanın sahibine ait olduğuna dair bir belirti yok.
Birincil anahtar parmak izi: F858 D718 D0C4 E090 BE92  AD9D 5F48 CDD1 A3AD B120
taha@taha-X555LB:~/Masaüstü$ gpg fileToTaha.zip.sig.gpg

Gizli anahtarın kilidini açmak için bir anahtar parolasına ihtiyacınız var.
Anahtarın sahibi: "taha baskak (b21228104) <tahabaskak@gmail.com>"
1024 bitlik RSA anahtarı, 1ACE56BE kimliği ile 2016-12-03 tarihinde üretilmiş (a
sıl anahtar kimliği EBCEF24D)

gpg: 1024 bitlik RSA anahtarı ve 1ACE56BE kullanıcı kimliği ile şifrelendi, 2016
-12-03 tarihinde oluşturuldu
      "taha baskak (b21228104) <tahabaskak@gmail.com>"
taha@taha-X555LB:~/Masaüstü$ gpg --decrypt --output fileToTaha.zip fileToTaha.zi
p.sig
gpg: Paz 04 Ara 2016 00:45:50 EET imzası, RSA anahtarı ve A3ADB120 kullanıcı kim
liği ile yapılmış
gpg: "hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>" deki imza iyi
gpg: UYARI: Bu anahtar güven dereceli bir imza ile sertifikalanmamış!
```

taha@taha-X555LB: ~/Masaüstü

```
aha.txt.sig
gpg: Paz 04 Ara 2016 00:39:34 EET imzası, RSA anahtarı ve A3ADB120 kulla
liği ile yapılmış
gpg: "hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>" deki i
gpg: UYARI: Bu anahtar güven dereceli bir imza ile sertifikalanmamış!
gpg:          Bu imzanın sahibine ait olduğuna dair bir belirti yok.
Birincil anahtar parmak izi: F858 D718 D0C4 E090 BE92  AD9D 5F48 CDD1 A3
taha@taha-X555LB:~/Masaüstü$ gpg fileToTaha.zip.sig.gpg

Gizli anahtarın kilidini açmak için bir anahtar parolasına ihtiyacınız v
Anahtarın sahibi: "taha baskak (b21228104) <tahabaskak@gmail.com>"
1024 bitlik RSA anahtarı, 1ACE56BE kimliği ile 2016-12-03 tarihinde üretilmiş (a
sıl anahtar kimliği EBCEF24D)

gpg: 1024 bitlik RSA anahtarı ve 1ACE56BE kullanıcı kimliği ile şifrelendi, 2016
-12-03 tarihinde oluşturuldu
      "taha baskak (b21228104) <tahabaskak@gmail.com>"
taha@taha-X555LB:~/Masaüstü$ gpg --decrypt --output fileToTaha.zip fileToTaha.zi
p.sig
gpg: Paz 04 Ara 2016 00:45:50 EET imzası, RSA anahtarı ve A3ADB120 kullanıcı kim
liği ile yapılmış
gpg: "hasan huseyin topcu (HU.cs) <hasanhuseyintopcuu@gmail.com>" deki imza iyi
gpg: UYARI: Bu anahtar güven dereceli bir imza ile sertifikalanmamış!
gpg:          Bu imzanın sahibine ait olduğuna dair bir belirti yok.
```

message.txt (~/Masaüstü/fileToTaha/message) - gedit

message.txt

```
This car is amazing but too expensive.
Let's say you buy this car. Tax and gas prices are very
high in Turkey.
Come on let's buy you a broadway :)
```

SMIME

/ Sender peer */*
Sender Key File : hasankey.pem
Sender Certificate File : hasancert.pem

<u>Generate Certificate Authority(CA) With Openssl Command</u>

su
#be root user.

mkdir /root/CA
#create CA file in "/root" path.

cd /root/CA
#enter "/root/CA" path.

mkdir newcerts certs crl private requests
#create newcerts, certs, crl, private and requests folders.

touch index.txt
echo '01' > serial
#create some necessary files for Certification Authority.

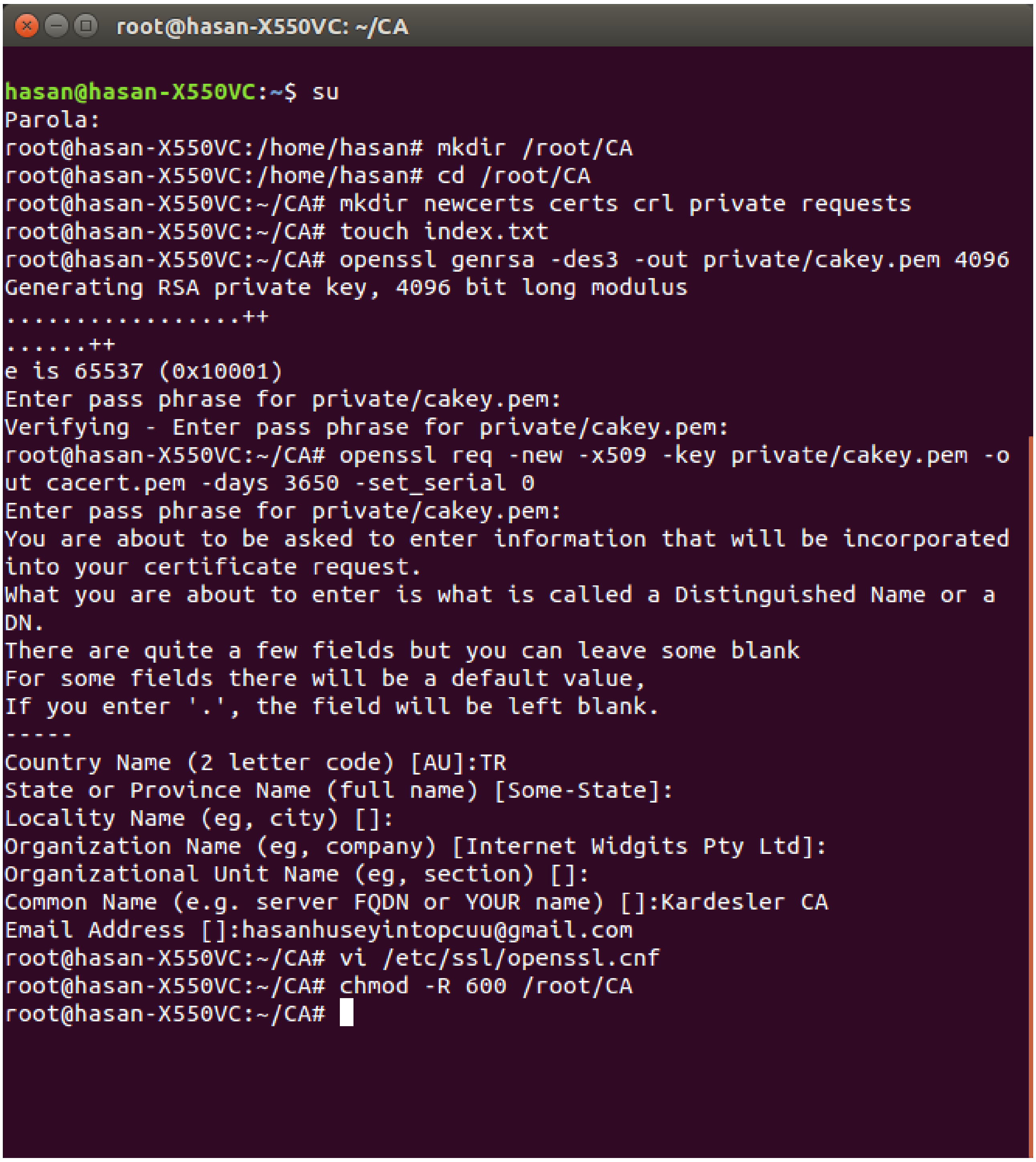
openssl genrsa -des3 -out private/cakey.pem 4096
#generate CA's private key using openssl.
#used encryption algorithm is RSA and des3 and 4096 bit long.
#created cakey.pem file is under private folder.
#want to password from us and enter password for cakey.pem

openssl req -new -x509 -key private/cakey.pem -out cacert.pem -days 3650 -set_serial 0
#use the root key (cakey.pem) to create a root certificate (cacert.pem) using x509 and openssl.
#the root certificate expiry date is 3650 days.
#once the root certificate expires, all certificates signed by the CA become invalid.
#and root CA informations are these:

> *Country name = TR*
> *common name = Kardesler CA*
> *email = hasanhuseyintopcuu@gmail.com*
> *others are optional so we enter nothing its empty*

vi /etc/ssl/openssl.cnf
#enter openssl.cnf file and edit "dir" line. Make "dir= /root/CA".

chmod -R 600 /root/CA
#limit access rights



```
root@hasan-X550VC: ~/CA

hasan@hasan-X550VC:~$ su
Parola:
root@hasan-X550VC:/home/hasan# mkdir /root/CA
root@hasan-X550VC:/home/hasan# cd /root/CA
root@hasan-X550VC:~/CA# mkdir newcerts certs crl private requests
root@hasan-X550VC:~/CA# touch index.txt
root@hasan-X550VC:~/CA# openssl genrsa -des3 -out private/cakey.pem 4096
Generating RSA private key, 4096 bit long modulus
.................++
......++
e is 65537 (0x10001)
Enter pass phrase for private/cakey.pem:
Verifying - Enter pass phrase for private/cakey.pem:
root@hasan-X550VC:~/CA# openssl req -new -x509 -key private/cakey.pem -o
ut cacert.pem -days 3650 -set_serial 0
Enter pass phrase for private/cakey.pem:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a
DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:TR
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:Kardesler CA
Email Address []:hasanhuseyintopcuu@gmail.com
root@hasan-X550VC:~/CA# vi /etc/ssl/openssl.cnf
root@hasan-X550VC:~/CA# chmod -R 600 /root/CA
root@hasan-X550VC:~/CA#
```

<u>Derive certificate from generated CA for sender peer</u>

cd /root/CA/requests
#enter "/root/CA/request" path.

openssl genrsa -des3 -out hasankey.pem 2048
#generate hasankey.pem using openssl.
#used encryption algorithm is RSA and des3 and 2048 bit long.

openssl req -new -key hasankey.pem -out hasancert.csr -days 3650
#use hasankey.pem to create certificate request that is hasancert.crs for 3650 days.

#enter these informations:

*password = *******

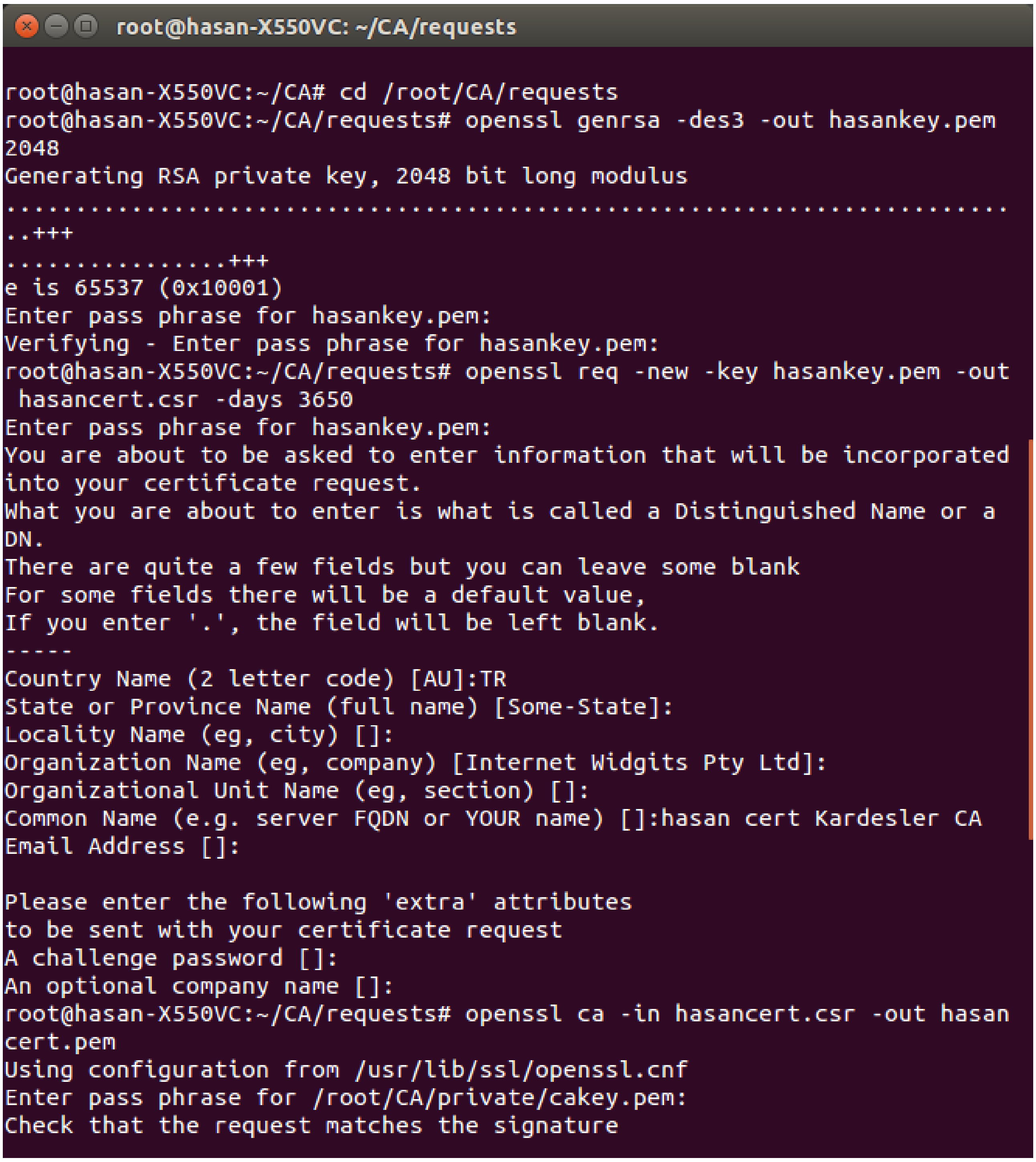
country name = TR

comman name = hasan cert Kardeşler CA

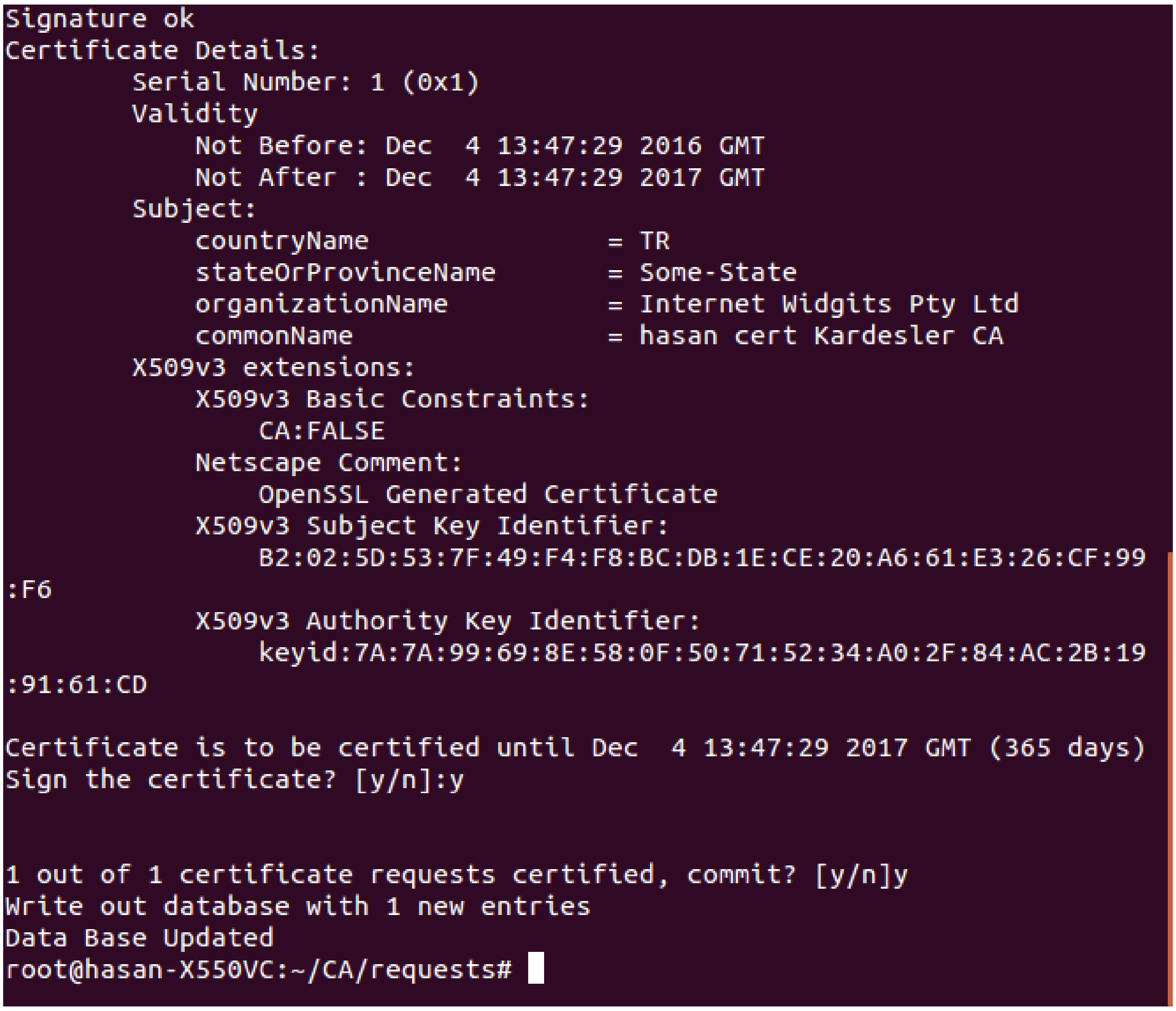
others are optional so we enter nothing its empty

openssl ca -in hasancert.csr -out hasancert.pem

#sign certificate(hasancert.pem) using hasancert.csr file and of course ca sign it.



```
root@hasan-X550VC:~/CA# cd /root/CA/requests
root@hasan-X550VC:~/CA/requests# openssl genrsa -des3 -out hasankey.pem
2048
Generating RSA private key, 2048 bit long modulus
.......................................................................
..+++
................+++
e is 65537 (0x10001)
Enter pass phrase for hasankey.pem:
Verifying - Enter pass phrase for hasankey.pem:
root@hasan-X550VC:~/CA/requests# openssl req -new -key hasankey.pem -out
 hasancert.csr -days 3650
Enter pass phrase for hasankey.pem:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a
DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:TR
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:hasan cert Kardesler CA
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
root@hasan-X550VC:~/CA/requests# openssl ca -in hasancert.csr -out hasan
cert.pem
Using configuration from /usr/lib/ssl/openssl.cnf
Enter pass phrase for /root/CA/private/cakey.pem:
Check that the request matches the signature
```



```
Signature ok
Certificate Details:
        Serial Number: 1 (0x1)
        Validity
            Not Before: Dec  4 13:47:29 2016 GMT
            Not After : Dec  4 13:47:29 2017 GMT
        Subject:
            countryName               = TR
            stateOrProvinceName       = Some-State
            organizationName          = Internet Widgits Pty Ltd
            commonName                = hasan cert Kardesler CA
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                B2:02:5D:53:7F:49:F4:F8:BC:DB:1E:CE:20:A6:61:E3:26:CF:99
:F6
            X509v3 Authority Key Identifier:
                keyid:7A:7A:99:69:8E:58:0F:50:71:52:34:A0:2F:84:AC:2B:19
:91:61:CD

Certificate is to be certified until Dec  4 13:47:29 2017 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
root@hasan-X550VC:~/CA/requests#
```

Note : Copied hasancert.pem certificate to receiver peer and copied tahacert.pem certificate to sender peer.

<u>Sign and Encrypt Unique Message(secretMessage.txt)</u>

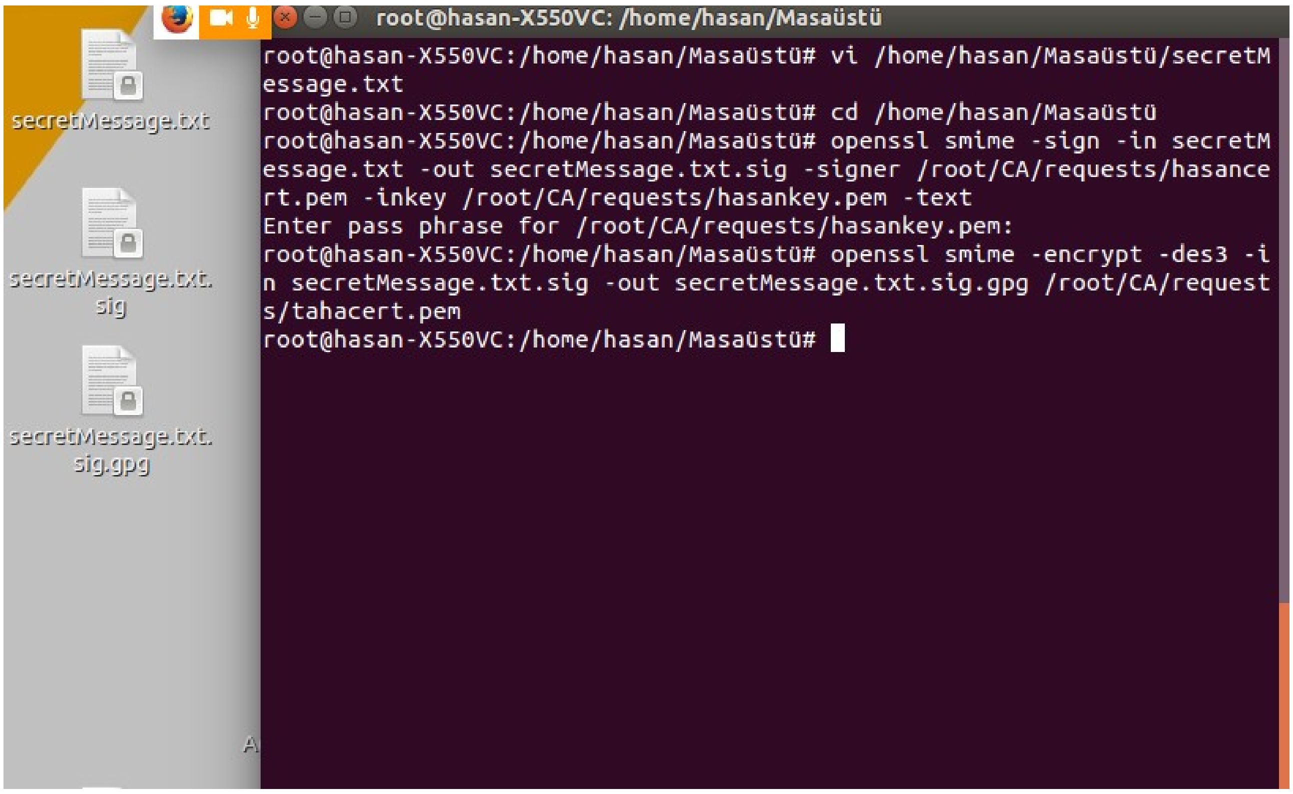
vi /home/hasan/Masaüstü/secretMessage.txt
#create secretMessage.txt file using vi editor in “/home/hasan/Masaüstü” folder.

cd /home/hasan/Masaüstü
#enter “/home/hasan/Masaüstü” folder. Now every file that will occur after this will occur in this path. This command is not required. It was made just for convenience.

openssl smime -sign -in secretMessage.txt -out secretMessage.txt.sig -signer /root/CA/requests/hasancert.pem -inkey /root/CA/requests/hasankey.pem
#The sender's key and certificate are used in the signing process.
#Sign(via -sign command) secretMessage.txt file(given -in command).
#Signer is hasancert.pem(via -signer command) and the key to be used is hasankey.pem(via -inkey command).
#Enter password that have hasankey.pem file and if it is correct then file is signed so created secretMessage.txt.sig file incurrent path.

openssl smime -encrypt -des3 -in secretMessage.txt.sig -out secretMessage.txt.sig.gpg /root/CA/requests/tahacert.pem

#The receiver's certificate is used in the encryption process.
#Encrypt(via -encrypt command) secretMessage.txt.sig(via -in command) file using des3 algortihm.
#Use receiver's certificate that is last parameter
#Finally, create secretMessage.txt.sig.gpg file via -out command in current path.



root@hasan-X550VC: /home/hasan/Masaüstü
secretMessage.txt
secretMessage.txt.sig
secretMessage.txt.sig.gpg

```
root@hasan-X550VC:/home/hasan/Masaüstü# vi /home/hasan/Masaüstü/secretM
essage.txt
root@hasan-X550VC:/home/hasan/Masaüstü# cd /home/hasan/Masaüstü
root@hasan-X550VC:/home/hasan/Masaüstü# openssl smime -sign -in secretM
essage.txt -out secretMessage.txt.sig -signer /root/CA/requests/hasance
rt.pem -inkey /root/CA/requests/hasankey.pem -text
Enter pass phrase for /root/CA/requests/hasankey.pem:
root@hasan-X550VC:/home/hasan/Masaüstü# openssl smime -encrypt -des3 -i
n secretMessage.txt.sig -out secretMessage.txt.sig.gpg /root/CA/request
s/tahacert.pem
root@hasan-X550VC:/home/hasan/Masaüstü#
```

/ Receiver Peer */*
Sender Key File : tahakey.pem
Sender Certificate File : tahacert.pem

<u>Generate Certificate Authority(CA) With Openssl Command</u>

su
#be root user.

mkdir /root/CA
#create CA file in “/root” path.

cd /root/CA
#enter “/root/CA” path.

mkdir newcerts certs crl private requests
#create newcerts, certs, crl, private and requests folders.

touch index.txt
echo '01' > serial
#create some necessary files for Certification Authority.

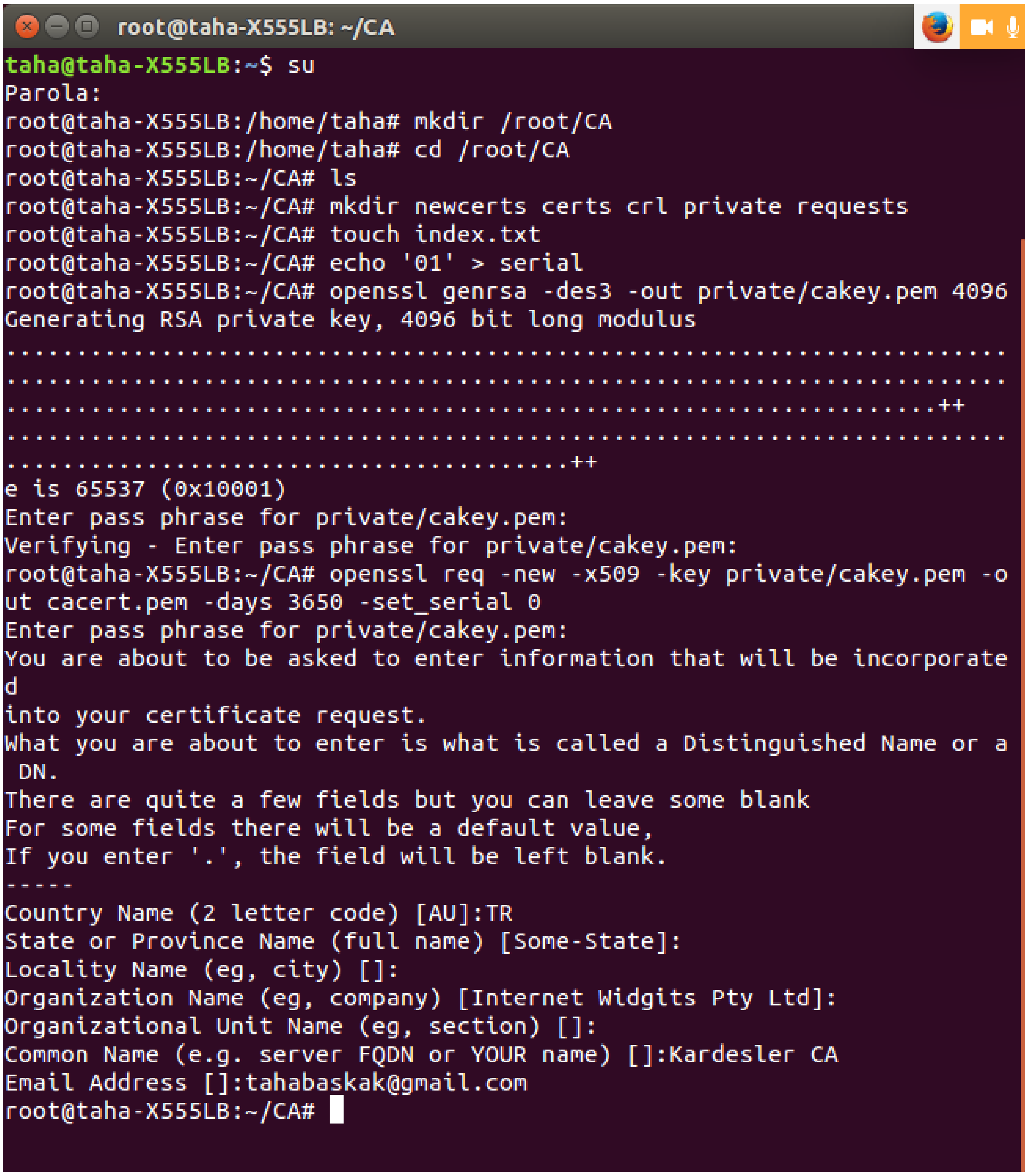
openssl genrsa -des3 -out private/cakey.pem 4096
#generate CA's private key using openssl.
#used encryption algorithm is RSA and des3 and 4096 bit long.
#created cakey.pem file is under private folder.
#want to password from us and enter password for cakey.pem

openssl req -new -x509 -key private/cakey.pem -out cacert.pem -days 3650 -set_serial 0
#use the root key (cakey.pem) to create a root certificate (cacert.pem) using x509 and openssl.
#the root certificate expiry date is 3650 days.
#once the root certificate expires, all certificates signed by the CA become invalid.
#and root CA informations are these:

> *Country name = TR*
> *common name = Kardesler CA*
> *email = tahabaskak@gmail.com*
> *others are optional so we enter nothing its empty*

vi /etc/ssl/openssl.cnf
#enter openssl.cnf file and edit “dir” line. Make “dir= /root/CA”.

chmod -R 600 /root/CA
#limit access rights



```
taha@taha-X555LB:~$ su
Parola:
root@taha-X555LB:/home/taha# mkdir /root/CA
root@taha-X555LB:/home/taha# cd /root/CA
root@taha-X555LB:~/CA# ls
root@taha-X555LB:~/CA# mkdir newcerts certs crl private requests
root@taha-X555LB:~/CA# touch index.txt
root@taha-X555LB:~/CA# echo '01' > serial
root@taha-X555LB:~/CA# openssl genrsa -des3 -out private/cakey.pem 4096
Generating RSA private key, 4096 bit long modulus
.......................................................................
.......................................................................
...................................................................++
.......................................................................
........................................++
e is 65537 (0x10001)
Enter pass phrase for private/cakey.pem:
Verifying - Enter pass phrase for private/cakey.pem:
root@taha-X555LB:~/CA# openssl req -new -x509 -key private/cakey.pem -o
ut cacert.pem -days 3650 -set_serial 0
Enter pass phrase for private/cakey.pem:
You are about to be asked to enter information that will be incorporate
d
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a
 DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:TR
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:Kardesler CA
Email Address []:tahabaskak@gmail.com
root@taha-X555LB:~/CA# 
```

Derive certificate from generated CA for receiver peer

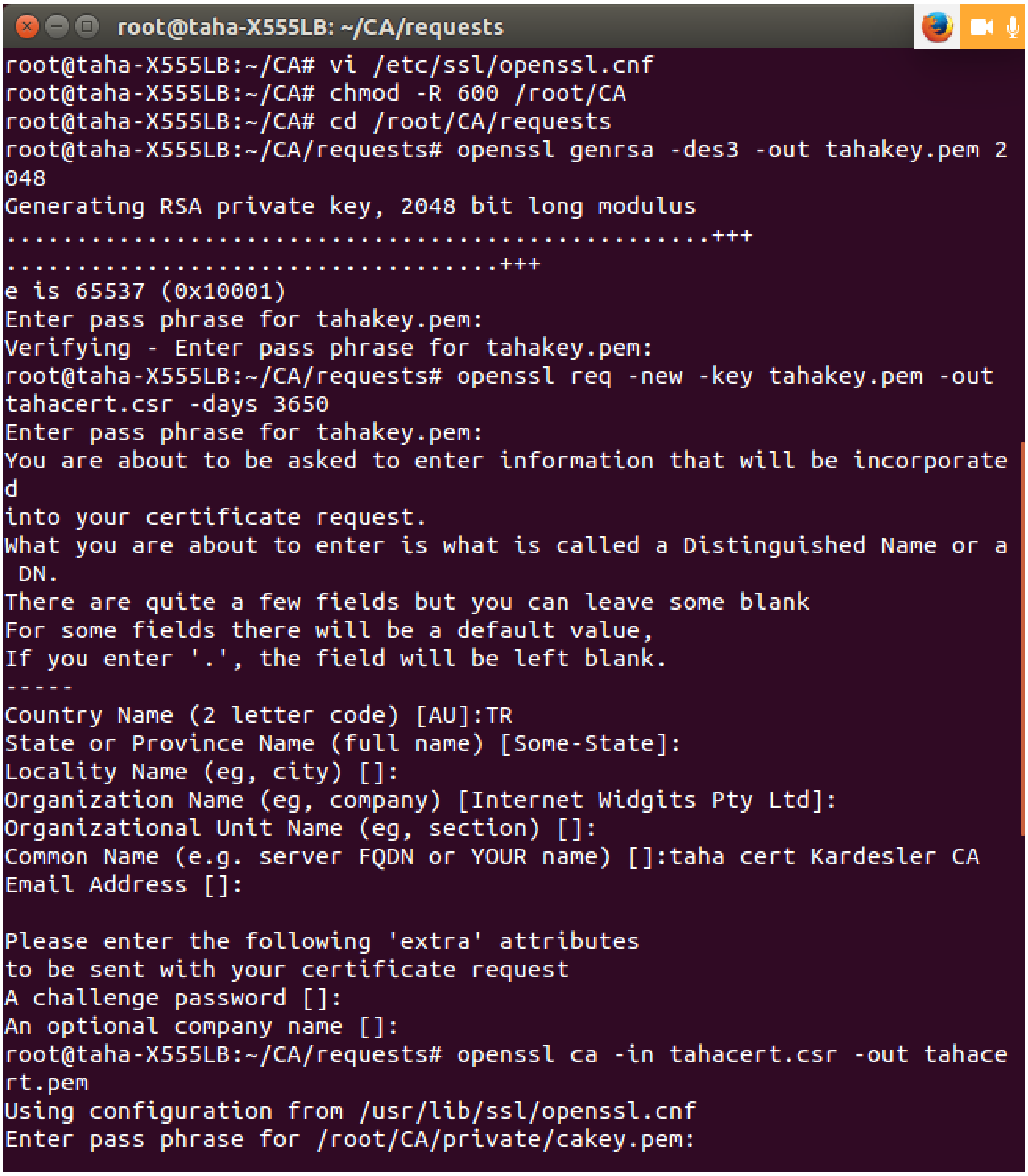
cd /root/CA/requests
#enter “/root/CA/request” path.

openssl genrsa -des3 -out tahakey.pem 2048
#generate tahakey.pem using openssl.
#used encryption algorithm is RSA and des3 and 2048 bit long.

openssl req -new -key tahakey.pem -out tahacert.csr -days 3650
#use tahakey.pem to create certificate request that is tahacert.crs for 3650 days.
#enter these informations:

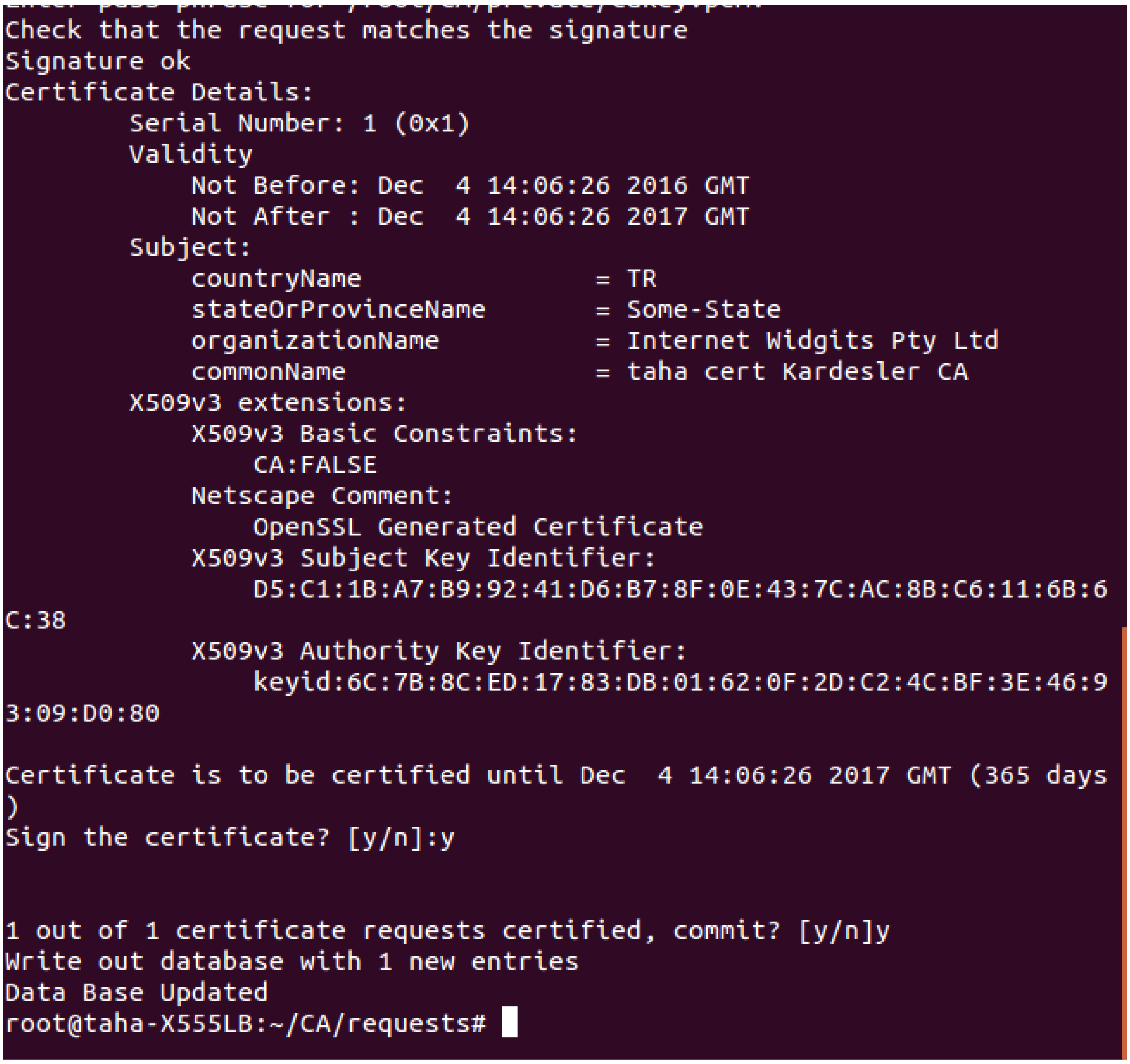
*password = *******
country name = TR
comman name = taha cert Kardeşler CA
others are optional so we enter nothing its empty

openssl ca -in tahacert.csr -out tahacert.pem
#sign certificate(tahacert.pem) using tahacert.csr file and of course ca sign it.



root@taha-X555LB: ~/CA/requests

```
root@taha-X555LB:~/CA# vi /etc/ssl/openssl.cnf
root@taha-X555LB:~/CA# chmod -R 600 /root/CA
root@taha-X555LB:~/CA# cd /root/CA/requests
root@taha-X555LB:~/CA/requests# openssl genrsa -des3 -out tahakey.pem 2
048
Generating RSA private key, 2048 bit long modulus
.................................................+++
...................................+++
e is 65537 (0x10001)
Enter pass phrase for tahakey.pem:
Verifying - Enter pass phrase for tahakey.pem:
root@taha-X555LB:~/CA/requests# openssl req -new -key tahakey.pem -out
tahacert.csr -days 3650
Enter pass phrase for tahakey.pem:
You are about to be asked to enter information that will be incorporate
d
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a
 DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:TR
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:taha cert Kardesler CA
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
root@taha-X555LB:~/CA/requests# openssl ca -in tahacert.csr -out tahace
rt.pem
Using configuration from /usr/lib/ssl/openssl.cnf
Enter pass phrase for /root/CA/private/cakey.pem:
```



```
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 1 (0x1)
        Validity
            Not Before: Dec  4 14:06:26 2016 GMT
            Not After : Dec  4 14:06:26 2017 GMT
        Subject:
            countryName               = TR
            stateOrProvinceName       = Some-State
            organizationName          = Internet Widgits Pty Ltd
            commonName                = taha cert Kardesler CA
        X509v3 extensions:
            X509v3 Basic Constraints: 
                CA:FALSE
            Netscape Comment: 
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier: 
                D5:C1:1B:A7:B9:92:41:D6:B7:8F:0E:43:7C:AC:8B:C6:11:6B:6
C:38
            X509v3 Authority Key Identifier: 
                keyid:6C:7B:8C:ED:17:83:DB:01:62:0F:2D:C2:4C:BF:3E:46:9
3:09:D0:80

Certificate is to be certified until Dec  4 14:06:26 2017 GMT (365 days
)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
root@taha-X555LB:~/CA/requests#
```

Note : Copied hasancert.pem certificate to receiver peer and copied tahacert.pem certificate to sender peer.

<u>Decrypt and Verify Unique Message(secretMessage.txt.sig.gpg)</u>

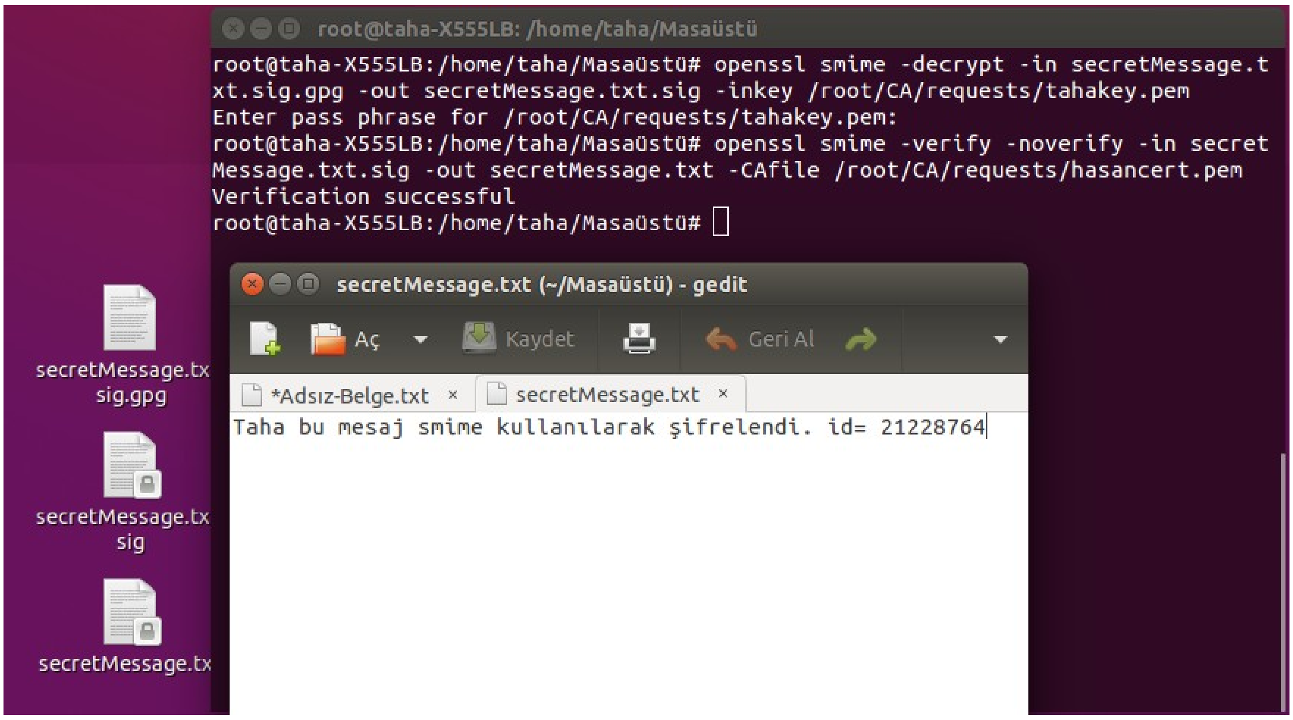
Note : Receiver peer received secretMessage.txt.sig.gpg file and copy “/home/taha/Masaüstü” directory.

cd /home/taha/Masaüstü
#enter “/home/taha/Masaüstü” path. Now every file that will occur after this will occur in this path. This command is not required. It was made just for convenience.

openssl smime -decrypt -in secretMessage.txt.sig.gpg -out secretMessage.txt.sig -inkey /root/CA/requests/tahakey.pem
#The receiver key is used in the decryption process because file is encrypted with its own certificate that created using own key file.
#Decrypt via -decrypt command and input file is given -in command, output file is given -out command.

#The key to be used in decryption is given in the last parameter via -inkey command.

openssl smime -verify -noverify -in secretMessage.txt.sig -out secretMessage.txt -CAfile /root/CA/requests/hasancert.pem
#Finally, the verification is done with the certificate received from the sender.
#Use sender's certificate to verify file because this file is signed sender's key.
#We use -verify commad with -noverify. This is look a bit weird but to suppress the checking of the key certificate when verifying a file we can supply the -noverify parameter to the verify command
#If veridication is succesful then create secretMessage.txt file in current path via –out command.



```
root@taha-X555LB:/home/taha/Masaüstü# openssl smime -decrypt -in secretMessage.t
xt.sig.gpg -out secretMessage.txt.sig -inkey /root/CA/requests/tahakey.pem
Enter pass phrase for /root/CA/requests/tahakey.pem:
root@taha-X555LB:/home/taha/Masaüstü# openssl smime -verify -noverify -in secret
Message.txt.sig -out secretMessage.txt -CAfile /root/CA/requests/hasancert.pem
Verification successful
root@taha-X555LB:/home/taha/Masaüstü#
```

```
Taha bu mesaj smime kullanılarak şifrelendi. id= 21228764
```

References

PGP

[1] https://tr.wikipedia.org/wiki/Pretty_Good_Privacy

http://bilgisayarkavramlari.sadievrenseker.com/2012/03/22/pgp-pretty-good-privacy/

https://www.bilgiguvenligi.gov.tr/gizlilik/pretty-good-privacy-pgp-sifreleme.html

http://www.pgpi.org/doc/pgpintro/

http://crypto.stackexchange.com/questions/5458/should-we-sign-then-encrypt-or-encrypt-then-sign

https://www.digitalocean.com/community/tutorials/how-to-use-gpg-to-encrypt-and-sign-messages-on-an-ubuntu-12-04-vps#how-to-verify-and-sign-keys

https://www.math.utah.edu/~beebe/PGP-notes.html#key-servers

https://help.ubuntu.com/community/GnuPrivacyGuardHowto#Signing_Data

http://keyserver.ubuntu.com/

https://www.gnupg.org/gph/en/manual/x110.html

https://www.gnupg.org/documentation/manpage.html

https://futureboy.us/pgp.html#SigningMessages

http://crypto.stackexchange.com/questions/5458/should-we-sign-then-encrypt-or-encrypt-then-sign

SMIME

https://technet.microsoft.com/en-us/library/aa995740(v=exchg.65).aspx

https://en.wikipedia.org/wiki/S/MIME

https://www.madboa.com/geek/openssl/#s-mime

https://technet.microsoft.com/en-us/library/aa995740(v=exchg.65).aspx

https://certificate.nikhef.nl/info/smime-manual.html

https://wiki.openssl.org/index.php/Manual:Smime(1)

https://wiki.openssl.org/index.php/Manual:Smime(1)