

Encryption and Decryption Lab

Copyrights 2016-2017 Frank Xu, Bowie State University.
The lab manual is developed for teaching cyber-related courses. Comments and suggestions can be sent to wxu@bowiestate.edu

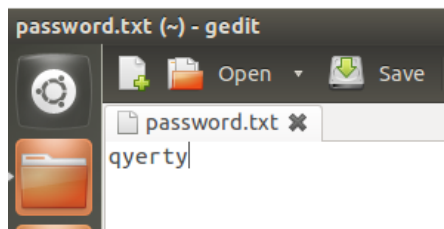
Lab Environment

We have created two accounts in the VM. The usernames and passwords are listed in the following:

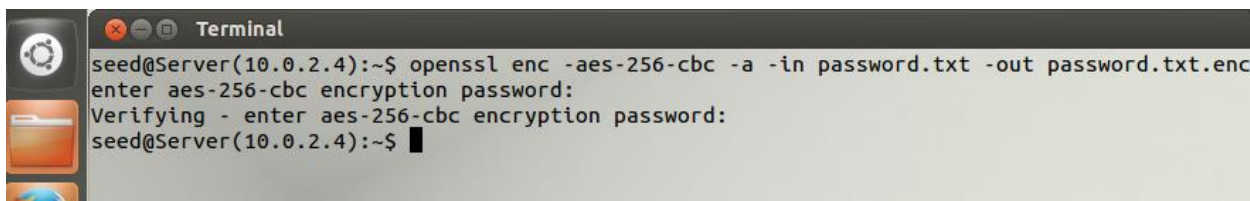
- User ID: root, Password: seedubuntu.
 - Note: Ubuntu does not allow root to login directly from the login window. You have to login as a normal user, and then use the command su to login to the root account.
- User ID: seed, Password: dees

Task 1: AES Encryption and Decryption

1. Create a file named password.txt
 - 1.1 Type: *gedit password.txt*

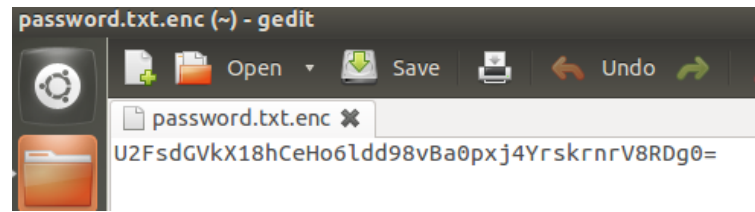


2. Encrypt with the encryption key *1234*
 - 2.1 Type following

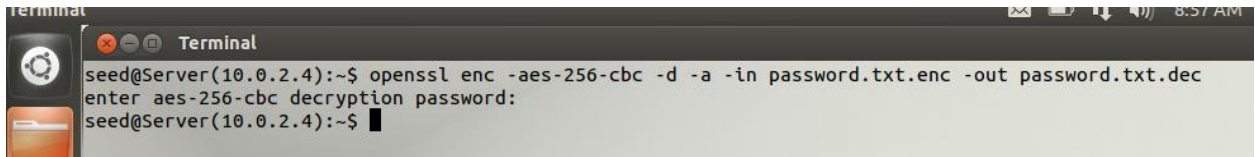


3. Questions:
 - 3.1 What does the option *-a* do? (base64-encode)
 - 3.2 What does the option *-in* do?
 - 3.3 What does the option *-out* do?
 - 3.4 What does the option *-aes-256-cbc* mean?

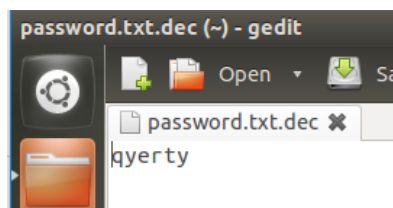
4. Show the encrypted the password



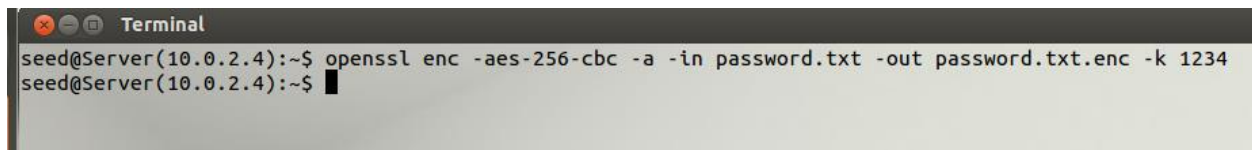
5. Decrypt with the encryption key 1234



6. Show the decrypted the password



7. Encrypt with the option -k



8. Questions:

- 8.1 What does the option -k mean?
- 8.2 Decrypt the password using the option -k.

Task 2: RSA Encryption and Decryption (Public Key)

The following commands are relevant when you work with RSA keys:

- **openssl genrsa**: Generates an RSA private keys.
- **openssl rsa**: Manage RSA private keys (includes generating a public key from it).
- **openssl rsautl**: Encrypt and decrypt files with RSA keys.

1. Generate the Public key.

Using OpenSSL on the command line you'd first need to generate a public and private key, you should password protect this file using the **-passout** argument. This creates a key file called **private.pem** that uses **1024** bits. This file actually have both the private and public keys

```
seed@Server(10.0.2.4):~$ openssl genrsa -out private.pem 1024
Generating RSA private key, 1024 bit long modulus
.....+++++
.....+++++
e is 65537 (0x10001)
seed@Server(10.0.2.4):~$ ls
Desktop      examples.desktop  openssl_1.0.1-4ubuntu5.11.dsc  password.txt.dec  Public
Documents    Music             openssl_1.0.1.orig.tar.gz      password.txt.enc  Templates
Downloads    openssl-1.0.1     password.txt                  Pictures
elggData     openssl_1.0.1-4ubuntu5.11.debian.tar.gz  password.txt~               Videos
seed@Server(10.0.2.4):~$ cat private.pem
-----BEGIN RSA PRIVATE KEY-----
MIICWwIBAAKBgQC9x0xk+NDnjewnvU2kw2NBqCwQLW5cJtLaaxxc150EIP0dJiQw
/687oHD2dszsSe1LWwww7tUApF19xL3+gaXZ4z4HsD5uzE7PvaNNoLxzeo4w1cTC
HCphexg0nxBEJznCueGq3p2Q5mxFUj9qJcJJ7Ss2j9PEyqy1DjFZTD+Q9wIDAQAB
AoGAPJcw8Nvv9r39NALGk0Y+nmvPBas7nPhYQZwJBtd0cWminPvAiXgEK18430y
oYSI0NJUShqarwXLQ6hY8LT7+jLZfzb5/CFTpacGpo3TrS+GADpVKv2/rK+EKsgG
vwZVjhUCHFdWtrNPagjCGpXMnkM9Ft0ry4y31Hzc0sJECAECQD8+4wqA3V5J2dd
8XJGqlc2Nik6n+w6DZ7w5pQf90/E2/R40q12ztLMXgysJNwBMLdhpmCfwSIj/xfB
MR5u5xX3AkeAwArDwKYcw9z75GnlrBC0CWHel08/2XdqBN+gP9L+CVbZ6+imRrx8
p4lu64vn8nVkcj6y/Gwa/GDD2kIxiCdAQJAFEETfL2gJx4HgZb2HKA/EUE01fm9
0PwMMJ9VVqXXRhySGxofjzaKYTnaa+Hbl2DvbAsFC2VMFX0z9kGctVAdCwJAFyCL
yWNLiLiPGatlljQEmDXPxoYnlvGo33J0oFdQ+dDBXJcLe3SHBgU9uN30t0eg2Dg85
w5GOWiWOH55g716+AQJAZziQSNuPDadF9aUcFBfQAUNavfYfjx3PK31+qhMZr/qj
GxHrNSnX7SomTFCeemsSGFpwmmlfDaAg5uRn5EiGHg==
-----END RSA PRIVATE KEY-----
seed@Server(10.0.2.4):~$
```

2. Extract the public one from this file.

```
seed@Server(10.0.2.4):~$ openssl rsa -in private.pem -out public.pem -outform PEM -pubout
writing RSA key
seed@Server(10.0.2.4):~$ ls
Desktop      examples.desktop  openssl_1.0.1-4ubuntu5.11.dsc  password.txt.dec  Public
Documents    Music             openssl_1.0.1.orig.tar.gz      password.txt.enc  public.pem
Downloads    openssl-1.0.1     password.txt                  Pictures
elggData     openssl_1.0.1-4ubuntu5.11.debian.tar.gz  password.txt~               Templates
seed@Server(10.0.2.4):~$ cat public.pem
-----BEGIN PUBLIC KEY-----
MIGfMA0GCsGqSIb3DQEBAQUAA4GNADCBiQKBgQC9x0xk+NDnjewnvU2kw2NBqCwQ
LW5cJtLaaxxc150EIP0dJiQw/687oHD2dszsSe1LWwww7tUApF19xL3+gaXZ4z4H
sD5uzE7PvaNNoLxzeo4w1cTCHCphexg0nxBEJznCueGq3p2Q5mxFUj9qJcJJ7Ss2
j9PEyqy1DjFZTD+Q9wIDAQAB
-----END PUBLIC KEY-----
seed@Server(10.0.2.4):~$
```

- You can freely share this with 3rd parties. You can test it all by just encrypting something yourself using your public key and then decrypting using your private key, first we need a bit of data to encrypt.

```
seed@Server(10.0.2.4):~$ echo 'mypassword' > fileRSA.txt
seed@Server(10.0.2.4):~$ ls
Desktop      examples.desktop  openssl_1.0.1-4ubuntu5.11.debian.tar.gz  password.txt~  private.pem  Videos
Documents    fileRSA.txt       openssl_1.0.1-4ubuntu5.11.dsc             password.txt.dec  Public
Downloads    Music            openssl_1.0.1.orig.tar.gz                 password.txt.enc  public.pem
elggData     openssl-1.0.1     password.txt                                Pictures        Templates
seed@Server(10.0.2.4):~$ cat fileRSA.txt
mypassword
seed@Server(10.0.2.4):~$
```

- You now have some data in *fileRSA.txt*, lets encrypt it using OpenSSL and the public key. Note that for showing the encrypted file, you need to decode it in base 64 (represent binary data in an ASCII string format by translating it into a radix-64 representation).

4.1 **-inkey file:** the input key file, by default it should be an RSA private key.

4.2 **-pubin:** the input file is an RSA public key.

4.3 **-in filename:** This specifies the input filename to read data from or standard input if this option is not specified.

```
seed@Server(10.0.2.4):~$ openssl rsautl -encrypt -inkey public.pem -pubin -in fileRSA.txt -out file.ssl
seed@Server(10.0.2.4):~$ ls
Desktop      examples.desktop  openssl-1.0.1                password.txt  Pictures  Templates
Documents    fileRSA.txt       openssl_1.0.1-4ubuntu5.11.debian.tar.gz  password.txt~  private.pem  Videos
Downloads    file.ssl         openssl_1.0.1-4ubuntu5.11.dsc             password.txt.dec  Public
elggData     Music            openssl_1.0.1.orig.tar.gz                 password.txt.enc  public.pem
seed@Server(10.0.2.4):~$ cat file.ssl | base64
a+DFmpBzJ4rN2IoKD9gQd0/HYb8gXVsTRdmF0KOrNUPA76ey0ZUppUfC3CikNqmK1IJXxcLCmPCR
NF+VQ4j0968RQYtybTjquWGZ+9fGYqbBXvTOxjGRuN5NDjfmT/IXKjg3GP+92Iq5kmpVcD4XwliX
jQWPq7ieTHT11vrB5I4=
seed@Server(10.0.2.4):~$
```

ASCII - Binary Character Table

Letter	ASCII Code	Binary
B	066	01000010
C	067	01000011
D	068	01000100
E	069	01000101

- This creates an encrypted version of *fileRSA.txt* calling it *file.ssl*, if you look at this file it's just binary junk, nothing very useful to anyone. Now you can unencrypt it using the private key.

```
seed@Server(10.0.2.4):~$ openssl rsautl -decrypt -inkey private.pem -in file.ssl -out fileRSA.txt.dec
seed@Server(10.0.2.4):~$ ls
Desktop      fileRSA.txt       openssl_1.0.1-4ubuntu5.11.debian.tar.gz  password.txt.dec  public.pem
Documents    fileRSA.txt.dec  openssl_1.0.1-4ubuntu5.11.dsc             password.txt.enc  Templates
Downloads    file.ssl         openssl_1.0.1.orig.tar.gz                 Pictures          Videos
elggData     Music            password.txt                                private.pem
examples.desktop  openssl-1.0.1     password.txt~                               Public
seed@Server(10.0.2.4):~$ cat fileRSA.txt.dec
mypassword
seed@Server(10.0.2.4):~$ cat fileRSA.txt
mypassword
seed@Server(10.0.2.4):~$
```

Reference:

- https://www.devco.net/archives/2006/02/13/public__private_key_encryption_using_openssl.php
- http://www.sis.pitt.edu/lersais/education/labs/access_control.php
- <https://www.madboa.com/geek/openssllection-13537064>