Asymmetric Encryption

Part I: RSA/AES-cbc encryption/decryption Observations

This lab will compare the performance of asymmetric key algorithms to that of symmetric key encryption algorithms.

1. I began by creating different size files. First I created a 20byte file, then a 16byte file and a 20KB file. These files will be used with the RSA encryption. See **Figure 1.1 – 1.2** below.

Figure 1.1

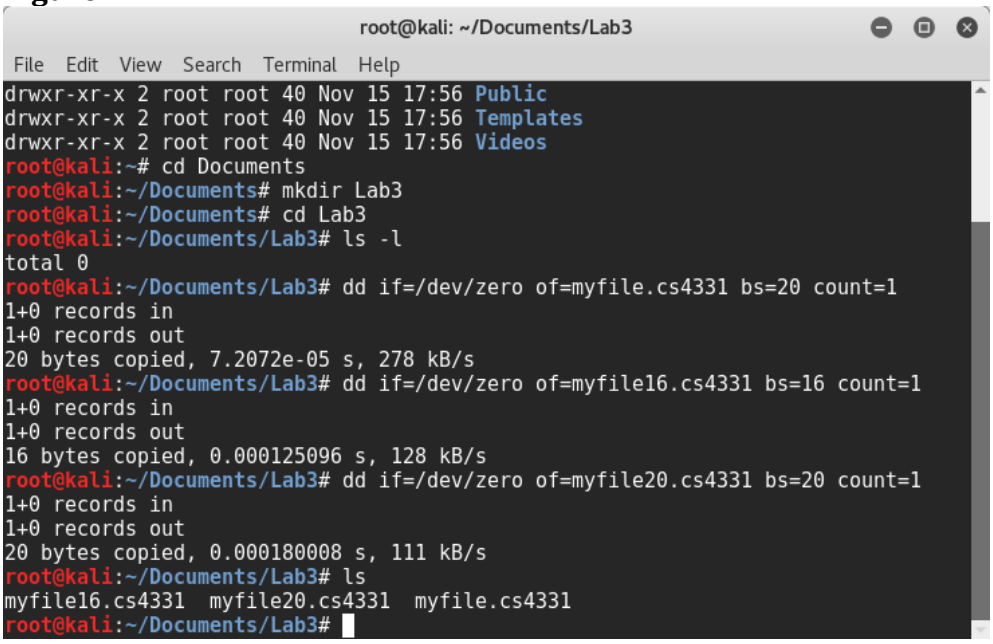
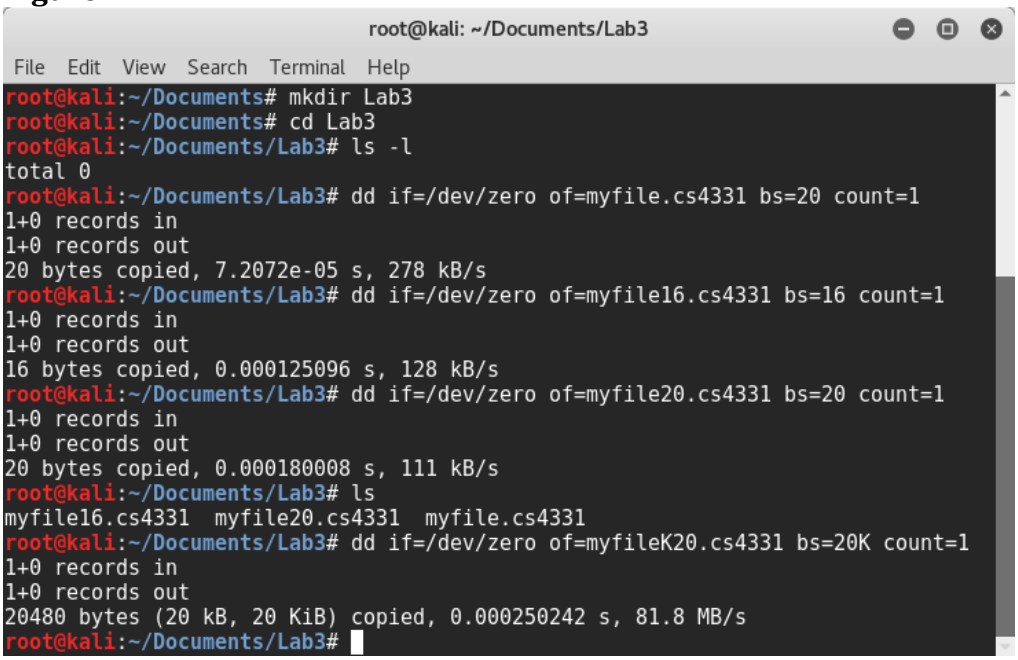
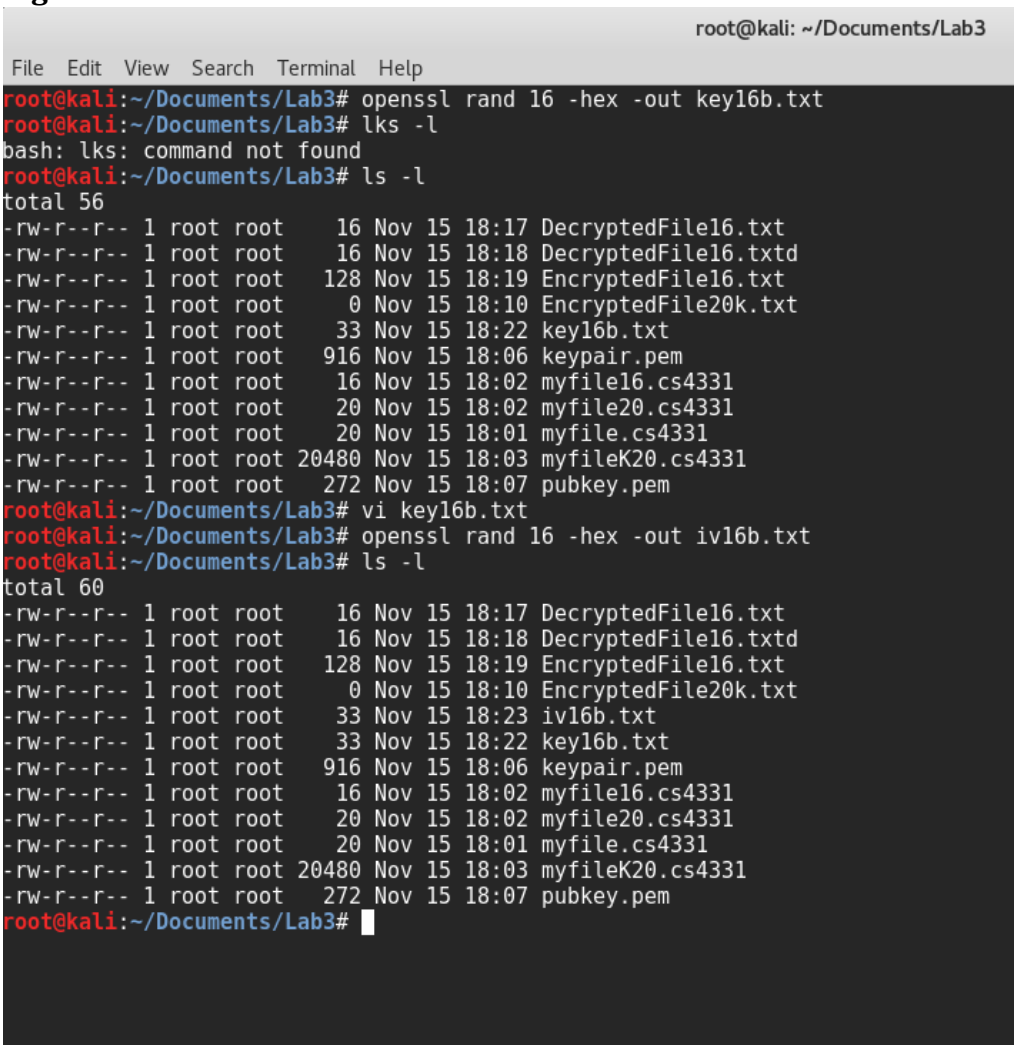


Figure 1.2



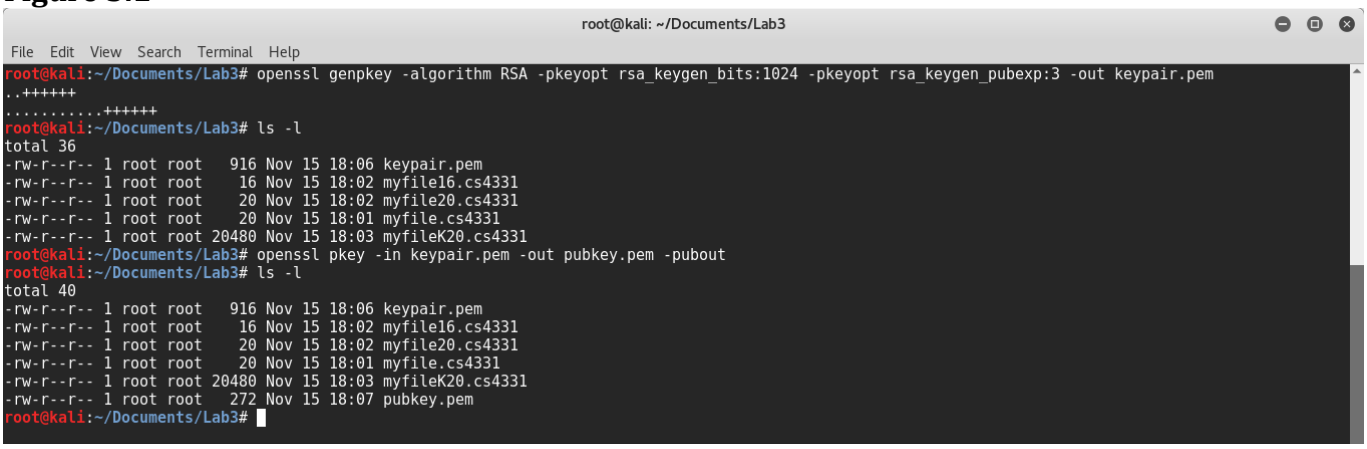
2. The next step was creating the AES key and IV. (I initially skipped this step and continued with the encryption decryption, which I will explain next.) **See Figure 2.1** below.

Figure 2.1



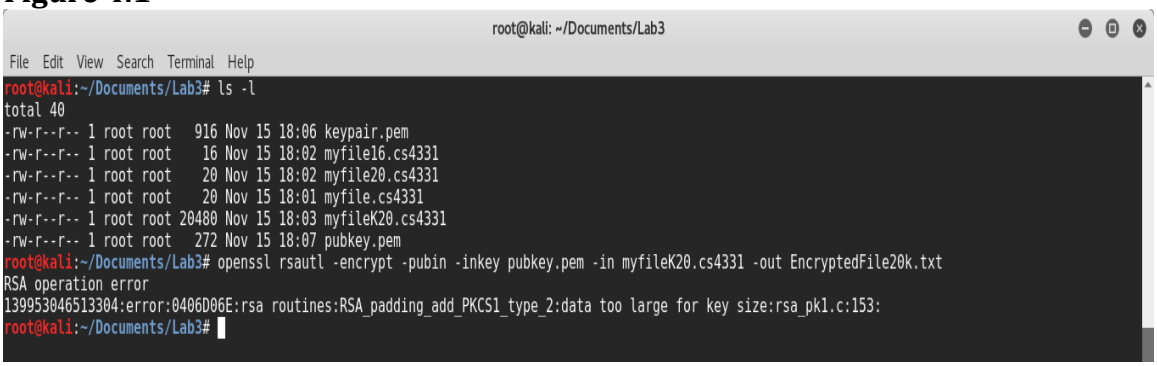
3. Then I created the RSA key pair. See **Figure 3.1** below.

Figure 3.1



4. I then attempted to encrypt the 20KB file using the 1024-bit RSA key. The encryption failed because the key is much smaller than the file size. The key is 1024-bits and the file is 20KB. Therefore there was an error thrown when the encryption began. See **Figure 4.1** below.

Figure 4.1



5. Now I began encrypting the 16-byte file with the public key. After encrypting I decrypted the file. When decrypting I included the time command to observe the real value of time it takes to complete the RSA decryption. The average real value for decryption is 0.01025s. See **Figure 5.1-5.2** below.

Figure 5.1

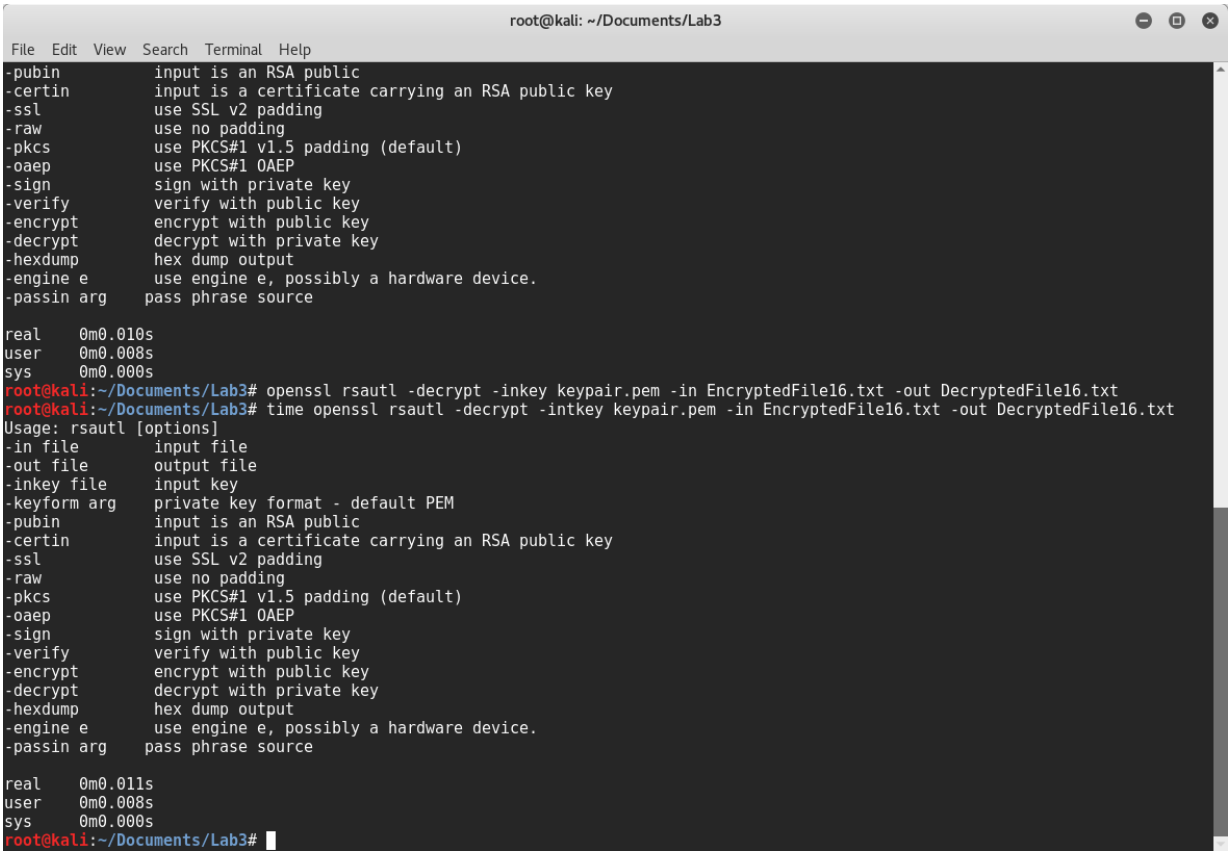


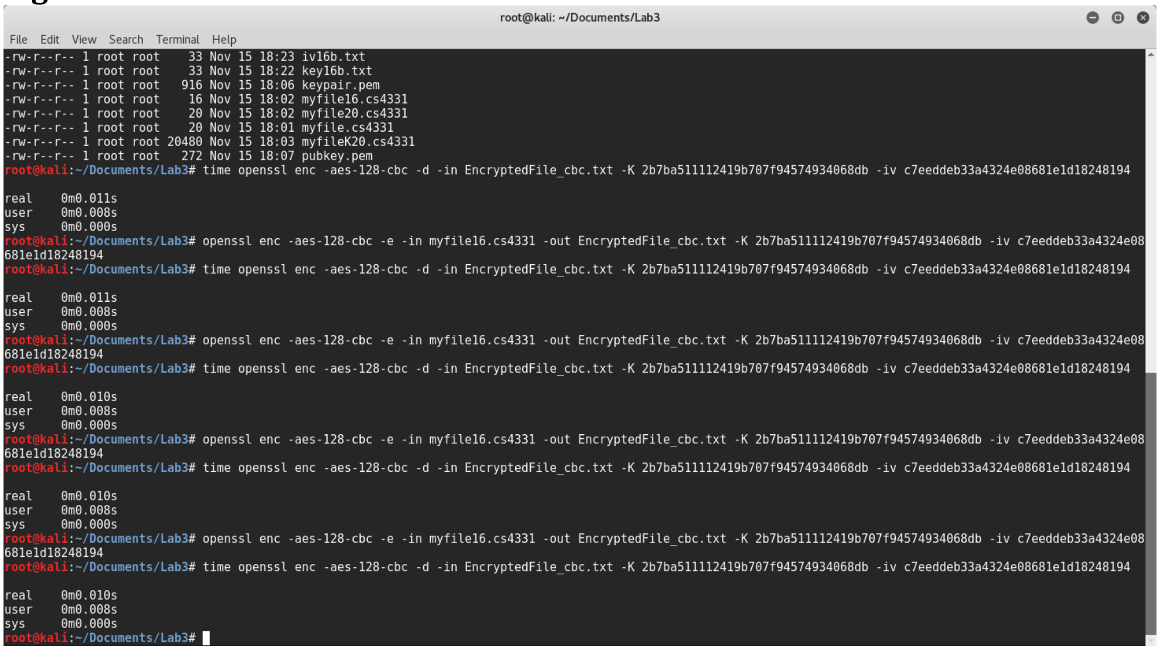
Figure 5.2

```
root@kali: ~/Documents/Lab3
File  Edit  View  Search  Terminal  Help
-pubin          input is an RSA public
-certin         input is a certificate carrying an RSA public key
-ssl            use SSL v2 padding
-raw            use no padding
-pkcs           use PKCS#1 v1.5 padding (default)
-oaep           use PKCS#1 OAEP
-sign           sign with private key
-verify         verify with public key
-encrypt        encrypt with public key
-decrypt        decrypt with private key
-hexdump        hex dump output
-engine e       use engine e, possibly a hardware device.
-passin arg    pass phrase source

real    0m0.010s
user    0m0.008s
sys     0m0.000s
root@kali:~/Documents/Lab3# openssl rsautl -encrypt -pubin -inkey pubkey.pem -in myfile16.cs4331 -out EncryptedFile16.txt
root@kali:~/Documents/Lab3# time openssl rsautl -decrypt -intkey keypair.pem -in EncryptedFile16.txt -out DecryptedFile16.txt
Usage: rsautl [options]
-in file        input file
-out file       output file
-inkey file     input key
-keyform arg    private key format - default PEM
-pubin          input is an RSA public
-certin         input is a certificate carrying an RSA public key
-ssl            use SSL v2 padding
-raw            use no padding
-pkcs           use PKCS#1 v1.5 padding (default)
-oaep           use PKCS#1 OAEP
-sign           sign with private key
-verify         verify with public key
-encrypt        encrypt with public key
-decrypt        decrypt with private key
-hexdump        hex dump output
-engine e       use engine e, possibly a hardware device.
-passin arg    pass phrase source

real    0m0.010s
user    0m0.008s
sys     0m0.000s
root@kali:~/Documents/Lab3#
```

6. I then encrypted the 16byte file using AES in cbc mode. When decrypting the file I used the time command to observe the real value of time. The average real value of time for decryption is 0.0105s. See **Figure 6.1-6.2** below.

```
root@kali: ~/Documents/Lab3
File  Edit  View  Search  Terminal  Help
root@kali:~/Documents/Lab3# openssl enc -aes-128-cbc -e -in myfile16.cs4331 -out EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08
681e1d18248194
root@kali:~/Documents/Lab3# ls -l
total 60
-rw-r--r-- 1 root root    16 Nov 15 18:17 DecryptedFile16.txt
-rw-r--r-- 1 root root   128 Nov 15 18:19 EncryptedFile16.txt
-rw-r--r-- 1 root root     0 Nov 15 18:10 EncryptedFile20k.txt
-rw-r--r-- 1 root root    32 Nov 15 18:29 EncryptedFile_cbc.txt
-rw-r--r-- 1 root root    33 Nov 15 18:23 iv16b.txt
-rw-r--r-- 1 root root    33 Nov 15 18:22 key16b.txt
-rw-r--r-- 1 root root   916 Nov 15 18:06 keypair.pem
-rw-r--r-- 1 root root    16 Nov 15 18:02 myfile16.cs4331
-rw-r--r-- 1 root root    20 Nov 15 18:02 myfile20.cs4331
-rw-r--r-- 1 root root    20 Nov 15 18:01 myfile.cs4331
-rw-r--r-- 1 root root 20480 Nov 15 18:03 myfileK20.cs4331
-rw-r--r-- 1 root root   272 Nov 15 18:07 pubkey.pem
root@kali:~/Documents/Lab3#
```

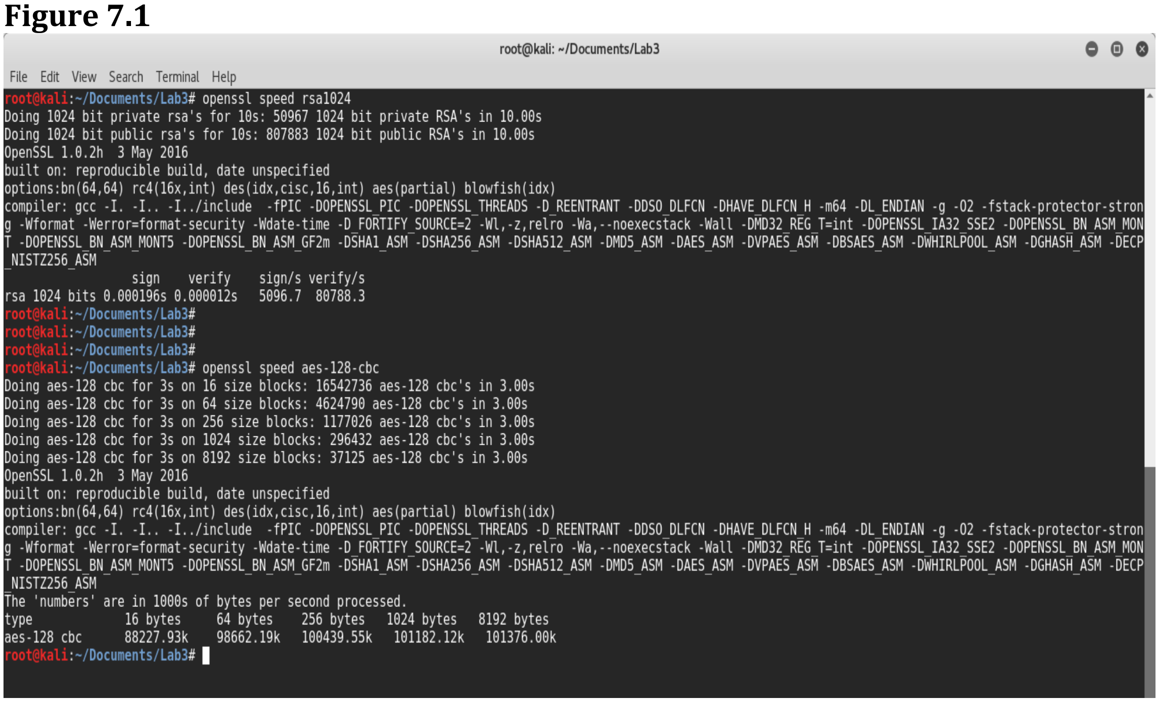
Figure 6.2



The difference between the RSA decryption and the AES-cbc decryption in the real value of time is 2.5×10^{-4}. Therefore the RSA decryption is 2.5×10^{-4}s faster than the AES-cbc decryption.

7. To more clearly see this difference, I used the standard benchmarks to compare. See **Figure 7.1** below.

Figure 7.1



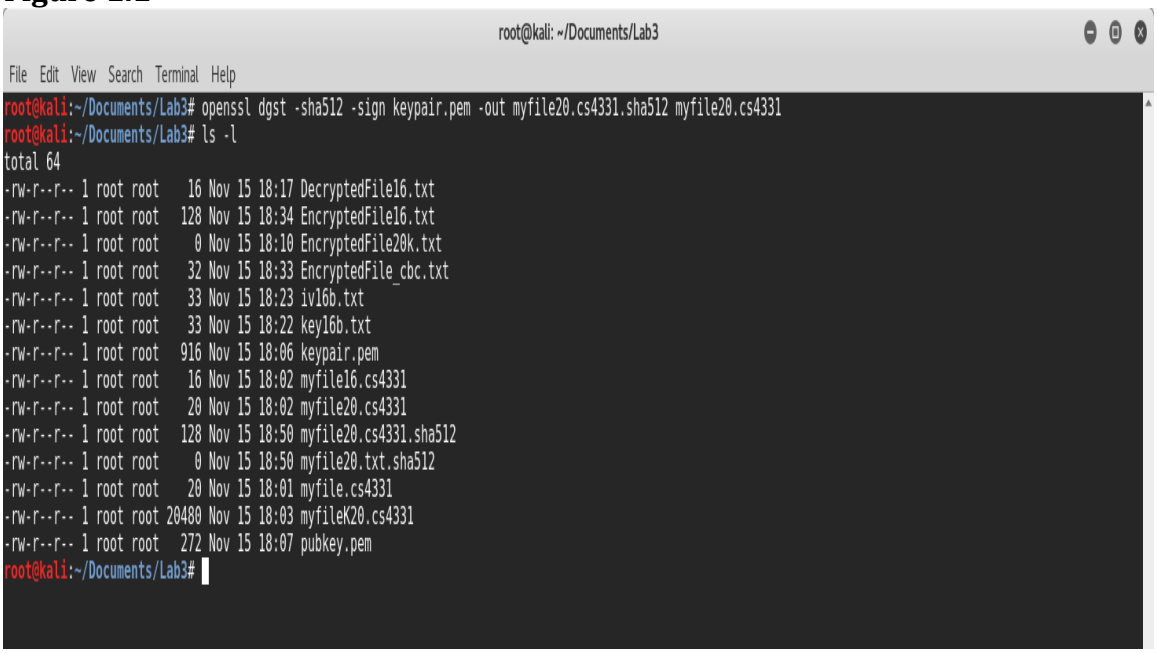
In conclusion, step 7 shows the benchmarks for the encryption/decryption speed for RSA and AES-cbc. The data shows that AES-cbc with a 16block should only take 3s but my observations show it takes 0.0105s on average. The benchmark for RSA encryption/decryption data shows it takes 10s but my observations show that RSA takes 0.01025s on average. I believe the speed of my machine allowed for the encryption of both cyphers to beat the benchmark for each respectively.

Part II: Digital Signatures

This part of the lab will explain the creating and verifying of digital signatures using openssl.

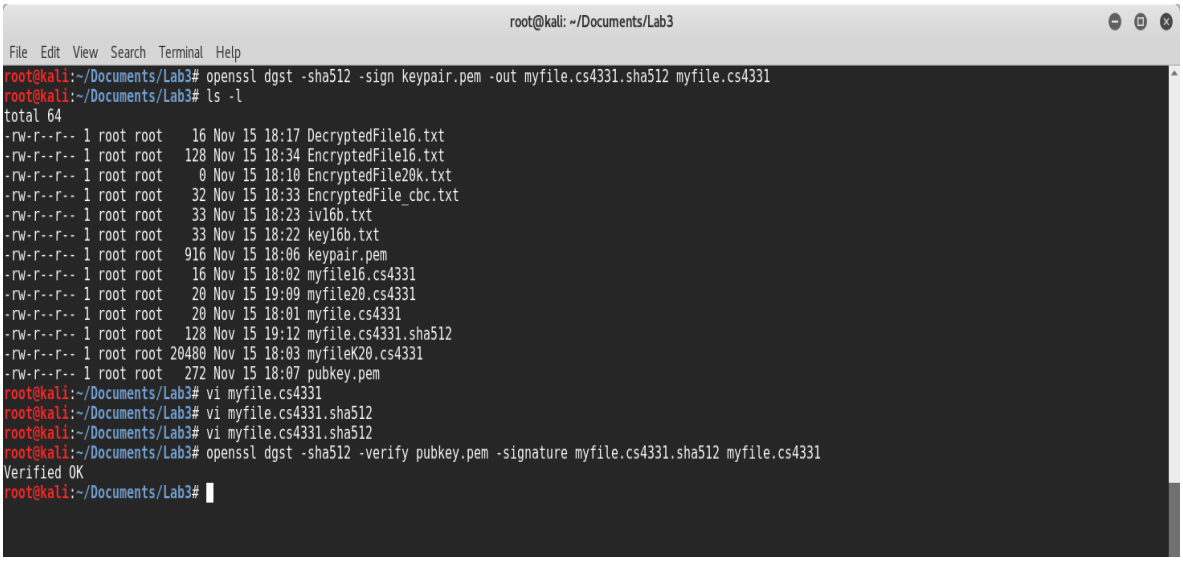
1. First I signed the myfile20.cs4331 using the private key created in the previous steps. I also used SHA512 for the hash. **See Figure 1.1** below.

Figure 1.1



2. I then verified the signature of the output file using the public key that corresponds with the private key used above. See **Figure 2.1** below.

Figure 2.1



3. I then edited the file. After editing the file I attempted to verify the signature again. See **Figure 3.1-3.2** below.

Figure 3.1

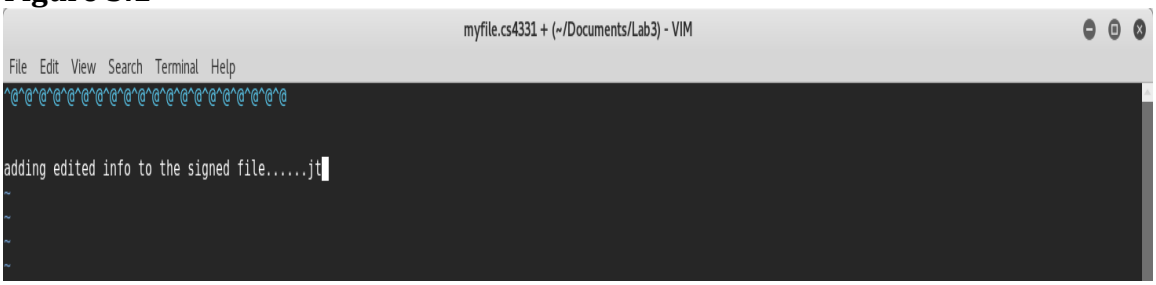
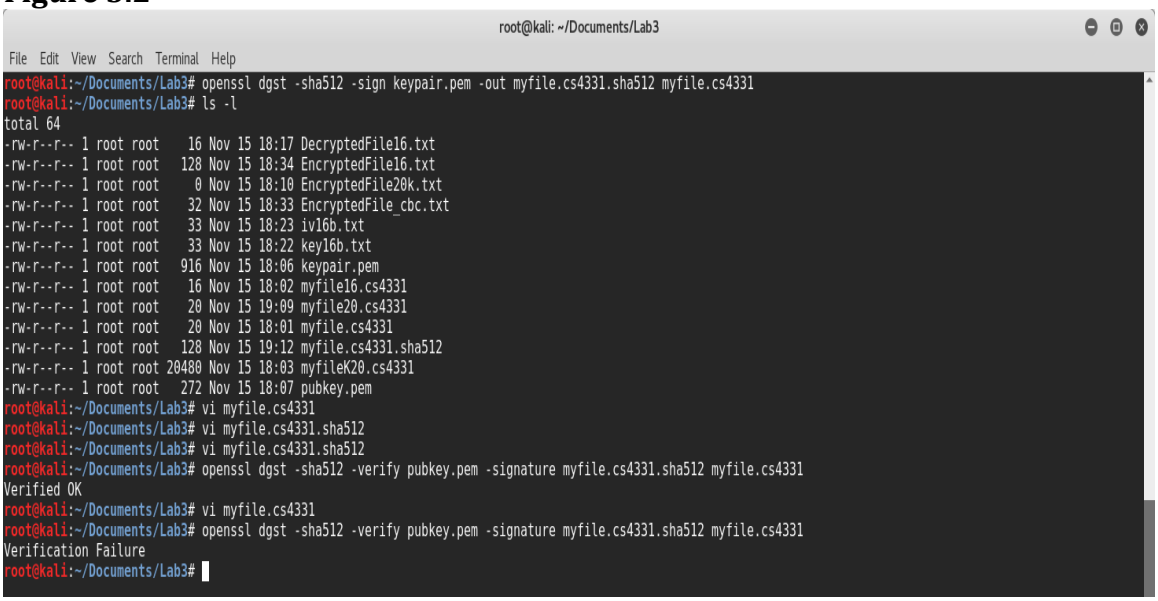


Figure 3.2



In conclusion my observations show that editing a signed document without authenticating it after causes the signature to become invalid. This can help when sending information or downloading information, the user can verify the file downloaded or sent is the correct file and has not been tampered with