

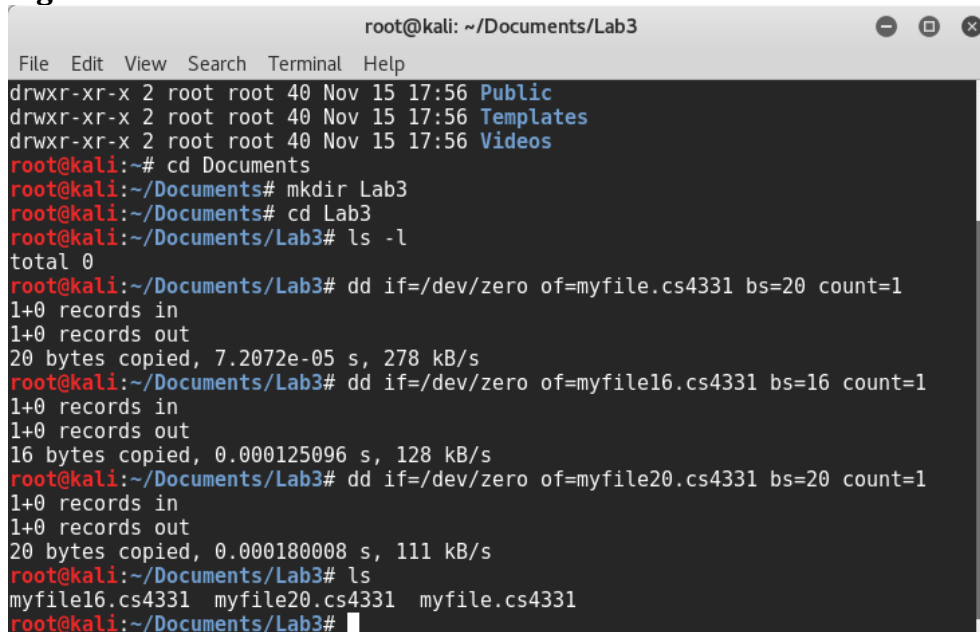
## Asymmetric Encryption

### 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.

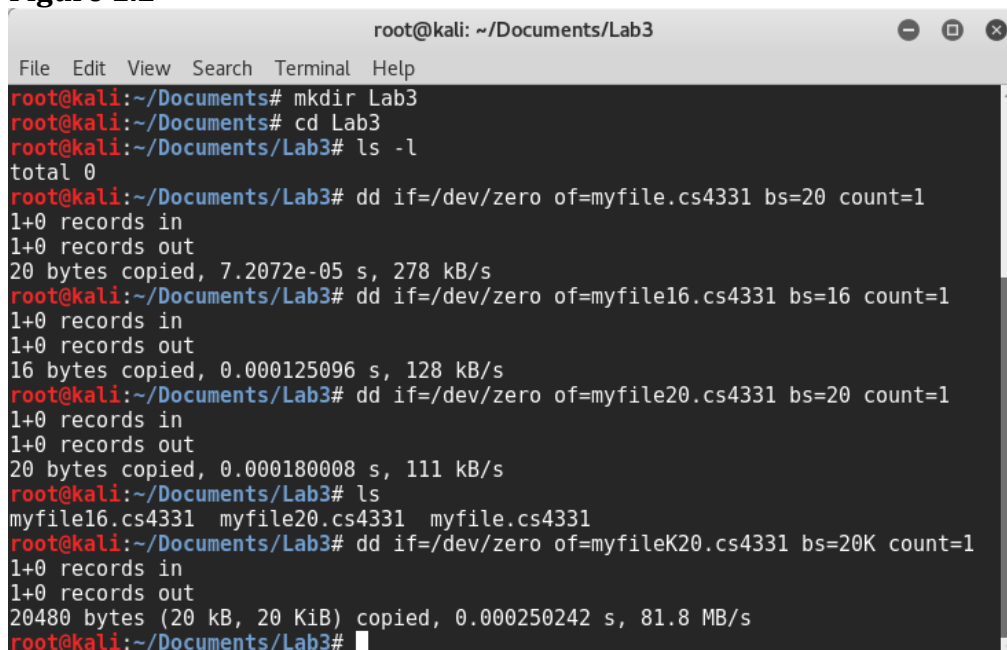
1. I began by creating different size files. First I created a 20byte file, then a 16byte file and a 20KB file. These files will be used with the RSA encryption. See **Figure 1.1 – 1.2** below.

**Figure 1.1**



```
root@kali: ~/Documents/Lab3
File Edit View Search Terminal Help
drwxr-xr-x 2 root root 40 Nov 15 17:56 Public
drwxr-xr-x 2 root root 40 Nov 15 17:56 Templates
drwxr-xr-x 2 root root 40 Nov 15 17:56 Videos
root@kali:~# cd Documents
root@kali:~/Documents# mkdir Lab3
root@kali:~/Documents# cd Lab3
root@kali:~/Documents/Lab3# ls -l
total 0
root@kali:~/Documents/Lab3# dd if=/dev/zero of=myfile.cs4331 bs=20 count=1
1+0 records in
1+0 records out
20 bytes copied, 7.2072e-05 s, 278 kB/s
root@kali:~/Documents/Lab3# dd if=/dev/zero of=myfile16.cs4331 bs=16 count=1
1+0 records in
1+0 records out
16 bytes copied, 0.000125096 s, 128 kB/s
root@kali:~/Documents/Lab3# dd if=/dev/zero of=myfile20.cs4331 bs=20 count=1
1+0 records in
1+0 records out
20 bytes copied, 0.000180008 s, 111 kB/s
root@kali:~/Documents/Lab3# ls
myfile16.cs4331 myfile20.cs4331 myfile.cs4331
root@kali:~/Documents/Lab3#
```

**Figure 1.2**



```
root@kali: ~/Documents/Lab3
File Edit View Search Terminal Help
root@kali:~/Documents# mkdir Lab3
root@kali:~/Documents# cd Lab3
root@kali:~/Documents/Lab3# ls -l
total 0
root@kali:~/Documents/Lab3# dd if=/dev/zero of=myfile.cs4331 bs=20 count=1
1+0 records in
1+0 records out
20 bytes copied, 7.2072e-05 s, 278 kB/s
root@kali:~/Documents/Lab3# dd if=/dev/zero of=myfile16.cs4331 bs=16 count=1
1+0 records in
1+0 records out
16 bytes copied, 0.000125096 s, 128 kB/s
root@kali:~/Documents/Lab3# dd if=/dev/zero of=myfile20.cs4331 bs=20 count=1
1+0 records in
1+0 records out
20 bytes copied, 0.000180008 s, 111 kB/s
root@kali:~/Documents/Lab3# ls
myfile16.cs4331 myfile20.cs4331 myfile.cs4331
root@kali:~/Documents/Lab3# dd if=/dev/zero of=myfileK20.cs4331 bs=20K count=1
1+0 records in
1+0 records out
20480 bytes (20 kB, 20 KiB) copied, 0.000250242 s, 81.8 MB/s
root@kali:~/Documents/Lab3#
```

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**

```
root@kali: ~/Documents/Lab3
File Edit View Search Terminal Help
root@kali:~/Documents/Lab3# openssl rand 16 -hex -out key16b.txt
root@kali:~/Documents/Lab3# lks -l
bash: lks: command not found
root@kali:~/Documents/Lab3# ls -l
total 56
-rw-r--r-- 1 root root 16 Nov 15 18:17 DecryptedFile16.txt
-rw-r--r-- 1 root root 16 Nov 15 18:18 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 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# vi key16b.txt
root@kali:~/Documents/Lab3# openssl rand 16 -hex -out iv16b.txt
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 16 Nov 15 18:18 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 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#
```

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

**Figure 3.1**

```
root@kali: ~/Documents/Lab3
File Edit View Search Terminal Help
root@kali:~/Documents/Lab3# openssl genkey -algorithm RSA -pkeyopt rsa_keygen_bits:1024 -pkeyopt rsa_keygen_pubexp:3 -out keypair.pem
.....
root@kali:~/Documents/Lab3# ls -l
total 36
-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
root@kali:~/Documents/Lab3# openssl pkey -in keypair.pem -out pubkey.pem -pubout
root@kali:~/Documents/Lab3# ls -l
total 40
-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#
```

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**



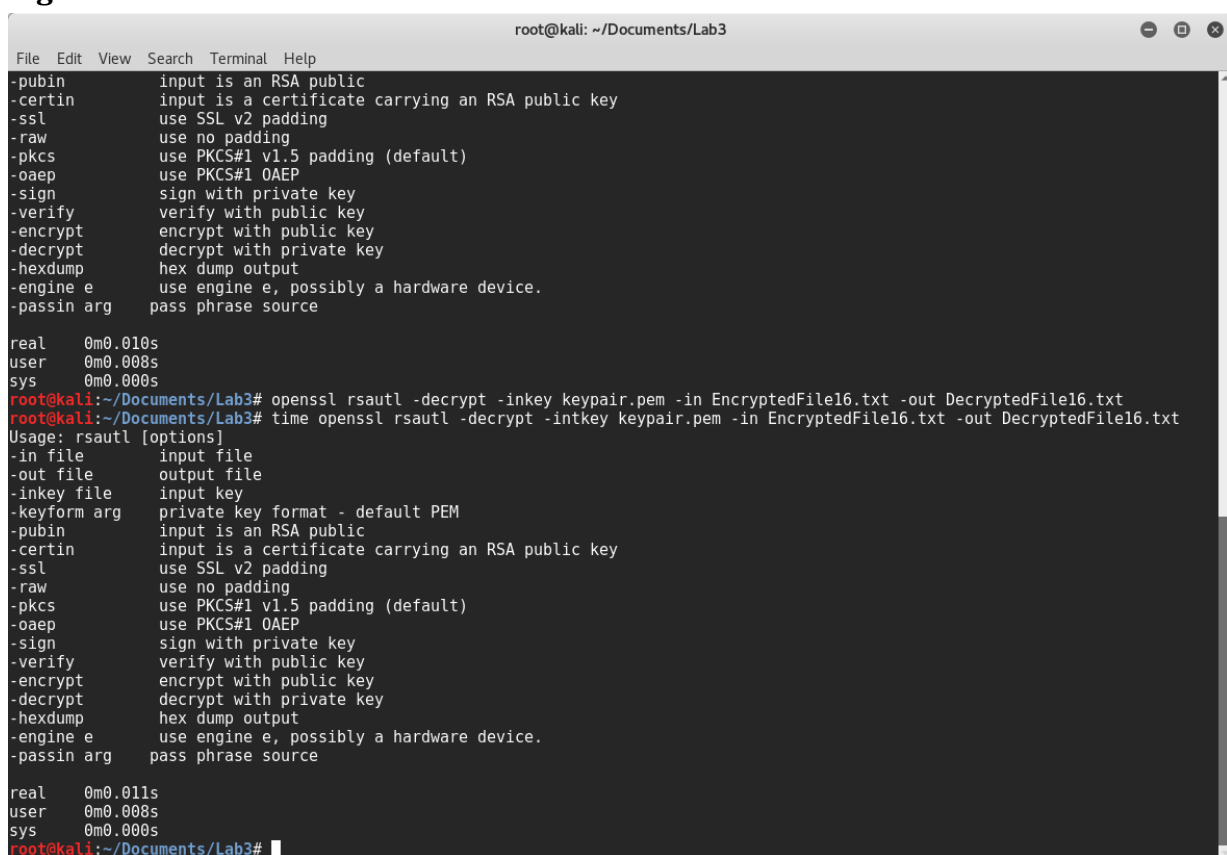
```

root@kali: ~/Documents/Lab3
File Edit View Search Terminal Help
root@kali:~/Documents/Lab3# ls -l
total 40
-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# openssl rsautl -encrypt -pubin -inkey pubkey.pem -in myfileK20.cs4331 -out EncryptedFile20k.txt
RSA operation error
139953046513304:error:0406D06E:rsa routines:RSA_padding_add_PKCS1_type_2:data too large for key size:rsa_pk1.c:153:
root@kali:~/Documents/Lab3#

```

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**



```

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 -decrypt -inkey keypair.pem -in EncryptedFile16.txt -out DecryptedFile16.txt
root@kali:~/Documents/Lab3# time openssl rsautl -decrypt -inkey 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    0m0.011s
user    0m0.008s
sys     0m0.000s
root@kali:~/Documents/Lab3#

```

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 -inkey 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#

```

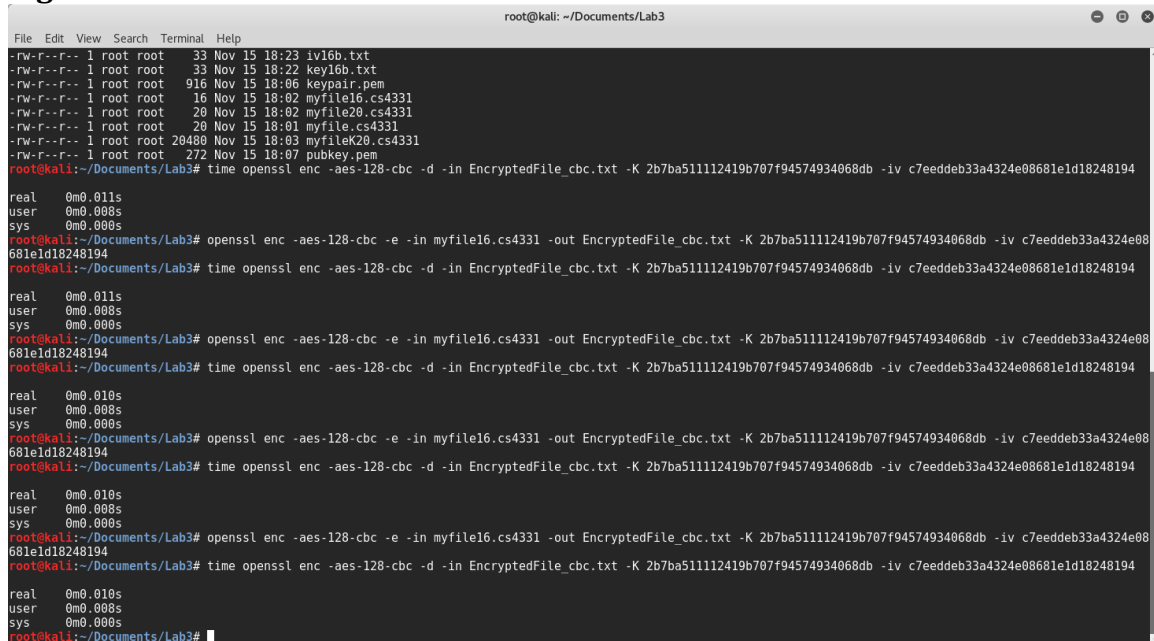
- I then encrypted the 16byte file using AES in cbc mode. When decrypting the file I used the time command to observe the real value of time. The average real value of time for decryption is 0.0105s. See **Figure 6.1-6.2** below.

```

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 2b7ba511112419b707f94574934060db -iv c7eeddeb33a4324e00
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 18:18 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

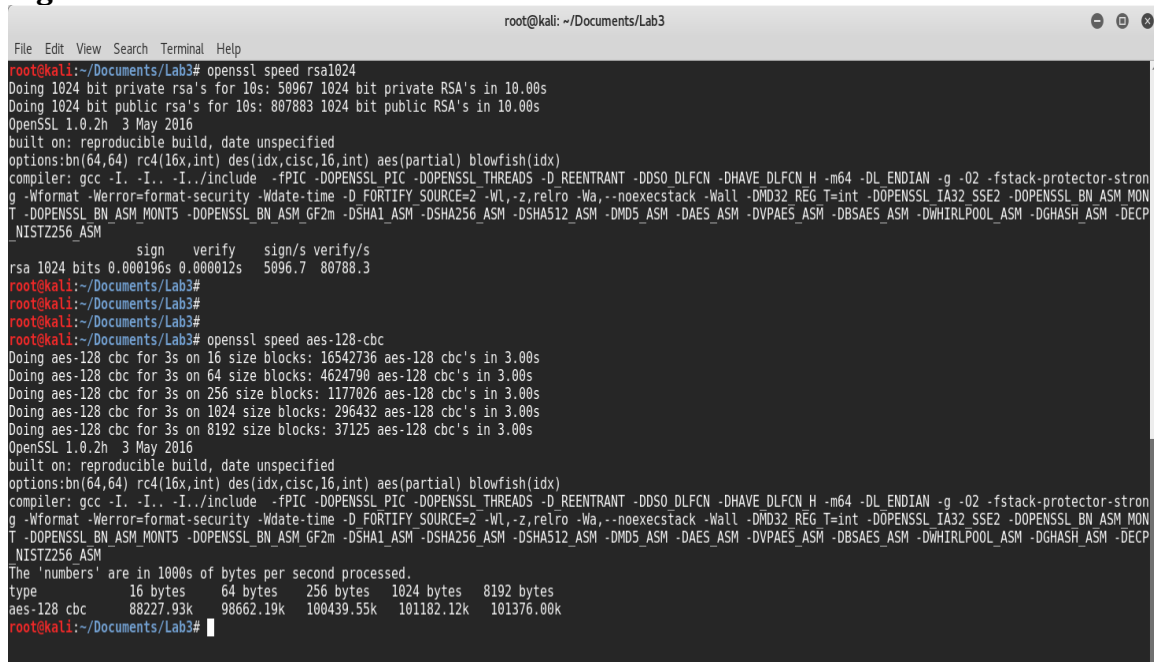


```
root@kali: ~/Documents/Lab3
File Edit View Search Terminal Help
-rw-r--r-- 1 root root 33 Nov 15 18:23 iv16b.txt
-rw-r--r-- 1 root root 33 Nov 15 18:22 key16b.txt
-rw-r--r-- 1 root root 916 Nov 15 18:06 keypair.pem
-rw-r--r-- 1 root root 16 Nov 15 18:02 myfile16.cs4331
-rw-r--r-- 1 root root 20 Nov 15 18:02 myfile20.cs4331
-rw-r--r-- 1 root root 20 Nov 15 18:01 myfile.cs4331
-rw-r--r-- 1 root root 20460 Nov 15 18:03 myfilek20.cs4331
-rw-r--r-- 1 root root 272 Nov 15 18:07 pubkey.pem
root@kali:~/Documents/Lab3# time openssl enc -aes-128-cbc -d -in EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08681d18248194
real    0m0.011s
user    0m0.008s
sys     0m0.000s
root@kali:~/Documents/Lab3# openssl enc -aes-128-cbc -e -in myfile16.cs4331 -out EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08681d18248194
root@kali:~/Documents/Lab3# time openssl enc -aes-128-cbc -d -in EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08681d18248194
real    0m0.011s
user    0m0.008s
sys     0m0.000s
root@kali:~/Documents/Lab3# openssl enc -aes-128-cbc -e -in myfile16.cs4331 -out EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08681d18248194
root@kali:~/Documents/Lab3# time openssl enc -aes-128-cbc -d -in EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08681d18248194
real    0m0.010s
user    0m0.008s
sys     0m0.000s
root@kali:~/Documents/Lab3# openssl enc -aes-128-cbc -e -in myfile16.cs4331 -out EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08681d18248194
root@kali:~/Documents/Lab3# time openssl enc -aes-128-cbc -d -in EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08681d18248194
real    0m0.010s
user    0m0.008s
sys     0m0.000s
root@kali:~/Documents/Lab3# openssl enc -aes-128-cbc -e -in myfile16.cs4331 -out EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08681d18248194
root@kali:~/Documents/Lab3# time openssl enc -aes-128-cbc -d -in EncryptedFile_cbc.txt -K 2b7ba511112419b707f94574934068db -iv c7eeddeb33a4324e08681d18248194
real    0m0.010s
user    0m0.008s
sys     0m0.000s
root@kali:~/Documents/Lab3#
```

The difference between the RSA decryption and the AES-cbc decryption in the real value of time is  $2.5 \times 10^{-4}$ . Therefore the RSA decryption is  $2.5 \times 10^{-4}$ 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



```
root@kali: ~/Documents/Lab3
File Edit View Search Terminal Help
root@kali:~/Documents/Lab3# openssl speed rsa1024
Doing 1024 bit private rsa's for 10s: 50967 1024 bit private RSA's in 10.00s
Doing 1024 bit public rsa's for 10s: 807883 1024 bit public RSA's in 10.00s
OpenSSL 1.0.2h  3 May 2016
built on: reproducible build, date unspecified
options:bn(64,64) rc4(16x,int) des(idx,cisc,16,int) aes(partial) blowfish(idx)
compiler: gcc -I. -I. -I../include -fPIC -DOPENSSL_PIC -DOPENSSL_THREADS -D_REENTRANT -DDSO_DLFCN -DHAVE_DLFCN_H -m64 -DL_ENDIAN -g -O2 -fstack-protector-strong -Wformat -Werror=format-security -Wdate-time -D_FORTIFY_SOURCE=2 -Wl,-z,relro -Wa,--noexecstack -Wall -DMD32_REG_T=int -DOPENSSL_IA32_SSE2 -DOPENSSL_BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DSHA1_ASM -DSHA256_ASM -DSHA512_ASM -DMD5_ASM -DAES_ASM -DVPAES_ASM -DBSAES_ASM -DWHIRLPOOL_ASM -DGHASH_ASM -DECP_NISTZ256_ASM
sign    verify    sign/s verify/s
rsa 1024 bits 0.000196s 0.000012s 5096.7 80788.3
root@kali:~/Documents/Lab3#
root@kali:~/Documents/Lab3#
root@kali:~/Documents/Lab3#
root@kali:~/Documents/Lab3# openssl speed aes-128-cbc
Doing aes-128 cbc for 3s on 16 size blocks: 16542736 aes-128 cbc's in 3.00s
Doing aes-128 cbc for 3s on 64 size blocks: 4624790 aes-128 cbc's in 3.00s
Doing aes-128 cbc for 3s on 256 size blocks: 1177826 aes-128 cbc's in 3.00s
Doing aes-128 cbc for 3s on 1024 size blocks: 296432 aes-128 cbc's in 3.00s
Doing aes-128 cbc for 3s on 8192 size blocks: 37125 aes-128 cbc's in 3.00s
OpenSSL 1.0.2h  3 May 2016
built on: reproducible build, date unspecified
options:bn(64,64) rc4(16x,int) des(idx,cisc,16,int) aes(partial) blowfish(idx)
compiler: gcc -I. -I. -I../include -fPIC -DOPENSSL_PIC -DOPENSSL_THREADS -D_REENTRANT -DDSO_DLFCN -DHAVE_DLFCN_H -m64 -DL_ENDIAN -g -O2 -fstack-protector-strong -Wformat -Werror=format-security -Wdate-time -D_FORTIFY_SOURCE=2 -Wl,-z,relro -Wa,--noexecstack -Wall -DMD32_REG_T=int -DOPENSSL_IA32_SSE2 -DOPENSSL_BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DSHA1_ASM -DSHA256_ASM -DSHA512_ASM -DMD5_ASM -DAES_ASM -DVPAES_ASM -DBSAES_ASM -DWHIRLPOOL_ASM -DGHASH_ASM -DECP_NISTZ256_ASM
The 'numbers' are in 1000s of bytes per second processed.
type      16 bytes    64 bytes    256 bytes    1024 bytes    8192 bytes
aes-128 cbc 88227.93k  98662.19k  100439.55k  101182.12k  101376.00k
root@kali:~/Documents/Lab3#
```

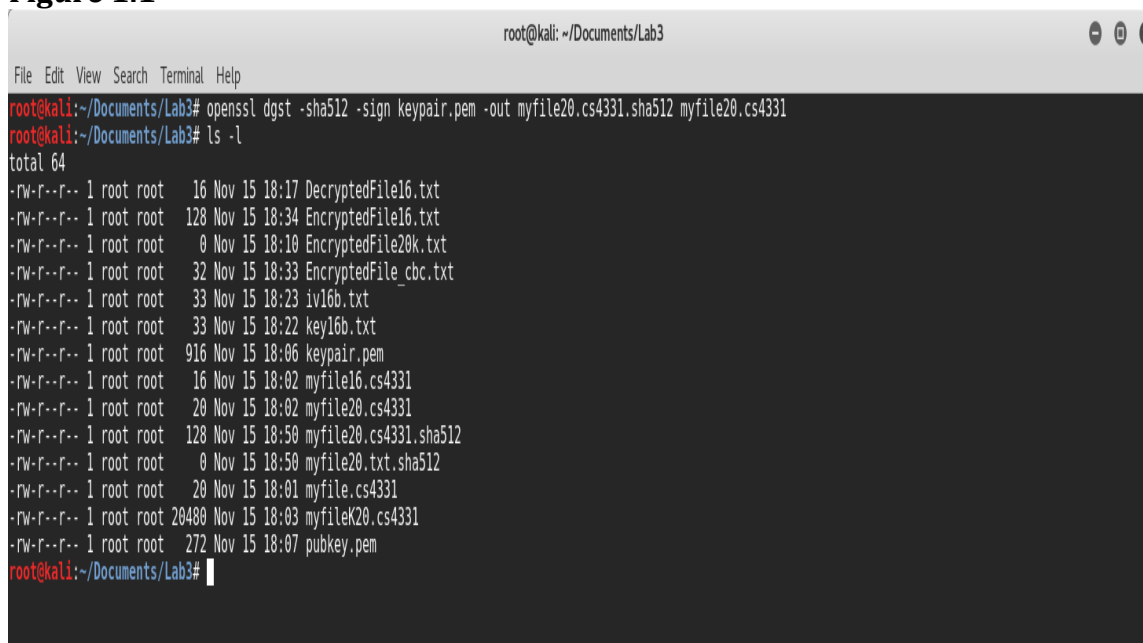
In conclusion, step 7 shows the benchmarks for the encryption/decryption speed for RSA and AES-cbc. The data shows that AES-cbc with a 16block should only take 3s but my observations show it takes 0.0105s on average. The benchmark for RSA encryption/decryption data shows it takes 10s but my observations 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**



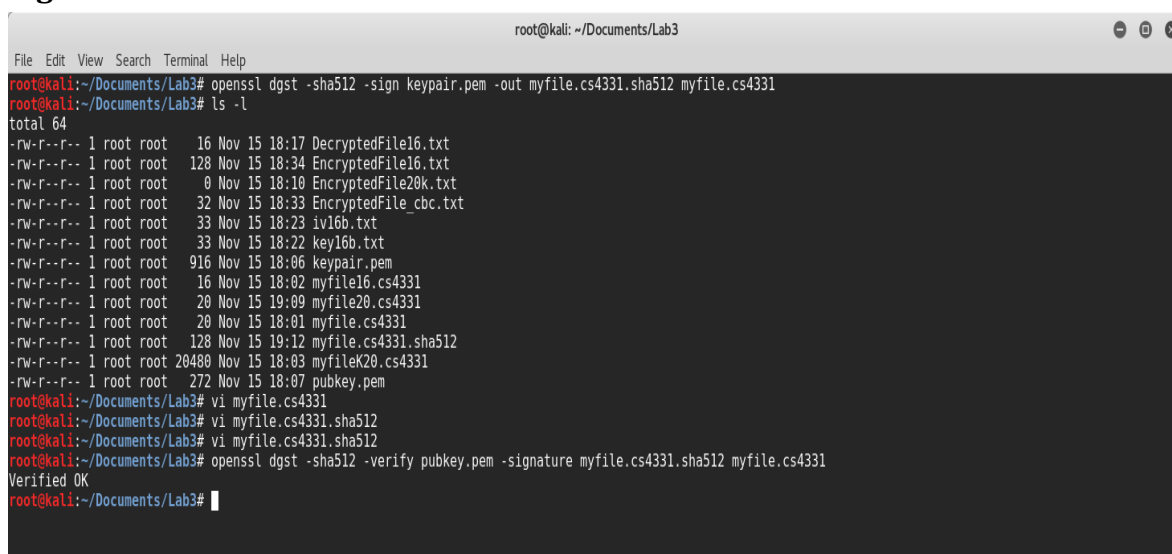
```

root@kali: ~/Documents/Lab3
File Edit View Search Terminal Help
root@kali:~/Documents/Lab3# openssl dgst -sha512 -sign keypair.pem -out myfile20.cs4331.sha512 myfile20.cs4331
root@kali:~/Documents/Lab3# ls -l
total 64
-rw-r--r-- 1 root root 16 Nov 15 18:17 DecryptedFile16.txt
-rw-r--r-- 1 root root 128 Nov 15 18:34 EncryptedFile16.txt
-rw-r--r-- 1 root root 0 Nov 15 18:10 EncryptedFile20k.txt
-rw-r--r-- 1 root root 32 Nov 15 18:33 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 128 Nov 15 18:50 myfile20.cs4331.sha512
-rw-r--r-- 1 root root 0 Nov 15 18:50 myfile20.txt.sha512
-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#

```

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**



```

root@kali: ~/Documents/Lab3
File Edit View Search Terminal Help
root@kali:~/Documents/Lab3# openssl dgst -sha512 -sign keypair.pem -out myfile.cs4331.sha512 myfile.cs4331
root@kali:~/Documents/Lab3# ls -l
total 64
-rw-r--r-- 1 root root 16 Nov 15 18:17 DecryptedFile16.txt
-rw-r--r-- 1 root root 128 Nov 15 18:34 EncryptedFile16.txt
-rw-r--r-- 1 root root 0 Nov 15 18:10 EncryptedFile20k.txt
-rw-r--r-- 1 root root 32 Nov 15 18:33 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 19:09 myfile20.cs4331
-rw-r--r-- 1 root root 20 Nov 15 18:01 myfile.cs4331
-rw-r--r-- 1 root root 128 Nov 15 19:12 myfile.cs4331.sha512
-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# vi myfile.cs4331
root@kali:~/Documents/Lab3# vi myfile.cs4331.sha512
root@kali:~/Documents/Lab3# vi myfile.cs4331.sha512
root@kali:~/Documents/Lab3# openssl dgst -sha512 -verify pubkey.pem -signature myfile.cs4331.sha512 myfile.cs4331
Verified OK
root@kali:~/Documents/Lab3#

```

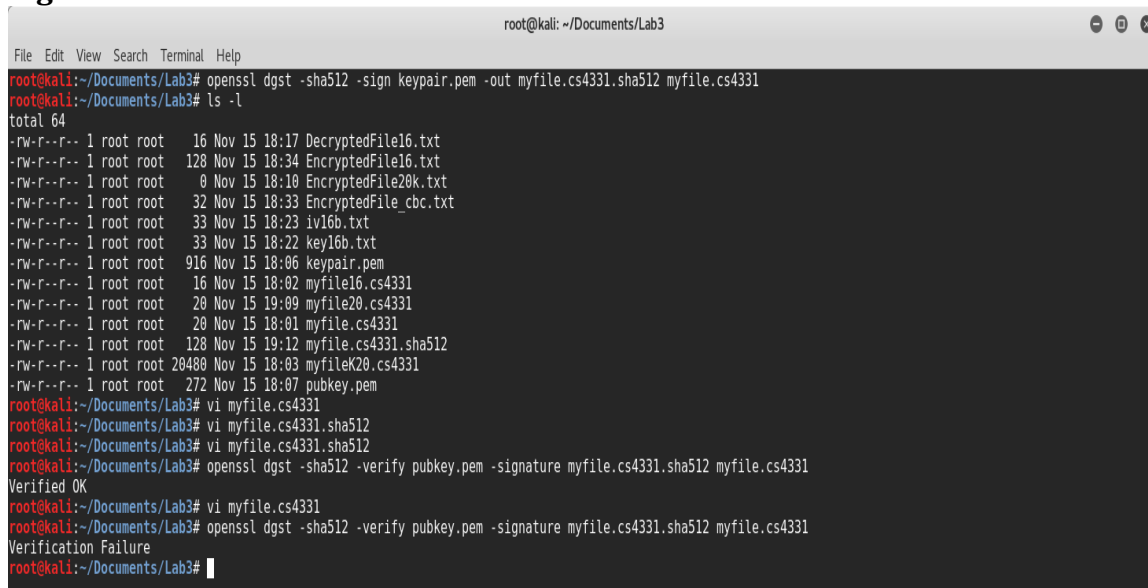
3. I then edited the file. After editing the file I attempted to verify the signature again. See **Figure 3.1-3.2** below.

**Figure 3.1**



```
myfile.cs4331 + (~/.Documents/Lab3) - VIM
File Edit View Search Terminal Help
~
~
~
~
~
adding edited info to the signed file.....jt
~
~
~
```

**Figure 3.2**



```
root@kali: ~/.Documents/Lab3
root@kali:~/.Documents/Lab3# openssl dgst -sha512 -sign keypair.pem -out myfile.cs4331.sha512 myfile.cs4331
root@kali:~/.Documents/Lab3# ls -l
total 64
-rw-r--r-- 1 root root 16 Nov 15 18:17 DecryptedFile16.txt
-rw-r--r-- 1 root root 128 Nov 15 18:34 EncryptedFile16.txt
-rw-r--r-- 1 root root 8 Nov 15 18:18 EncryptedFile20k.txt
-rw-r--r-- 1 root root 32 Nov 15 18:33 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 19:09 myfile20.cs4331
-rw-r--r-- 1 root root 20 Nov 15 18:01 myfile.cs4331
-rw-r--r-- 1 root root 128 Nov 15 19:12 myfile.cs4331.sha512
-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# vi myfile.cs4331
root@kali:~/.Documents/Lab3# vi myfile.cs4331.sha512
root@kali:~/.Documents/Lab3# vi myfile.cs4331.sha512
root@kali:~/.Documents/Lab3# openssl dgst -sha512 -verify pubkey.pem -signature myfile.cs4331.sha512 myfile.cs4331
Verified OK
root@kali:~/.Documents/Lab3# vi myfile.cs4331
root@kali:~/.Documents/Lab3# openssl dgst -sha512 -verify pubkey.pem -signature myfile.cs4331.sha512 myfile.cs4331
Verification Failure
root@kali:~/.Documents/Lab3#
```

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