

OpenSSL - Introduction

[OpenSSL documentation \(man openssl\)](#) - ([openssl.pdf](#))

NOTE: *For most openssl commands you should first do the following:*

- > `cp /home/cs772/randomfile .`
- > `setenv RANDFILE randomfile`

Message Digest ([man dgst](#))

- > `more file1.txt`
Hussein Wahab
Old Dominion University
- > `openssl dgst -sha1 file1.txt`

SHA1(file.txt)= 439d855153b88dff064af44cc7794026bad31a45
- > `more file2.txt`
hussein Wahab
Old Dominion University
- > `openssl dgst -sha1 file2.txt`
SHA1(file.txt)= 3204dcbee726eb0319b3094f3947539b5e15c969
- > `diff file1.txt file2.txt`
1c1
< Hussein Wahab

> hussein Wahab

*This shows that small difference (H vs. h) completely changes the digest.
To record the digest of a file us:*

```
> openssl dgst -sha1 -out digest_file1.txt file1.txt
```

```
> cat digest_file1.txt
```

```
SHA1(file1.txt)= 439d855153b88dff064af44cc7794026bad31a45
```

Or you can use:

```
> openssl dgst -sha1 file1.txt > digest_file1.txt
```

Public Key Cryptography (Asymmetric) (man genrsa) & (man rsa)

➤ Generating RSA keys

```
> openssl genrsa -out rsaprivatekey.pem -des3 1024
```

This generate the private key and store it encrypted (using password)

```
Generating RSA private key, 1024 bit long modulus
```

```
.....++++++
```

```
.....++++++
```

```
e is 65537 (0x10001)
```

```
Enter PEM pass phrase:
```

```
Verifying password - Enter PEM pass phrase:
```

```
> openssl rsa -in rsaprivatekey.pem -pubout -out rsapublickey.pem
```

This generate the corresponding public key if the correct password is provided.

```
read RSA key
```

```
Enter PEM pass phrase:
```

```
writing RSA key
```

➤ Signing/Verifying message digest with RSA

```
> openssl dgst -sha1 -sign rsaprivatekey.pem -out  
mdrsasign_file1.cipher file1.txt
```

Enter PEM pass phrase:

```
> openssl dgst -sha1 -verify rsapublickey.pem -signature  
mdrsasign_file1.cipher file1.txt
```

Verified OK

Change one char in file1.txt

```
> openssl dgst -sha1 -verify rsapublickey.pem -signature  
mdrsasign_file1.cipher file1.txt
```

Verification Failure

NOTE: *file1.txt can be as large as you like, since you are signing the digest.*

➤ Message Encryption/Decryption with RSA ([man rsautl](#))

```
> openssl rsautl -encrypt -pubin -inkey rsapublickey.pem -in msg1.txt -  
out msg1.cipher
```

```
> openssl rsautl -decrypt -inkey rsaprivatekey.pem -in msg1.cipher -out  
msg1.txt
```

Enter PEM pass phrase:

NOTE: *msg1.txt has to be small (<=1024 bits or 128 bytes, the length of the RSA key), since you are encrypting/decryption the file itself, not its digest.*

➤ Message Signature/Verification with RSA

```
> openssl rsautl -sign -inkey rsaprivatekey.pem -in msg2.txt -out  
msg2.cipher
```

```
> openssl rsautl -verify -pubin -inkey rsapublickey.pem -out msg2.txt -in  
msg2.cipher
```

NOTE: *msg2.txt has to be small, since you are encrypting/decryption the file itself.*

Secret Key Cryptography (Symmetric) (man enc)

➤ Encrypt (-e):

```
> openssl enc -des3 -e -salt -a -in file1.txt -out file1cipher.base64
```

enter des-ede3-cbc encryption password:

Verifying password - enter des-ede3-cbc encryption password:

➤ Decrypt (-d):

```
> openssl enc -des3 -d -salt -a -out file1.txt -in file1cipher.base64
```

enter des-ede3-cbc decryption password:

Base64 Encode/Decode

In the above, we can encrypt/decrypt *without* `-a` option to produce `file1cipher`,

then we can use the following to encode/decode to/from base64.

```
> openssl enc -des3 -e -salt -in file1.txt -out file1cipher
```

➤ Encode to base64

To encode file1cipher to file1cipher.base64:

```
> openssl enc -base64 -e -out file1cipher.base64 -in file1cipher
```

➤ Decode to base64

To decode file1cipher.base64 to file1cipher:

```
> openssl enc -base64 -d -in file1cipher.base64 -out file1cipher
```

```
> openssl enc -des3 -d -salt -out file1.txt -in file1cipher
```