

---

***Title:*** *Cryptography Experiment Report*

---

**Task:** Task 6 – Introduction to Cryptography

**Internship:** Cyber Security Internship

**Name:** Adil Firoz

**Date:** 23-01-2026

**Operating System:** Kali Linux

**Tools Used:** OpenSSL, Linux Terminal

## **1. Introduction**

The objective of this task is to understand the fundamentals of cryptography and perform basic cryptographic operations using OpenSSL on Kali Linux.

In this task, symmetric encryption, asymmetric encryption, hashing, and digital signature verification were implemented. These cryptographic techniques are widely used to secure data, verify integrity, and authenticate users in modern cybersecurity systems.

## **2. Basic Network Connectivity Commands**

Basic network diagnostic commands were executed to verify internet connectivity and DNS resolution.

The following commands were used:

- ping to test network reachability
- curl to retrieve data from HTTP and HTTPS websites
- dig to query DNS servers

These commands help in verifying network availability before performing cryptographic operations.

```

(root@kali)-[/home/safe]
# ping -4 -c 4 google.com
PING google.com (142.250.183.14) 56(84) bytes of data.
64 bytes from bom07s30-in-f14.1e100.net (142.250.183.14): icmp_seq=1 ttl=255 time=47.2 ms
64 bytes from bom07s30-in-f14.1e100.net (142.250.183.14): icmp_seq=2 ttl=255 time=48.0 ms
64 bytes from bom07s30-in-f14.1e100.net (142.250.183.14): icmp_seq=3 ttl=255 time=133 ms
64 bytes from bom07s30-in-f14.1e100.net (142.250.183.14): icmp_seq=4 ttl=255 time=46.1 ms

--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 46.140/68.567/133.009/37.211 ms

```

AND

```

(root@kali)-[/home/safe]
# dig google.com @8.8.8.8

; <<>> DiG 9.20.15-2-Debian <<>> google.com @8.8.8.8
;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 18740
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;google.com.                IN      A

;; ANSWER SECTION:
google.com.                 5       IN      A      142.250.183.14

;; Query time: 36 msec
;; SERVER: 8.8.8.8#53(8.8.8.8) (UDP)
;; WHEN: Fri Jan 23 17:53:09 IST 2026
;; MSG SIZE rcvd: 55

```

Figure 1: Network connectivity and DNS testing using ping, curl, and dig

### 3. AES File Encryption and Decryption

#### 3.1 File Creation

```

(root@kali)-[/home/safe]
# echo "This is my secret file" > secret.txt

```

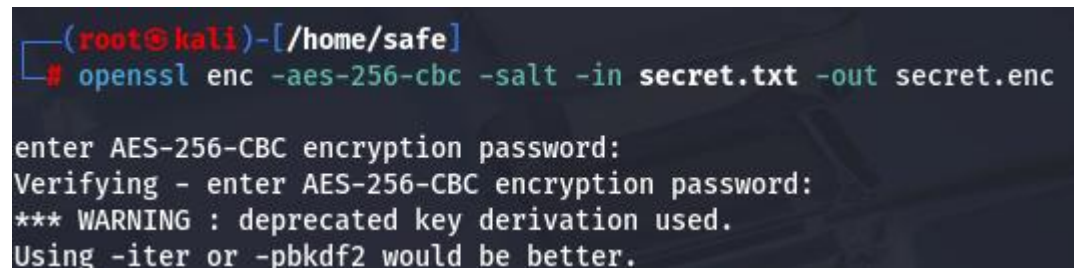
Figure 2: Creation of a secret text file

### 3.2 File Encryption Using AES-256

The file was encrypted using AES-256-CBC symmetric encryption with OpenSSL.

A password was used to protect the encrypted file.

AES is a widely used symmetric encryption algorithm due to its strong security and high performance.



```
(root@kali)-[/home/safe]
# openssl enc -aes-256-cbc -salt -in secret.txt -out secret.enc

enter AES-256-CBC encryption password:
Verifying - enter AES-256-CBC encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
```

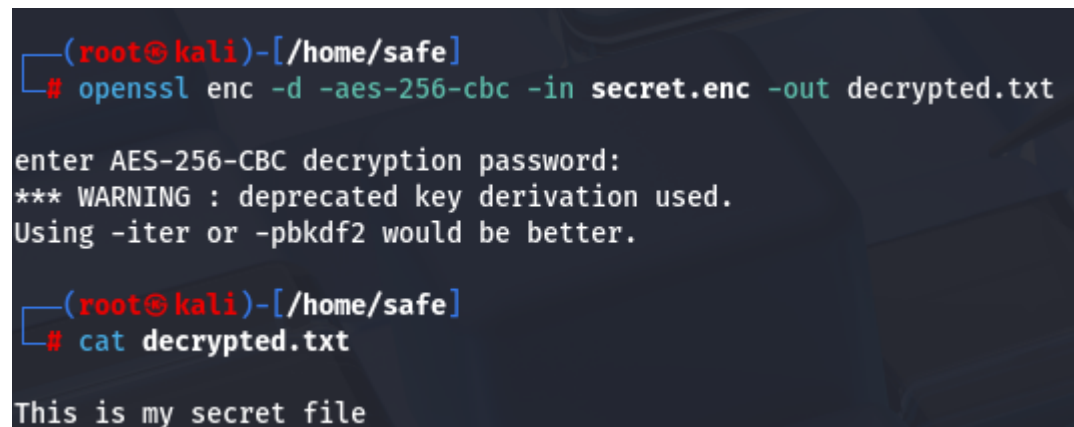
Figure 3: AES-256 encryption using OpenSSL

### 3.3 File Decryption

The encrypted file was decrypted using the same password.

After decryption, the original file content was successfully recovered.

This confirms that AES encryption and decryption were performed correctly.



```
(root@kali)-[/home/safe]
# openssl enc -d -aes-256-cbc -in secret.enc -out decrypted.txt

enter AES-256-CBC decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.

(root@kali)-[/home/safe]
# cat decrypted.txt

This is my secret file
```

Figure 4: AES decryption and recovered file content

## 4. RSA Key Pair Generation

An RSA key pair was generated using OpenSSL.

A 2048-bit private key and a corresponding public key were created.

RSA is an asymmetric encryption algorithm commonly used for secure key exchange and digital signatures.

```
(root@kali)-[/home/safe]
# openssl genrsa -out private.pem 2048
```

Figure 5: RSA private key generation

```
(root@kali)-[/home/safe]
# openssl rsa -in private.pem -pubout -out public.pem
writing RSA key
```

Figure 6: RSA public key generation

## 5. Digital Signature Creation and Verification

A digital signature was generated using the private key and applied to the secret file.

The signature was then verified using the public key.

The verification result displayed “Verified OK”, confirming file authenticity and integrity.

Digital signatures ensure authentication, integrity, and non-repudiation.

```
(root@kali)-[/home/safe]
# openssl dgst -sha256 -sign private.pem -out sign.bin secret.txt
```

Figure 7: Digital signature creation

```
(root@kali)-[/home/safe]
# openssl dgst -sha256 -verify public.pem -signature sign.bin secret.txt
Verified OK
```

Figure 8: Digital signature verification result

## 6. Hash Generation and Integrity Verification

### 6.1 Initial Hash Generation

```
(root@kali)-[/home/safe]
# sha256sum secret.txt
6224defe0232b56c3855e5632c97a69b560e1e613e6adb26543e17d616f58356 secret.txt
```

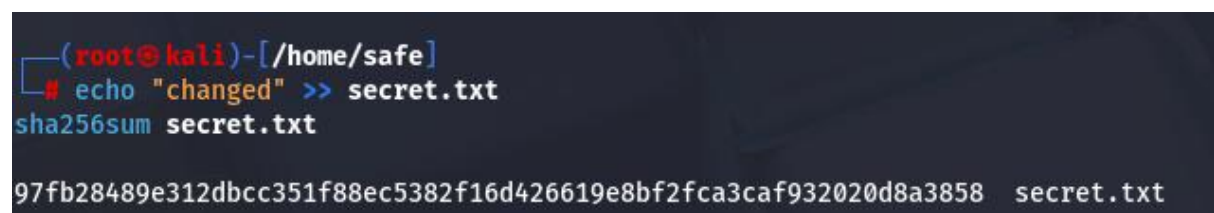
Figure 9: SHA-256 hash of original file

## 6.2 File Modification and Hash Comparison

After modifying the file, a new hash was generated.

The new hash value was different from the original hash.

This demonstrates that hashing can detect even a small change in data and is used for integrity verification.

A terminal window with a dark background. The prompt is (root@kali)-[/home/safe]. The user enters # echo "changed" >> secret.txt. Then they enter sha256sum secret.txt. The output is 97fb28489e312dbcc351f88ec5382f16d426619e8bf2fca3caf932020d8a3858 secret.txt.

```
(root@kali)-[/home/safe]
# echo "changed" >> secret.txt
sha256sum secret.txt
97fb28489e312dbcc351f88ec5382f16d426619e8bf2fca3caf932020d8a3858 secret.txt
```

Figure 10: SHA-256 hash after file modification

## 7. Comparison of Cryptographic Techniques

AES is a symmetric encryption algorithm used for fast and secure data encryption.

RSA is an asymmetric encryption algorithm used for secure key exchange and authentication.

SHA-256 is a hashing algorithm used to verify data integrity.

Digital signatures provide authentication, integrity, and non-repudiation.

## 8. Real-World Applications of Cryptography

Cryptography is widely used in real-world applications such as:

- HTTPS for secure web communication
- VPNs for encrypted network tunnels
- Digital certificates for identity verification
- Secure file storage systems
- Authentication mechanisms in operating systems

## **9. Key Observations**

- Symmetric encryption is fast and suitable for large data encryption
- Asymmetric encryption is essential for secure key exchange
- Hashing effectively detects file modifications
- Digital signatures verify authenticity and integrity
- OpenSSL provides reliable cryptographic operations

## **10. Conclusion**

This task provided hands-on experience with fundamental cryptographic operations using OpenSSL on Kali Linux. AES encryption, RSA key generation, digital signature verification, and hashing were successfully implemented.

These cryptographic techniques form the foundation of secure communication and data protection systems used in modern cybersecurity.