# IDATT2503 - Cryptography Assigment 2

Edvard Berdal Eek

October 2025

## Task 1 - AES a little history

1. Describe how the processes of establishing the two standards differ.

   Data Encryption Standard (DES) was the standard for encryption before AES. The process of establishing DES was largely closed. It was developed by IBM (based on Lucifer) and modified with input from the NSA before being adopted as a U.S. standard in 1977. In contrast, AES was chosen through an open, international competition organized by NIST from 1997–2000, where cryptographers worldwide submitted and analyzed algorithms publicly.

2. What is the name of the algorithm that is known as AES and who developed this algorithm? Why was it by many a surprise that it was this that was chosen to be the new standard?

   The algorithm selected as AES is called **Rijndael**, developed by Joan Daemen and Vincent Rijmen from Belgium. Its selection surprised many because it was a non-U.S. submission from two relatively unknown academics, chosen over candidates from major companies and institutions.

## Task 2

a) For each cipher above, encrypt both x1 and x2, using the key K1 and compare the results, with regards to diffusion.

```
Ciphertexts (for verification):
OTP_K1_x1: 0023456789ABCDEF0123456789ABCDEF
OTP_K1_x2: 0323456789ABCDEF0123456789ABCDEF
OTP_K2_x1: 1023456789ABCDEF0123456789ABCDEF
Affine_K1_x1: F023456789ABCDEF0123456789ABCDEF
Affine_K1_x2: DF23456789ABCDEF0123456789ABCDEF
Affine_K2_x1: 0023456789ABCDEF0123456789ABCDEF
```

Figure 1: Enter Caption

A XOR

```python
def xor_bytes(a: bytes, b: bytes) -> bytes:
    return bytes(x ^ y for x, y in zip(a, b))
```

B Affine Cipher

```
def affine_encrypt(key: bytes, plaintext: bytes) -> bytes:
    a = int.from_bytes(key, "big")
    b = a
    x = int.from_bytes(plaintext, "big")
    y = (a * x + b) % (1 << 128)
    return y.to_bytes(16, "big")
```

Result below:

```
Ciphertexts (for verification):
OTP_K1_x1: 0023456789ABCDEF0123456789ABCDEF
OTP_K1_x2: 0323456789ABCDEF0123456789ABCDEF
OTP_K2_x1: 1023456789ABCDEF0123456789ABCDEF
Affine_K1_x1: F023456789ABCDEF0123456789ABCDEF
Affine_K1_x2: DF23456789ABCDEF0123456789ABCDEF
Affine_K2_x1: 0023456789ABCDEF0123456789ABCDEF
```

Figure 2: Enter Caption

C One round of AES

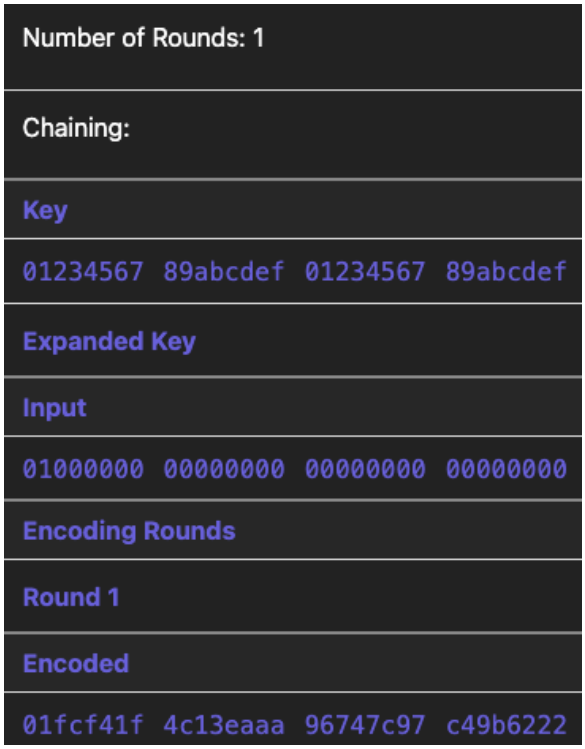


Figure 3: AES one round with $x1$ and $K1$

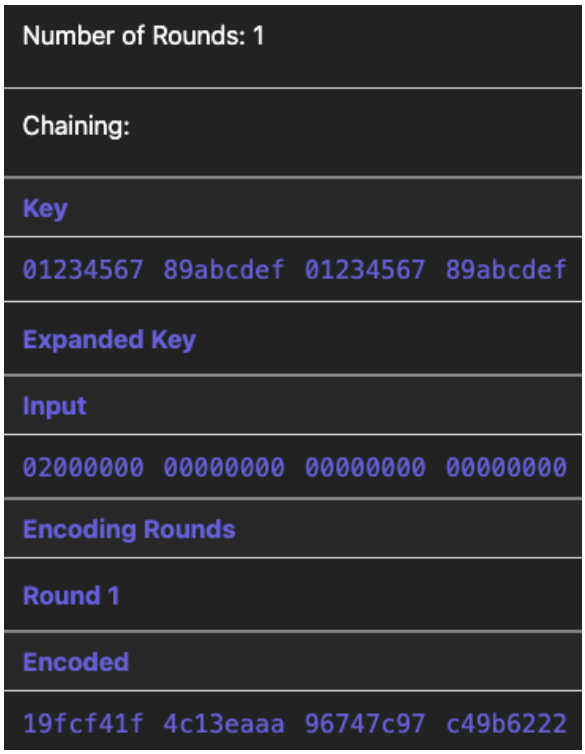


Figure 4: AES one round with $x2$ and $K1$

D Full AES

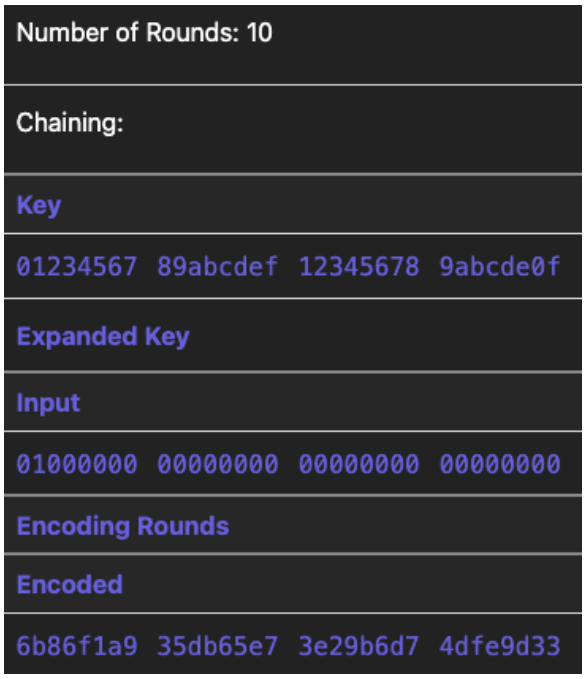


Figure 5: AES with 10 rounds on $x1$ with $K1$

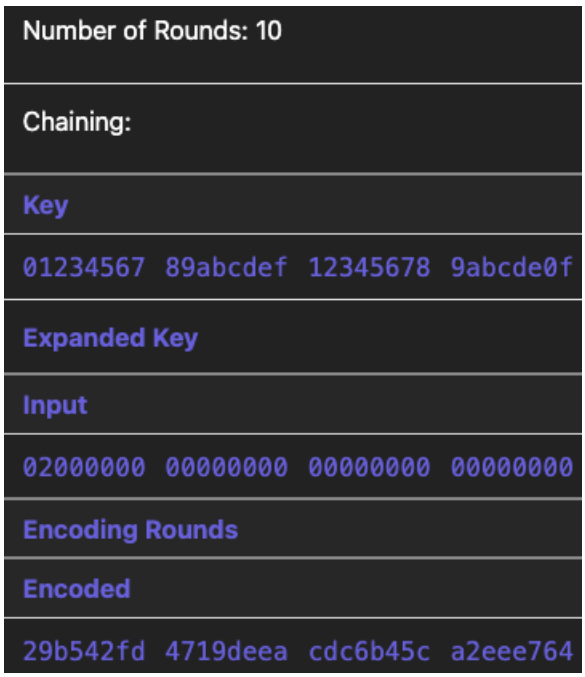


Figure 6: AES with 10 rounds on $x2$ with $K1$

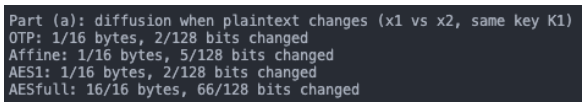


Figure 7: Byte and bits comparison part a)

b) For each cipher, encrypt x1 using K1 and K2, and compare the results. How many bits change?

A XOR

B Affine Cipher

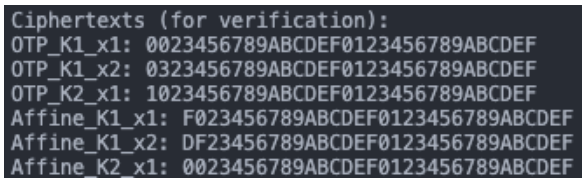


Figure 8: Enter Caption

C One round of AES

AES one round with $x1$ and $K1$ is the same as in precious task.

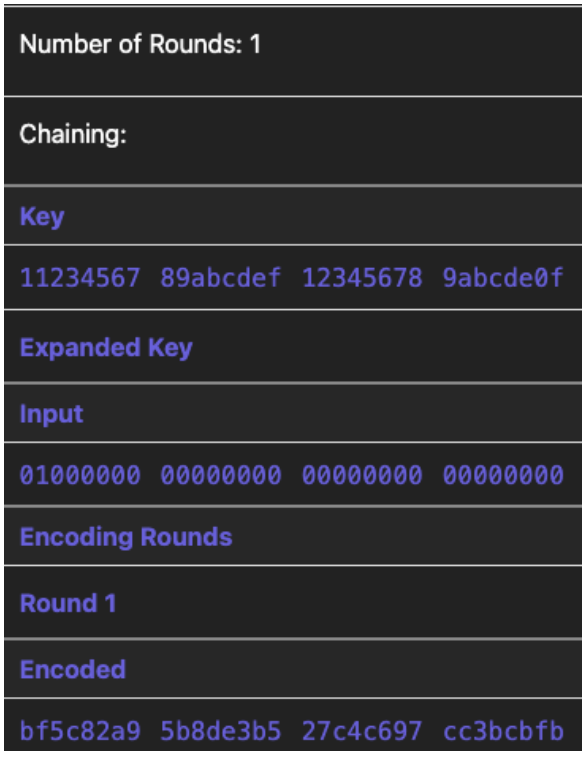


Figure 9: AES one round with $x1$ and $K2$

D Full AES

AES with 10 rounds on $x1$ with $K1$ is the same as in previous task

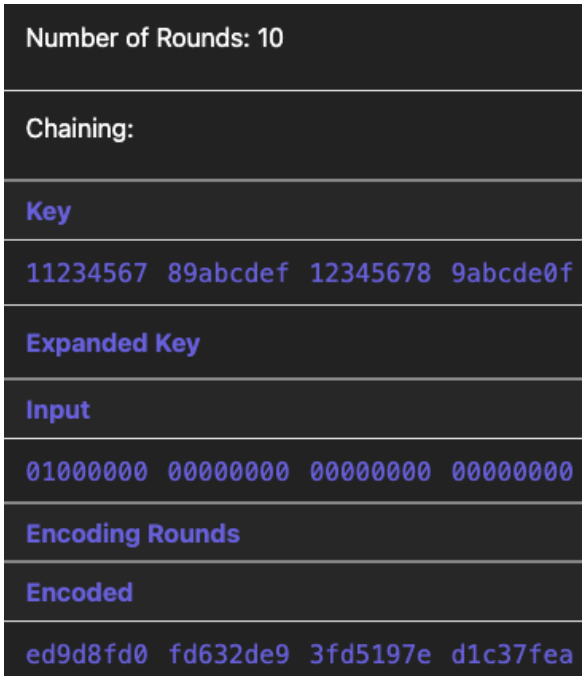


Figure 10: AES with 10 rounds on $x1$ with $K2$

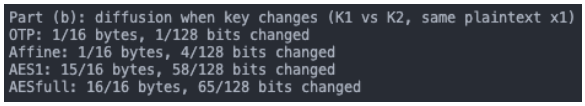


Figure 11: Byte and bits comparison part b)

## Task 3

Do the challenges on https://www.cryptohack.org/challenges/aes/, up to "Bringing it all together". Write down the captured flags, and submit the code (its in Python)

a) Keyed Permutation

**crypto{bijection}**

b) Resisting Bruteforce

**crypto{biclique}**

c) Structure of AES

**crypto{inmatrix}**

d) Round Keys
**crypto{r0undk3y}**

e) Confusion Through Substitution
**crypto{l1n34rly}**

f) Diffusion through Permutation
**crypto{d1ffUs3R}**

g) Bringing It All Together
**crypto{MYAES128}**

# Task 4

In a standard Feistel cipher, decryption is equivalent to encryption with the round keys applied in reverse order. In this cipher, the round keys are mirrored:

$$(k_1, k_2, \ldots, k_8, k_8, \ldots, k_2, k_1),$$

which means that *encryption itself performs the same operations as decryption.*

**Attack using a single oracle query:**

1. Send the encrypted ciphertext $c$ to the encryption oracle.
2. The oracle computes and returns the message $m$

$$\text{Encrypt}(c, k_1, \ldots, k_{16}) = m,$$

because the mirrored keys cause the encryption process to effectively reverse the original encryption steps.

With a single query to the encryption oracle, an attacker can recover the plaintext $m$ corresponding to any ciphertext $c$. This demonstrates that the cipher is vulnerable to a chosen-plaintext attack, as encryption and decryption are effectively identical due to the mirrored key schedule.