

30. Write a python program for CBC MAC of a oneblock message X, say $T = \text{MAC}(K, X)$, the adversary immediately knows the CBC MAC for the two-block message $X || (X \oplus T)$ since this is once again.

```
# -----
# Simple demonstration of the CBC-MAC attack on one-block MAC
# CBC-MAC for one block message:  $T = E(K, X)$ 
# Attacker can instantly compute  $\text{MAC}(K, X || (X \oplus T)) = T$ 
# -----
```

Code:

```
def xor(a, b):
    return bytes([x ^ y for x, y in zip(a, b)])
# Toy block cipher:  $E(K, M) = M \oplus K$  (just for illustration)
def E(K, M):
    return xor(K, M)
def CBC_MAC(K, blocks):
    """CBC-MAC with toy cipher"""
    IV = bytes(len(K)) # zero IV
    state = IV
    for blk in blocks:
        state = E(K, xor(state, blk))
    return state
# Example 1-block message X
X = b"\x10\x20\x30\x40" # 4-byte block
K = b"\xAA\xBB\xCC\xDD" # 4-byte key
# MAC of single-block message
T = CBC_MAC(K, [X])
print("X =", X.hex())
print("K =", K.hex())
```

```

print("T = MAC(K,X) =", T.hex())

# ----

# Attack:

# Attacker computes forged 2-block message:

# M = X || (X XOR T)

# Its CBC-MAC is exactly T again.

# ----

X2 = xor(X, T)           # second block = X ⊕ T

forged_mac = CBC_MAC(K, [X, X2])

print("\nForged 2-block message:")

print("Block 1:", X.hex())

print("Block 2:", X2.hex())

print("MAC(K, X || (X xor T)) =", forged_mac.hex())

print("\nAs expected, forged MAC equals original T.")

```

```

IDLE Shell 3.14.0
File Edit Shell Debug Options Window Help
Python 3.14.0 (tags/v3.14.0:ebf955d, Oct 7 2025, 10:15:03) [MSC v.1944 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.

>>> ===== RESTART: C:/Users/Maria/OneDrive/Documents/ex30.py =====
X = 10203040
K = aabbccdd
T = MAC(K,X) = ba9bfc9d

Forged 2-block message:
Block 1: 10203040
Block 2: aabbccdd
MAC(K, X || (X xor T)) = ba9bfc9d

As expected, forged MAC equals original T.
>>>

```