program for CBC MAC of a oneblock message X, say T = MAC(K, X), the adversary immediately knows the CBC MAC for the two-block message X || (X ⊕ T) since this is once again.

import os

def xor(a,b): return bytes(x^y for x,y in zip(a,b))

def encrypt\_block(key,block): return xor(block,key)

def cbc\_mac(key,blocks):

iv = bytes(len(key))

c = iv

for b in blocks:

c = encrypt\_block(key,xor(b,c))

return c

bs = 8

key = os.urandom(bs)

X = os.urandom(bs)

T = cbc\_mac(key,[X])

M2 = xor(X,T)

T2 = cbc\_mac(key,[X,M2])

print("Key :", key.hex().upper())

print("X :", X.hex().upper())

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

print("Two-block message: X || (X ⊕ T) ->", (X+M2).hex().upper())

print("MAC for two-block message:", T2.hex().upper())

print("Tag matches:", T==T2)

