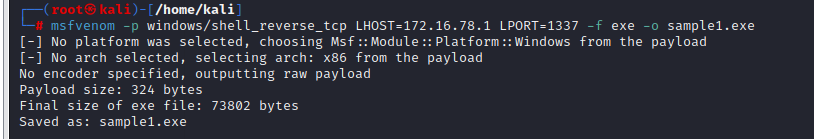
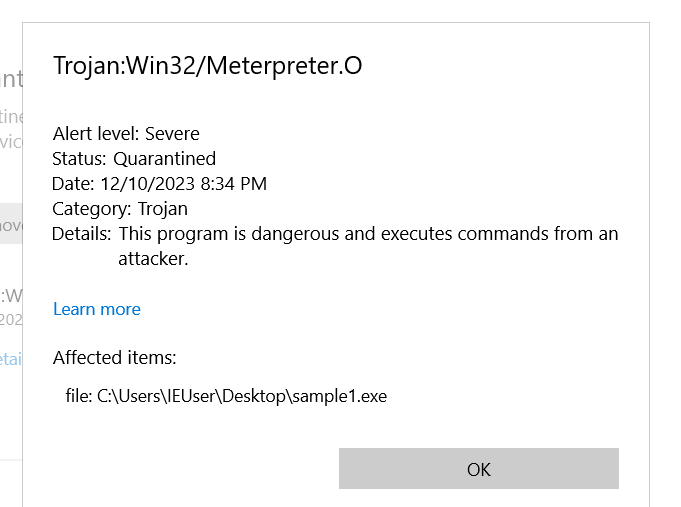
ЗВОРОТНА РОЗРОБКА ТА АНАЛIЗ ШКIДЛИВОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Лабораторна робота №2

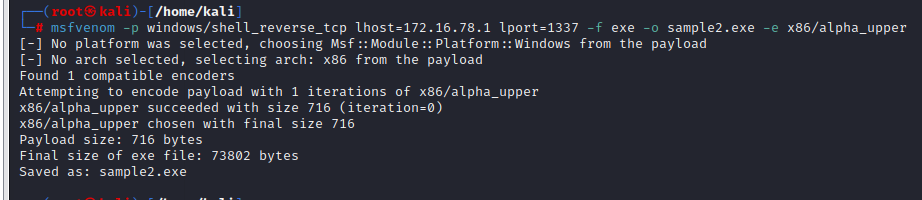
**2.3.1 Обфускацiя коду на прикладi Metasploit Encoders**

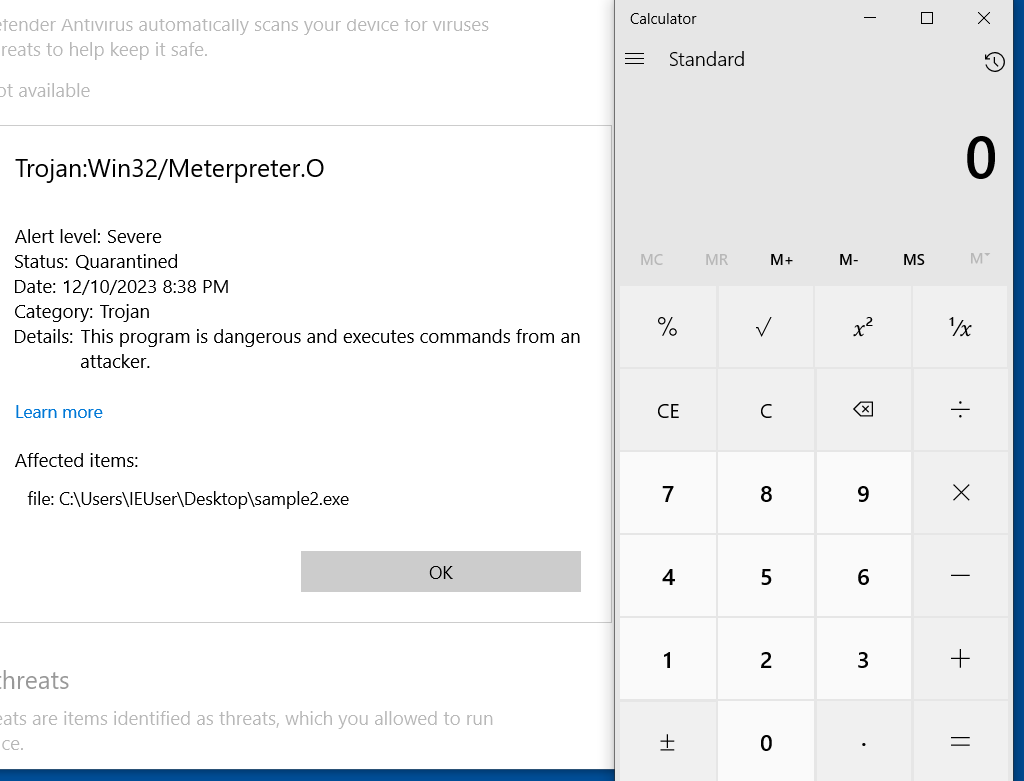


Файл блокується при збереженні

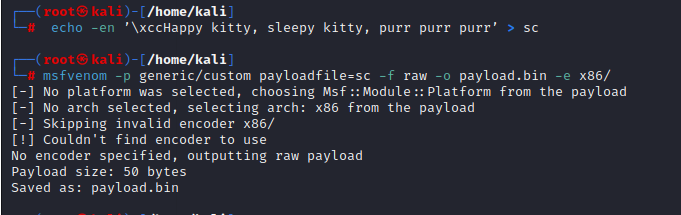


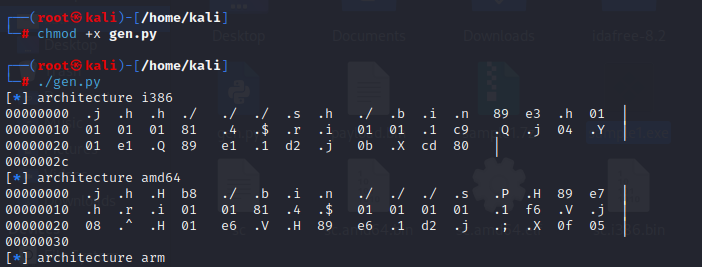
Другий зразок:

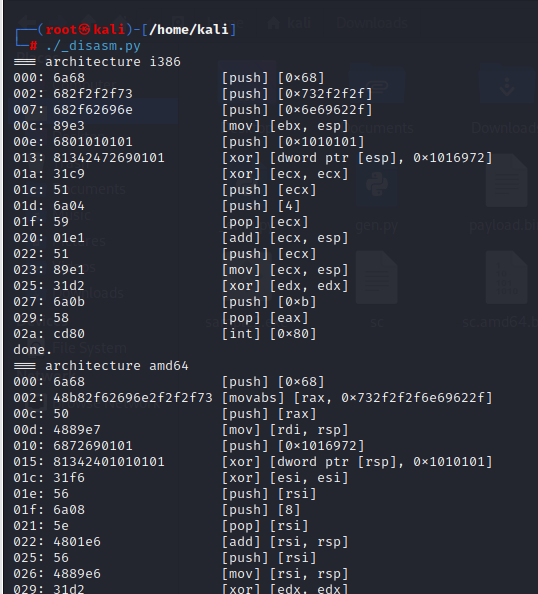




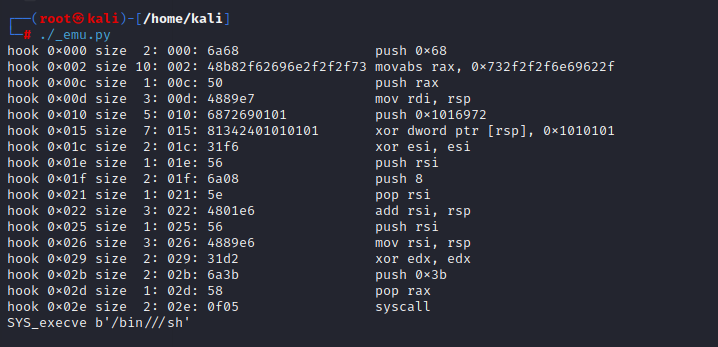
**2.3.2 Статичний аналiз**





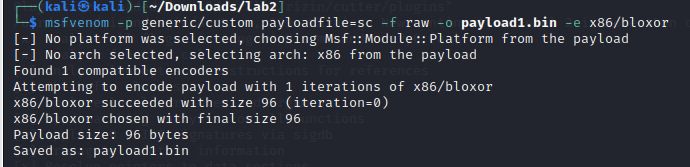


**2.3.3 Динамiчний аналiз**



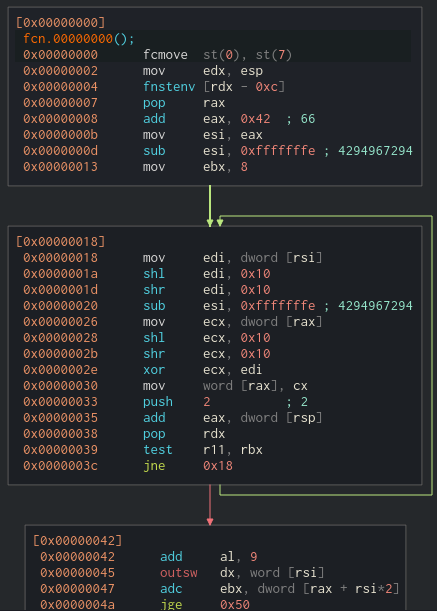
**Реалiзуйте статичний деобфускатор для Вашого варiанту, роздiл 2.3.2.**





відкриємо в cutter

у другому блоці відбувається цикл деобфускації



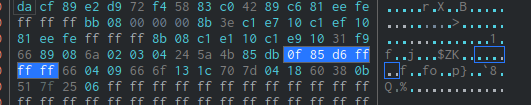
якщо додати вивід



отримуємо байти якими завершується цей цикл



далі іде корисне навантаження



import capstone

import pwn

file\_data = open("/home/kali/Downloads/lab2/payload1.bin", "rb").read()

md = capstone.Cs(capstone.CS\_ARCH\_X86, capstone.CS\_MODE\_32)

signature\_bytes = None

for instruction in md.disasm(file\_data, 0):

if instruction.mnemonic == "jne":

signature\_bytes = instruction.bytes

break

offset = file\_data.find(signature\_bytes)

l = file\_data[offset + len(signature\_bytes):]

decoded\_payload = b""

for i in range(0, len(l) - 2):

decoded\_payload += pwn.xor(l[i], l[i + 2])

decoded\_payload1 = b"".join(bytes([byte]) for byte in decoded\_payload if 32 <= byte <= 126)

print(decoded\_payload1.decode("utf-8"))

