Through the EDR Lens

Taking a look at what an EDR actually sees…

// SteelCon 2023

whoami

Brandon McGrath

Targeted Operations @ TrustedSec

Dev @ pre.empt.blog

https://twitter.com/__mez0__

https://github.com/mez-0

Goal

Setup some common telemetry sources

Review the information returned

Map the detection strategies

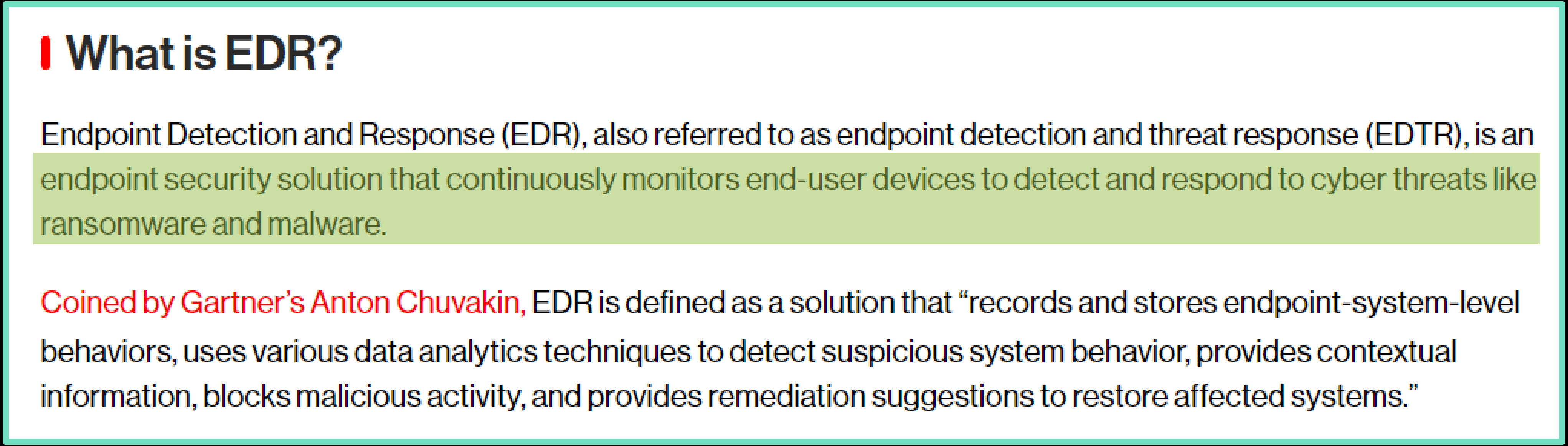
Talk about an example methodology

~~Goal~~

Not demoing bypasses for Twitter

Define:EDR

EDR – What?



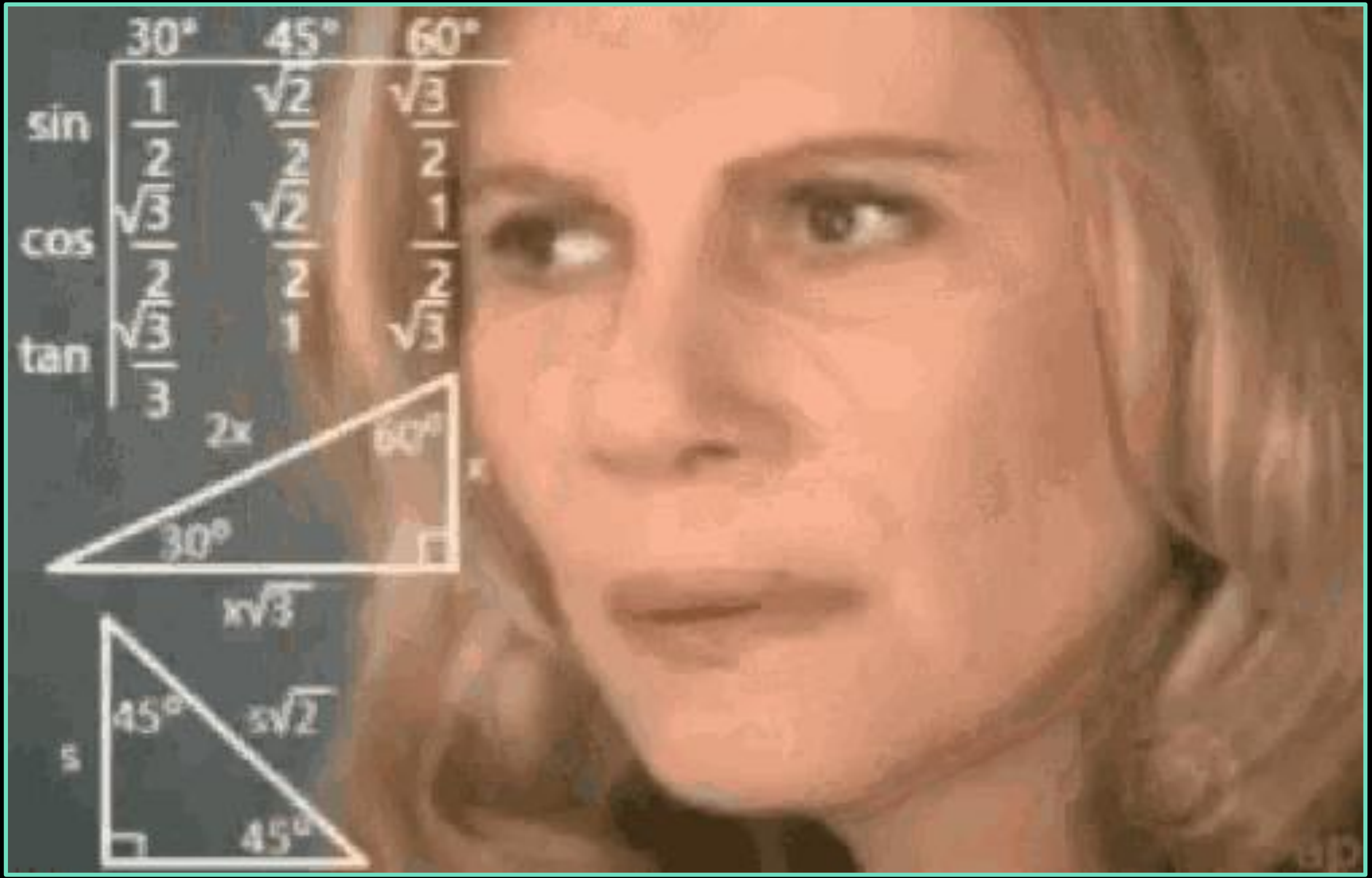
What is EDR?

Endpoint Detection and Response (EDR), also referred to as endpoint detection and threat response (EDTR), is an endpoint security solution that continuously monitors end-user devices to detect and respond to cyber threats like ransomware and malware.

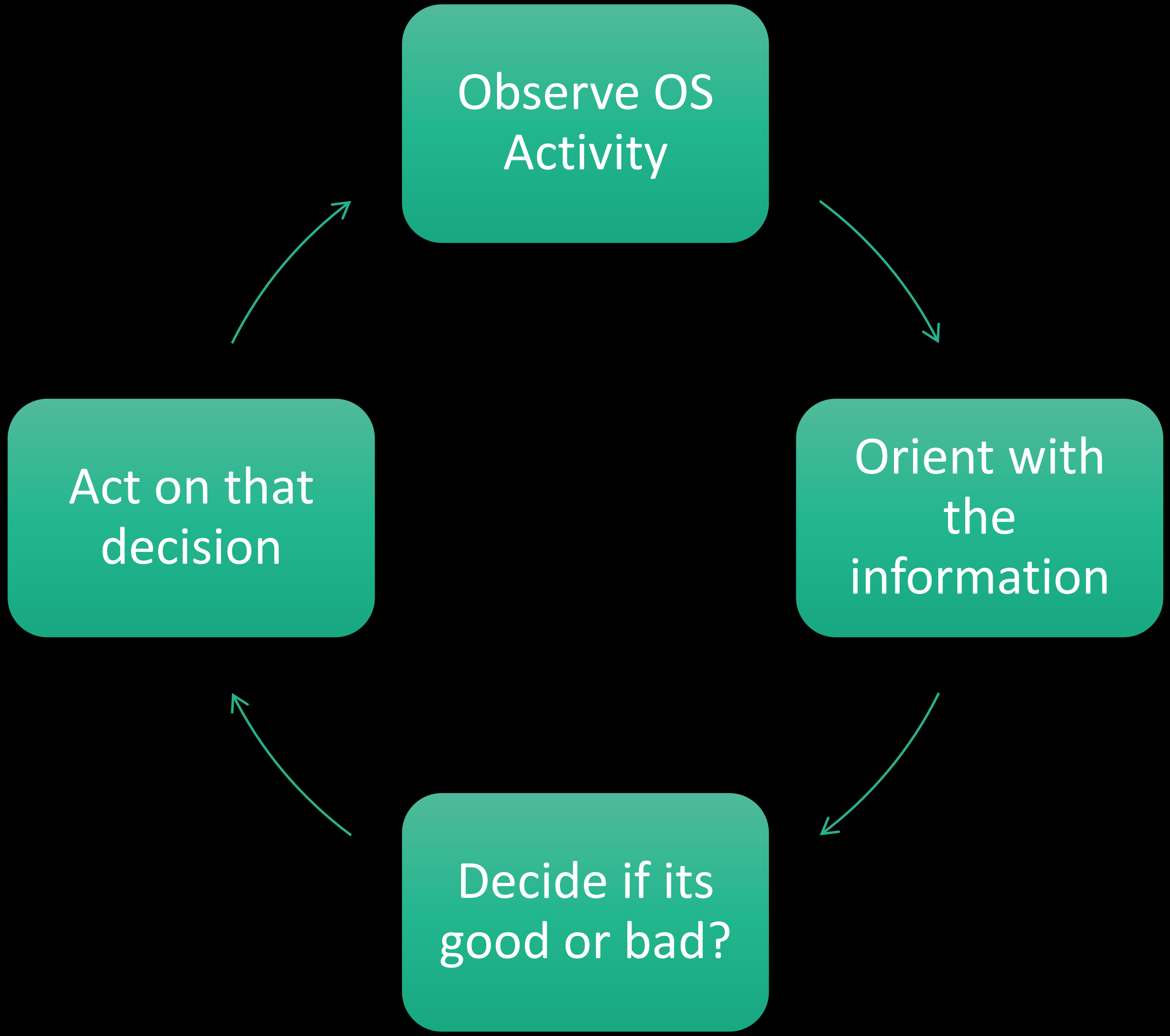
Coined by Gartner's Anton Chuvakin, EDR is defined as a solution that "records and stores endpoint-system-level behaviors, uses various data analytics techniques to detect suspicious system behavior, provides contextual information, blocks malicious activity, and provides remediation suggestions to restore affected systems."

CrowdStrike: What is Endpoint Detection and Response

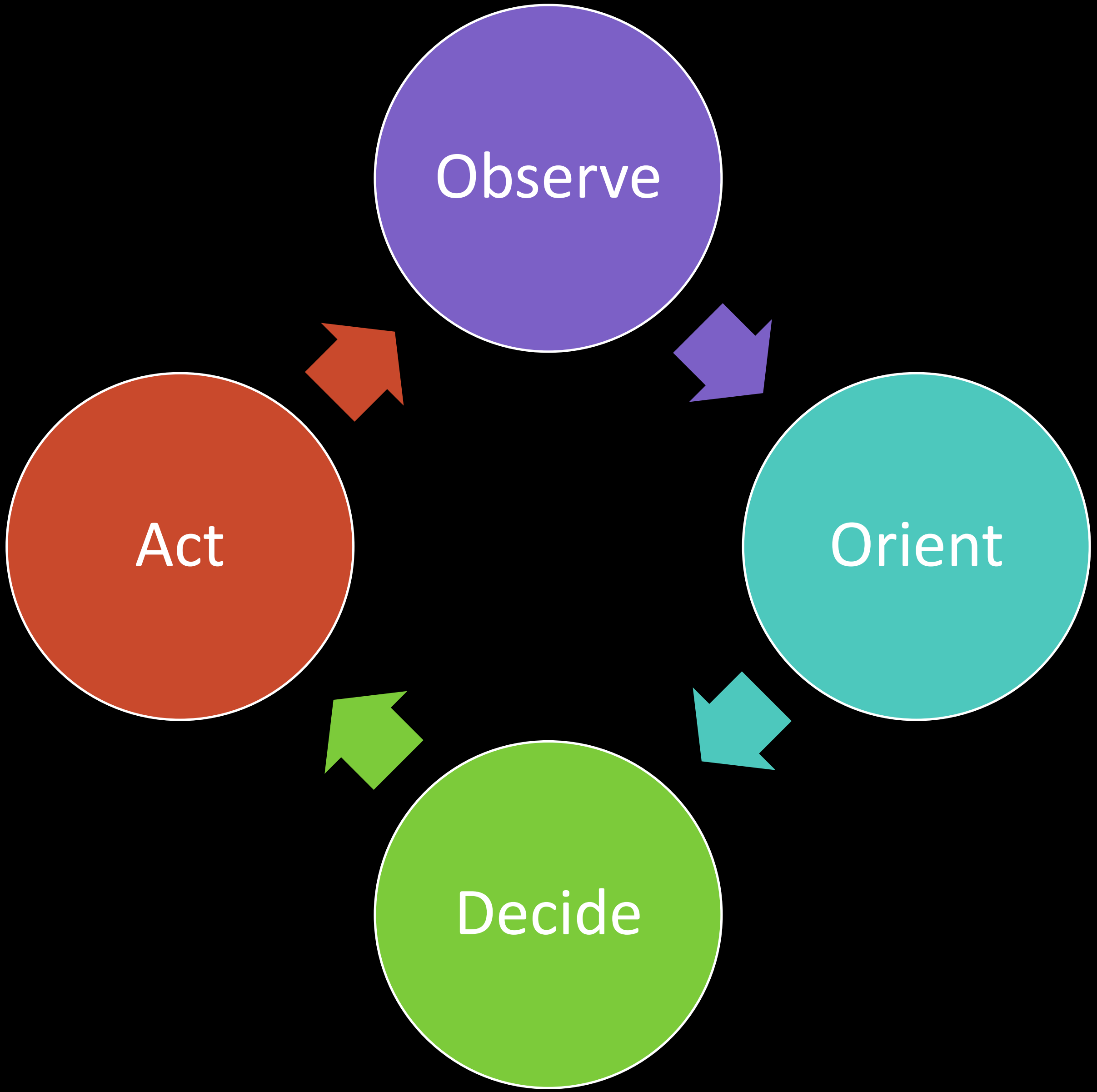
Methodology



The EDRs Job


Observe OS Activity
Orient with the information
Decide if its good or bad?
Act on that decision

OODA



EDR Evasion and OODA Loops

OODA – John Boyd

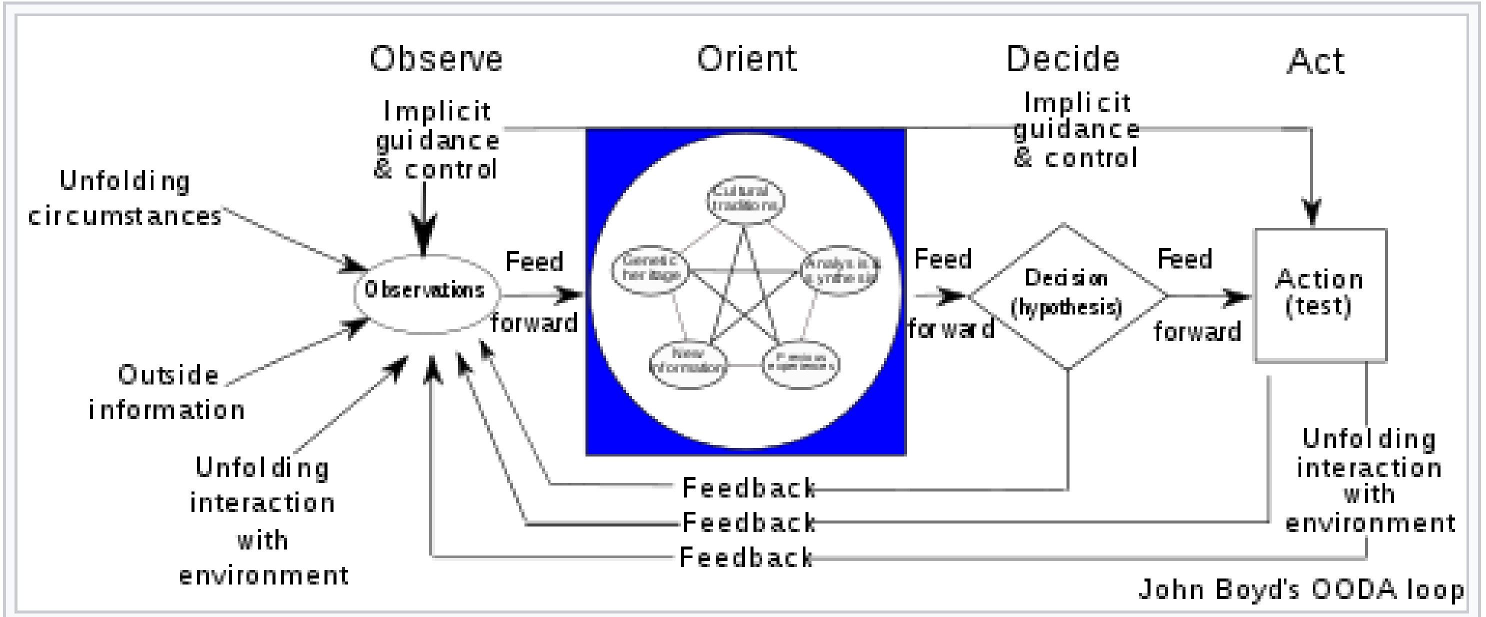
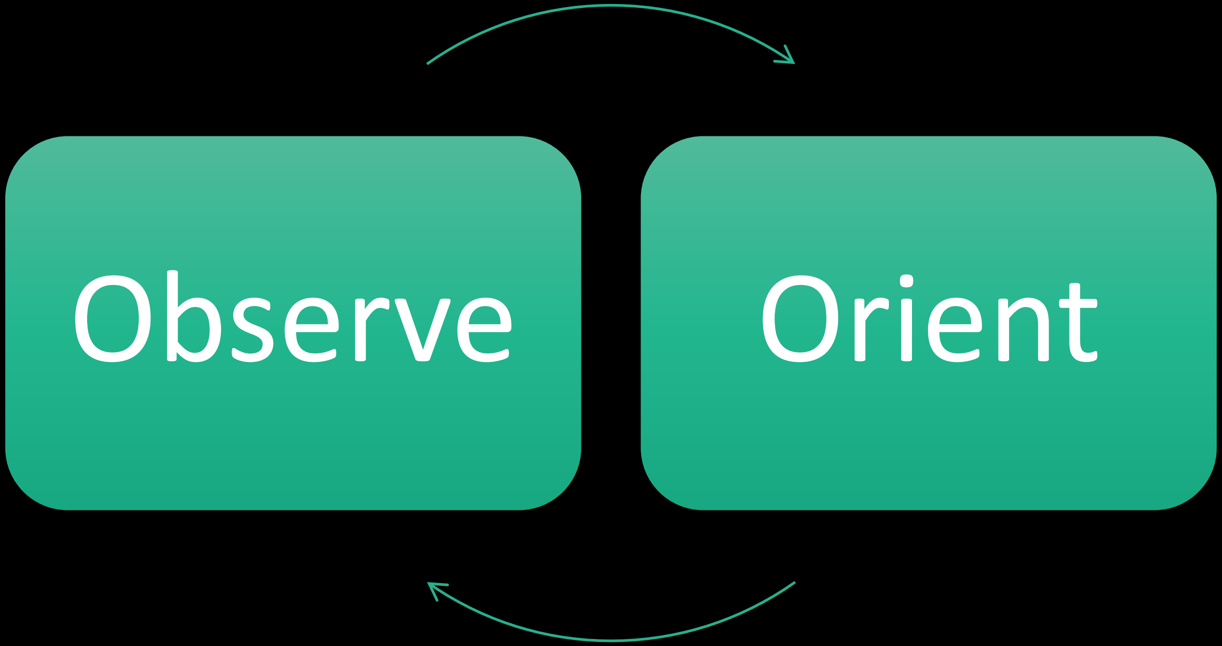


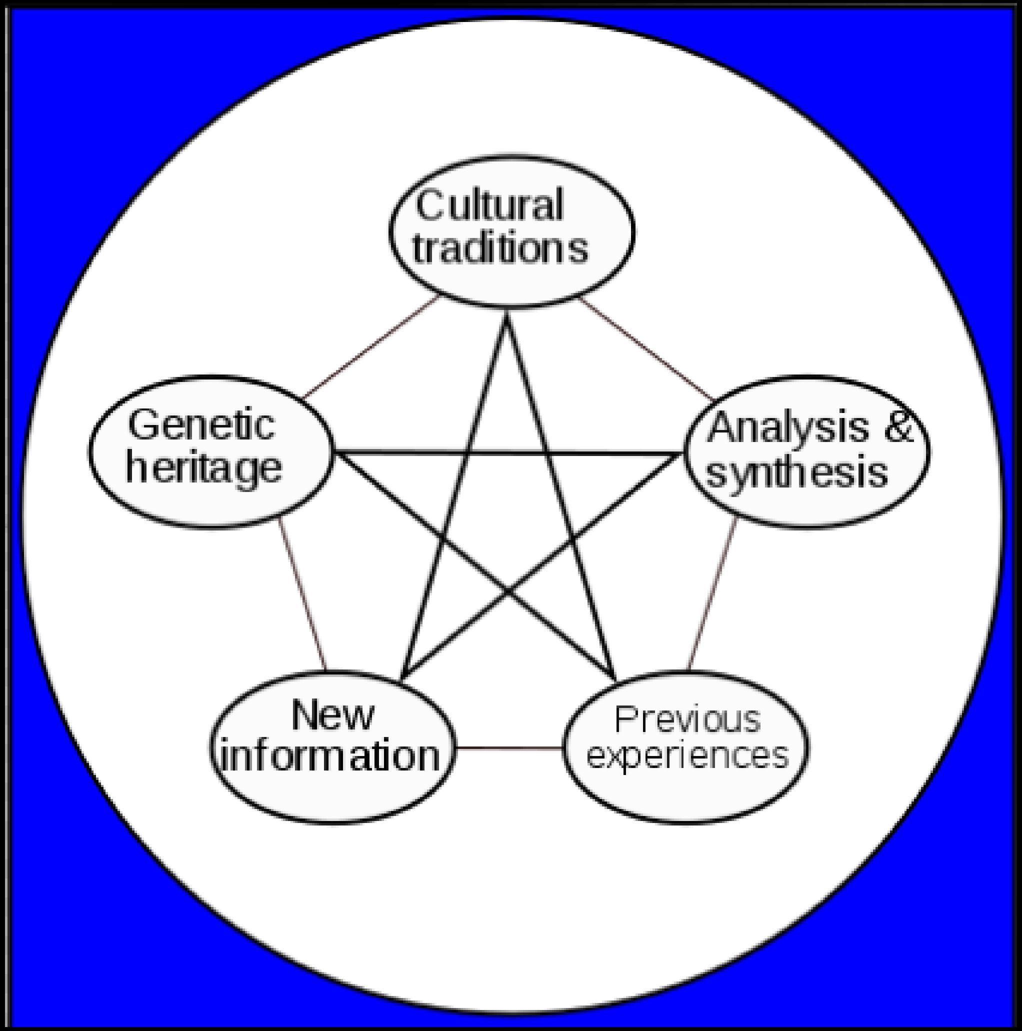
Diagram of the OODA loop

The idea is that an entity can process this cycle quickly, observing and reacting to unfolding events more rapidly than an opponent, can thereby "get inside" the opponent's decision cycle and gain the advantage – **John Boyd**

EDR – OODA



EDR – OODA


Cultural traditions
Genetic heritage
Analysis & synthesis
New information
Previous experiences

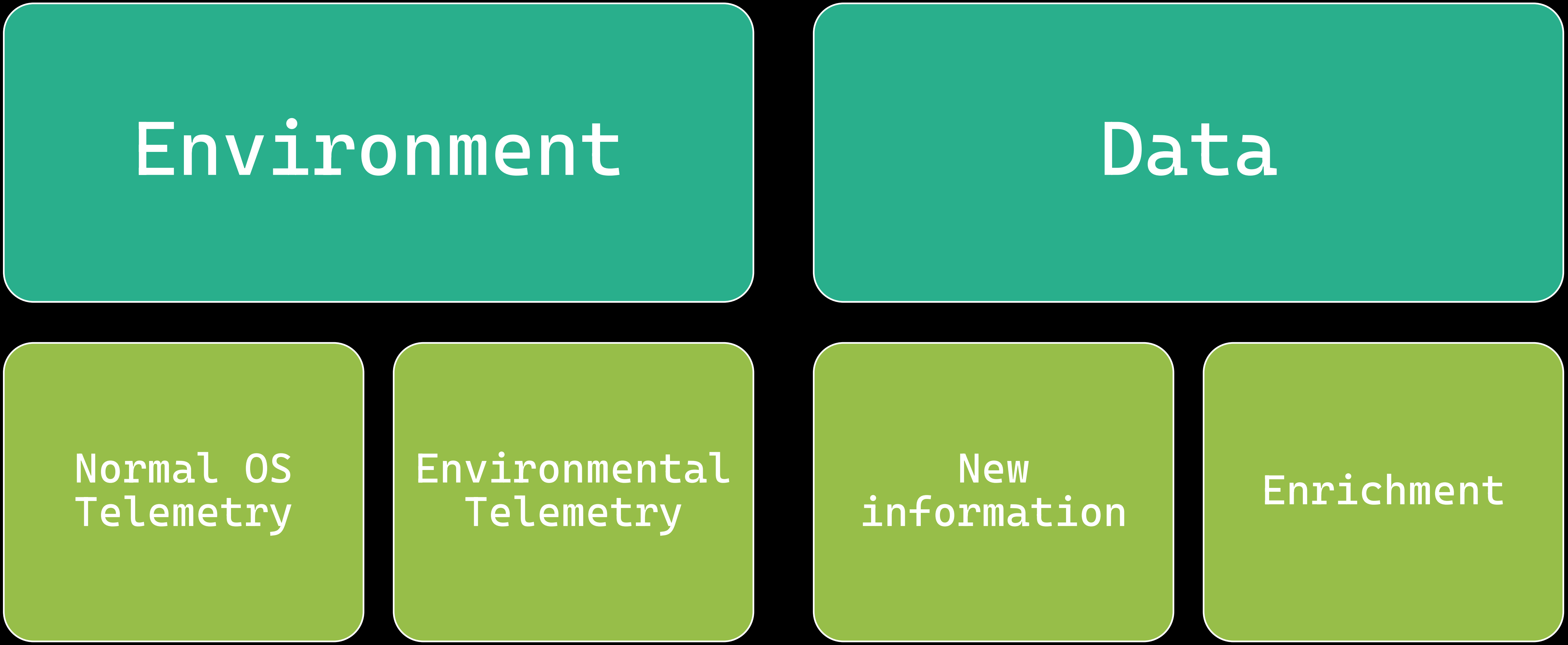
Observe

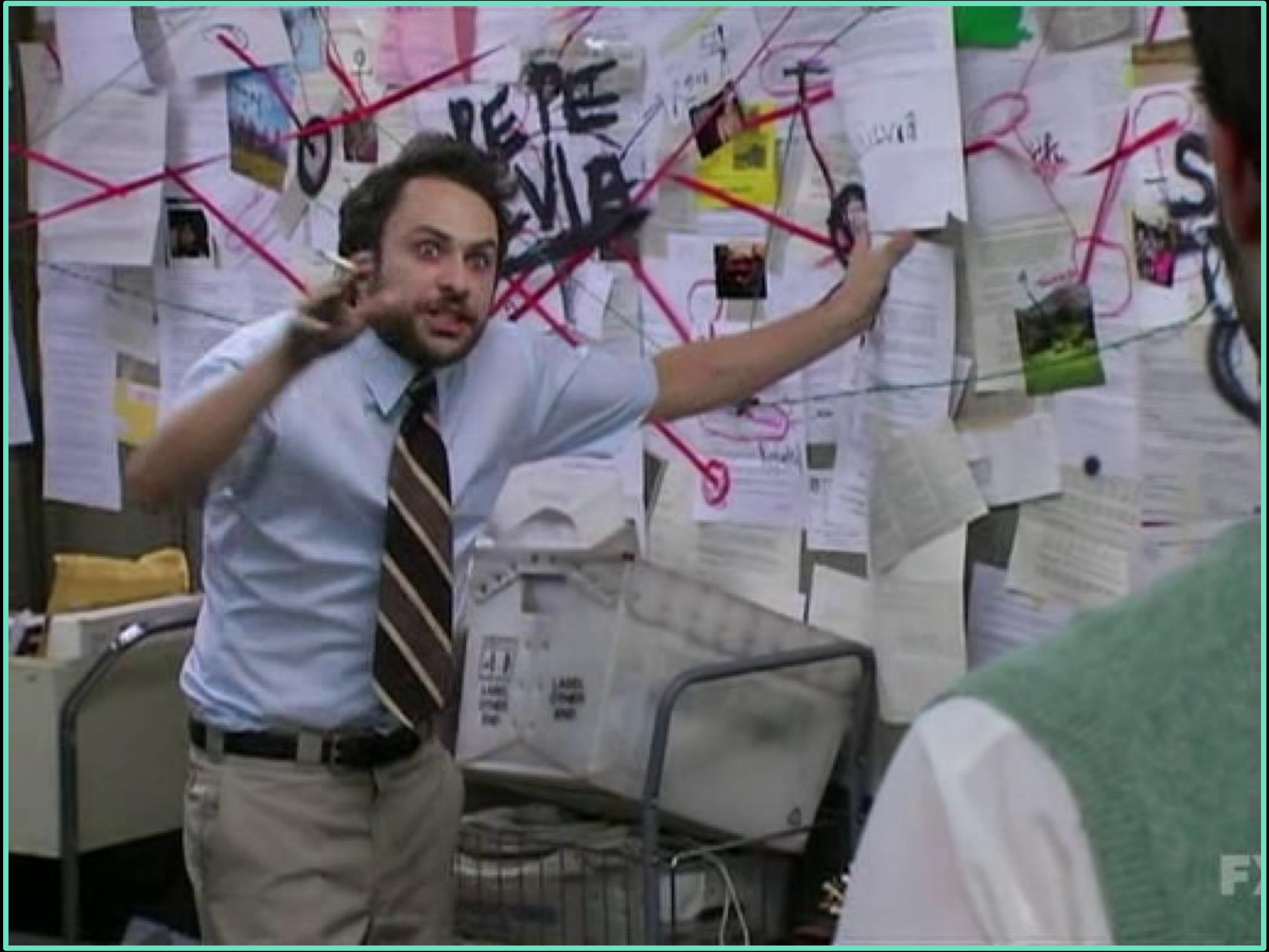
EDR – OODA (Observe)

Observe

- Process Instrumentation
- User-land Hooks
- Kernel Callbacks
- AMSI/ETW(Ti)
- Thread Call Stacks

EDR – OODA (Orient 1)





Telemetry Sources

Overview

Userland DLL

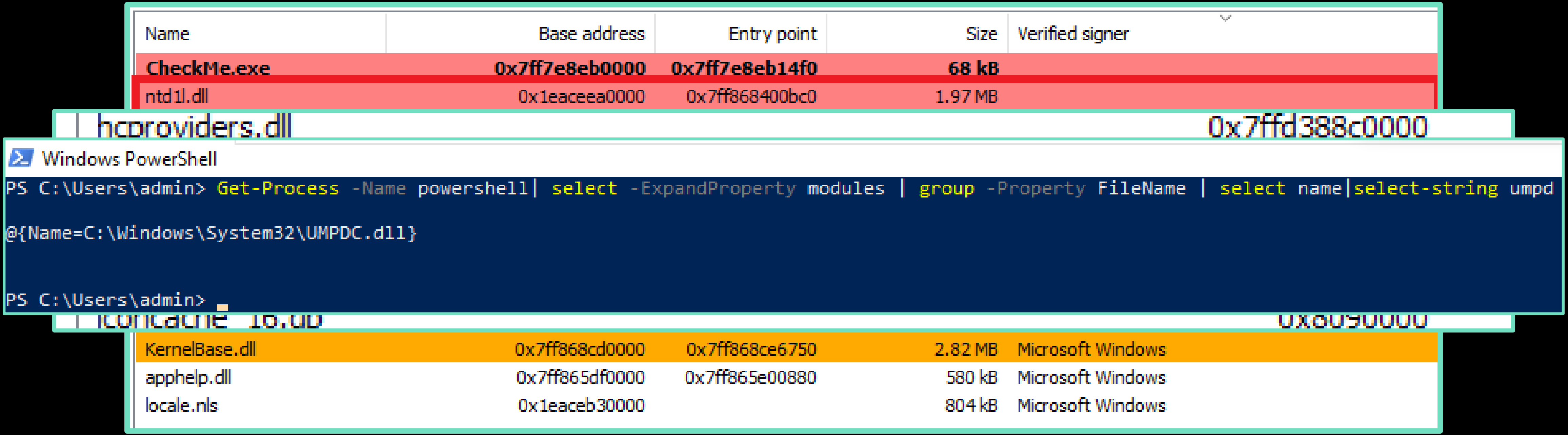
Callstack Analysis

Kernel Callbacks

Userland DLL – Why?

- Easy to apply hooks
- Can assess functionality as the process goes
- Instrumentation
- Stability
- General enrichment

Userland DLL – Examples



Name	Base address	Entry point	Size	Verified signer
CheckMe.exe	**0x7ff7e8eb0000**	**0x7ff7e8eb14f0**	**68 kB**	
ntd1l.dll	0x1eaceea0000	0x7ff868400bc0	1.97 MB	
KernelBase.dll	0x7ff868cd0000	0x7ff868ce6750	2.82 MB	Microsoft Windows
apphelp.dll	0x7ff865df0000	0x7ff865e00880	580 kB	Microsoft Windows
locale.nls	0x1eaceb30000		804 kB	Microsoft Windows

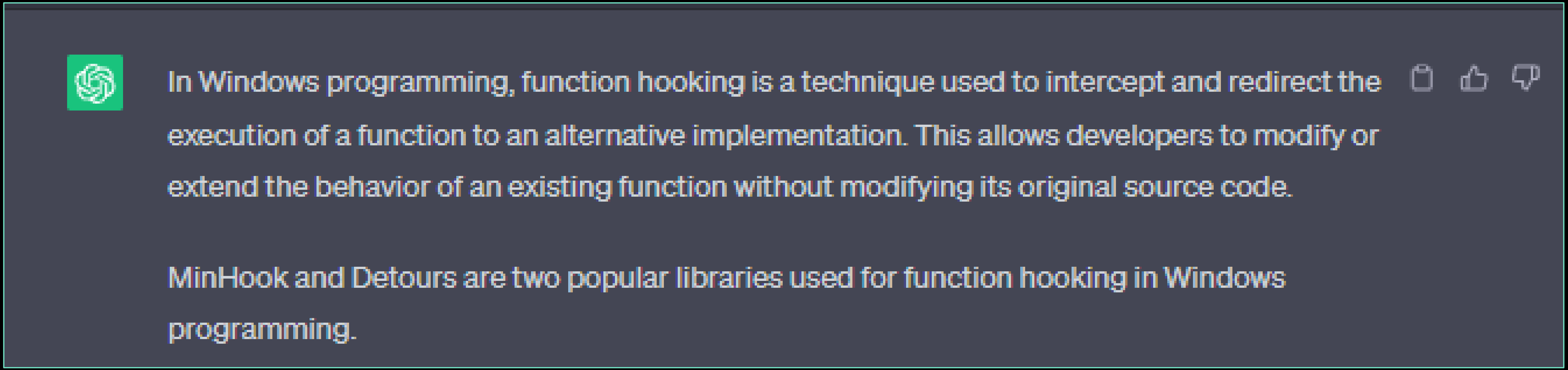
hcproviders.dll 0x7ffd388c0000

Windows PowerShell

```
PS C:\Users\admin> Get-Process -Name powershell| select -ExpandProperty modules | group -Property FileName | select name|select-string umpd

@{Name=C:\Windows\System32\UMPDC.dll}


PS C:\Users\admin> _
```

Implementing Hooks

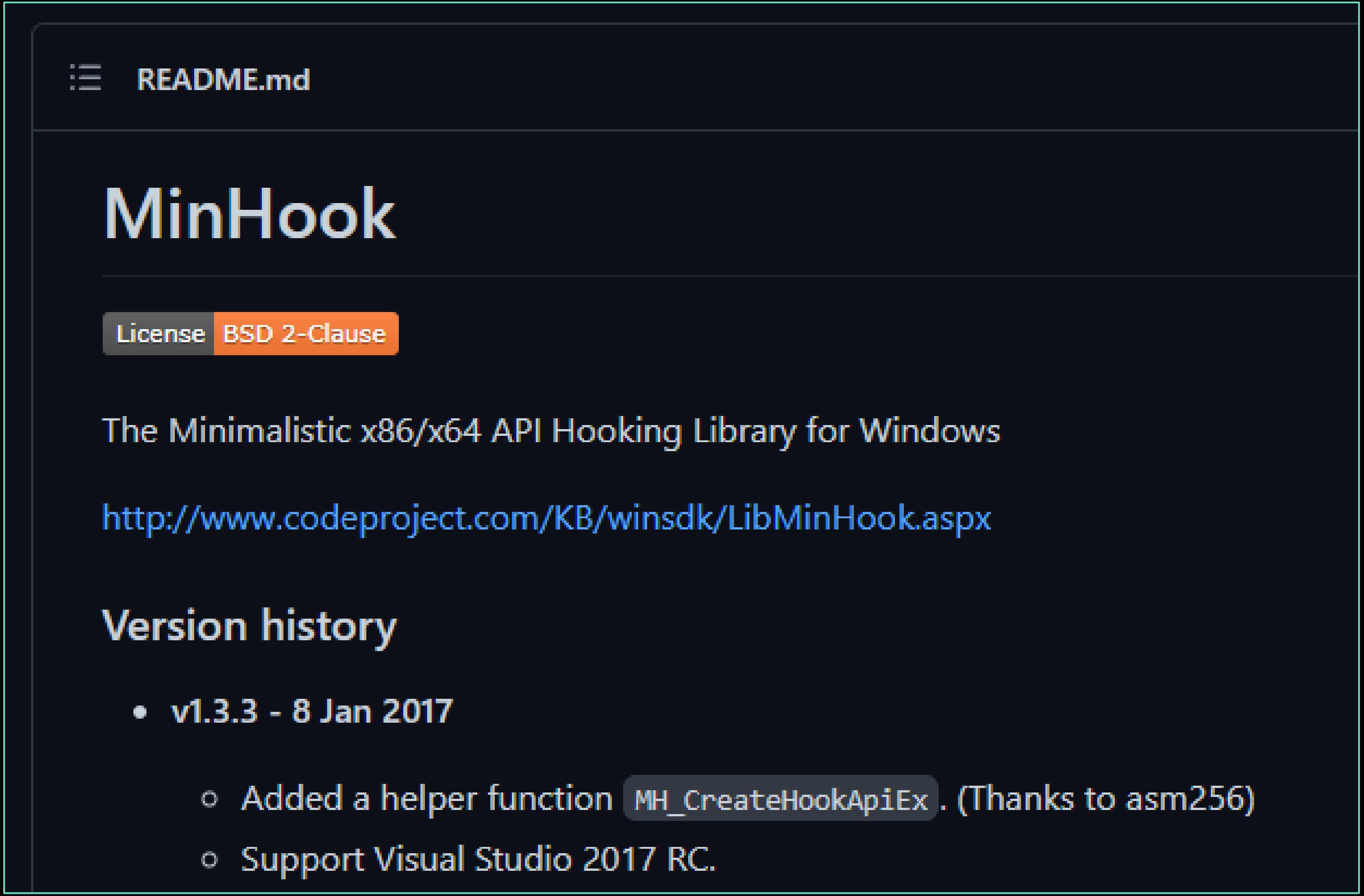
Function Hooking – wtf



In Windows programming, function hooking is a technique used to intercept and redirect the execution of a function to an alternative implementation. This allows developers to modify or extend the behavior of an existing function without modifying its original source code.

MinHook and Detours are two popular libraries used for function hooking in Windows programming.

Function Hooking



README.md

MinHook

License BSD 2-Clause

The Minimalistic x86/x64 API Hooking Library for Windows

http://www.codeproject.com/KB/winsdk/LibMinHook.aspx

Version history

- **v1.3.3 - 8 Jan 2017**
  - Added a helper function `MH_CreateHookApiEx`. (Thanks to asm256)
  - Support Visual Studio 2017 RC.

Userland DLL – Examples

```
DWORD WINAPI SetupHooks(LPVOID param)
{
    MH_STATUS status = MH_STATUS::MH_ERROR_NOT_INITIALIZED;

    status = MH_Initialize();

    status = MH_CreateHookApi(
        L"ntdll",
        "NtAllocateVirtualMemory",
        NtAllocateVirtualMemory_Hook,
        reinterpret_cast<LPVOID*>(&pNtAllocateVirtualMemory_Original
    );

    status = MH_EnableHook(MH_ALL_HOOKS);

    return status;
}
```

Function Hooking

```
NTSTATUS NTAPI NtAllocateVirtualMemory_Hook(IN HANDLE ProcessHandle,
                                            IN OUT PVOID* BaseAddress,
                                            IN ULONG_PTR ZeroBits,
                                            IN OUT PSIZE_T RegionSize,
                                            IN ULONG AllocationType,
                                            IN ULONG Protect
                                            )
{
    /*
        prechecks...
    */


    // Run real function
    NTSTATUS Status = pNtAllocateVirtualMemory_Original(ProcessHandle,
        BaseAddress,
        ZeroBits,
        RegionSize,
        AllocationType,
        Protect
    );

    /*
        postchecks...
    */

    FENNEC::Logger::WriteLogToFile(LOG_TYPE, Log);


    return Status;
}
```

The Log

```
{
  "event": "dll-hook-rwx",
  "data": {
    "parameters": {
      "ProcessHandle": {
        "type": "HANDLE",
        "value": "0xfffff"
      },
      "BaseAddress": {
        "type": "LPVOID",
        "value": "0xfffff"
      },
      "ZeroBits": {
        "type": "ULONG_PTR",
        "value": "0"
      },
      "RegionSize": {
        "type": "PSIZE_T",
        "value": "4096"
      },
      "AllocationType": {
        "type": "ULONG",
        "value": "0"
      },
      "Protect": {
        "type": "ULONG",
        "value": "0"
      }
    }
  },
  "process-name": "program.exe",
  "process-id": 3982,
}
```

Process Instrumentation

Process Instrumentation

```
NTSYSAPI NTSTATUS NTAPI NtSetInformationProcess
(
  IN HANDLE                   ProcessHandle,
  IN PROCESS_INFORMATION_CLASS ProcessInformationClass,
  IN PVOID                    ProcessInformation,
  IN ULONG                    ProcessInformationLength
);
```

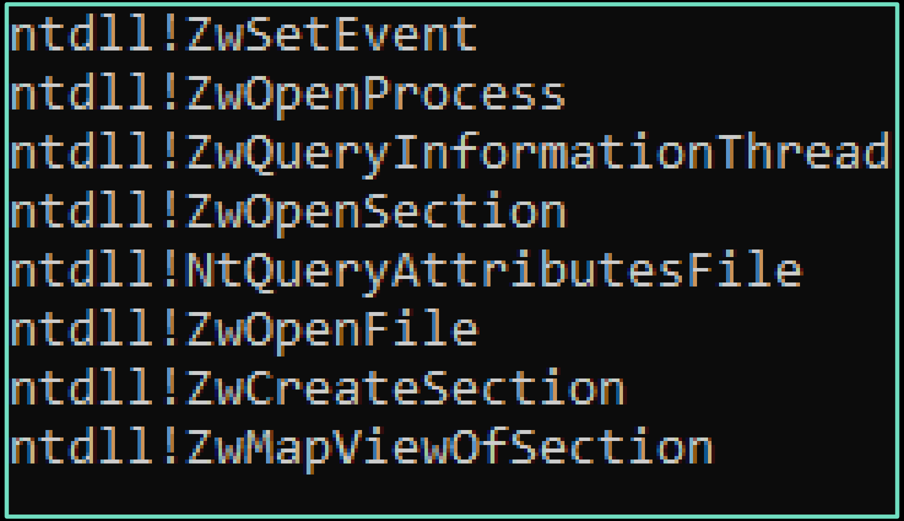
Detecting Manual Syscalls from User Mode

Process Instrumentation

```c
BOOL SetInstrumentationCallback() {
    PROCESS_INSTRUMENTATION_CALLBACK_INFORMATION Callback = { 0 };
    HANDLE hProcess = GetCurrentProcess();
    NTSTATUS Status = { 0 };

    Callback.Version = 0;
    Callback.Reserved = 0;
    Callback.Callback = InstrumentationCallbackThunk;

    Status = NtSetInformationProcess(hProcess, (PROCESSINFOCLASS)ProcessInstrumentationCallback,
&Callback, sizeof(Callback));

    if (NT_SUCCESS(Status))
    {
        return TRUE;
    }
    else
    {
        return FALSE;
    }
}
```

Process Instrumentation

```c
if (SymFromAddr(hProcess, Context->Rip, &Displacement, Symbol))
{
    if (SymGetModuleInfo64(hProcess, Symbol->Address, &ModuleInfo))
    {
        printf("%s!%s\n", ModuleInfo.ModuleName, Symbol->Name);
    }
}
```



```
ntdll!ZwSetEvent
ntdll!ZwOpenProcess
ntdll!ZwQueryInformationThread
ntdll!ZwOpenSection
ntdll!NtQueryAttributesFile
ntdll!ZwOpenFile
ntdll!ZwCreateSection
ntdll!ZwMapViewOfSection
```

Process Instrumentation

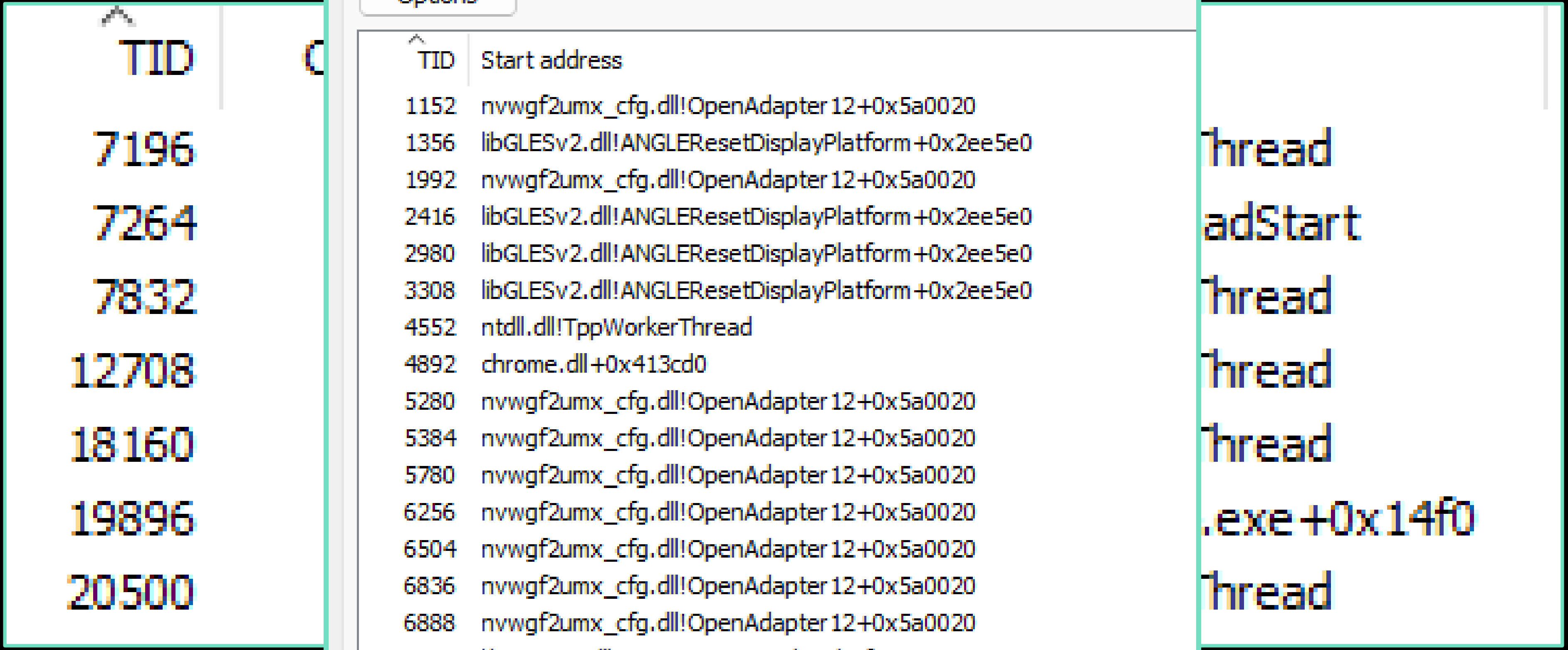
```
if (SymFromAddr(hProcess, Context->Rip, &Displacement, Symbol))
{
    if (SymGetModuleInfo64(hProcess, Symbol->Address, &ModuleInfo))
    {
        if (strcmp(ModuleInfo.ModuleName, "ntdll") != 0 || strlen(ModuleInfo.ModuleName) == 0)
        {
            print("Detected suspicious call: %s!%s\n", ModuleInfo.ModuleName, Symbol->Name);
        }
    }
}
```

Process Instrumentation

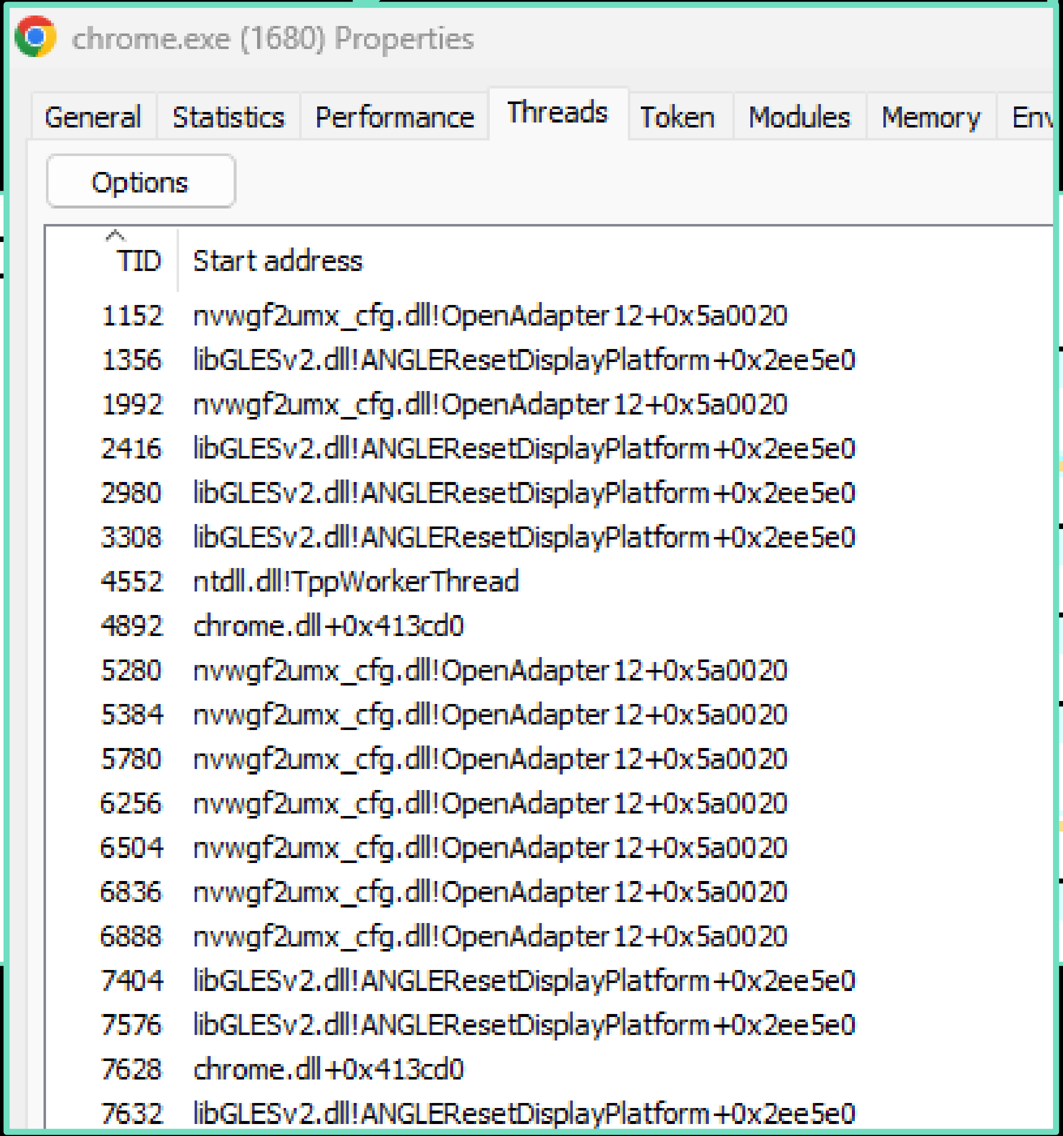
```
{
    "event": "process-instrumentation",
    "data": {
        "process-id": 5493,
        "thread-id": 1209,
        "image-file-name": "program.exe",
        "symbol-function": "NtWriteVirtualMemory",
        "symbol-module": ""
    }
}
```

Callstack Analysis

Callstack Analysis – Example



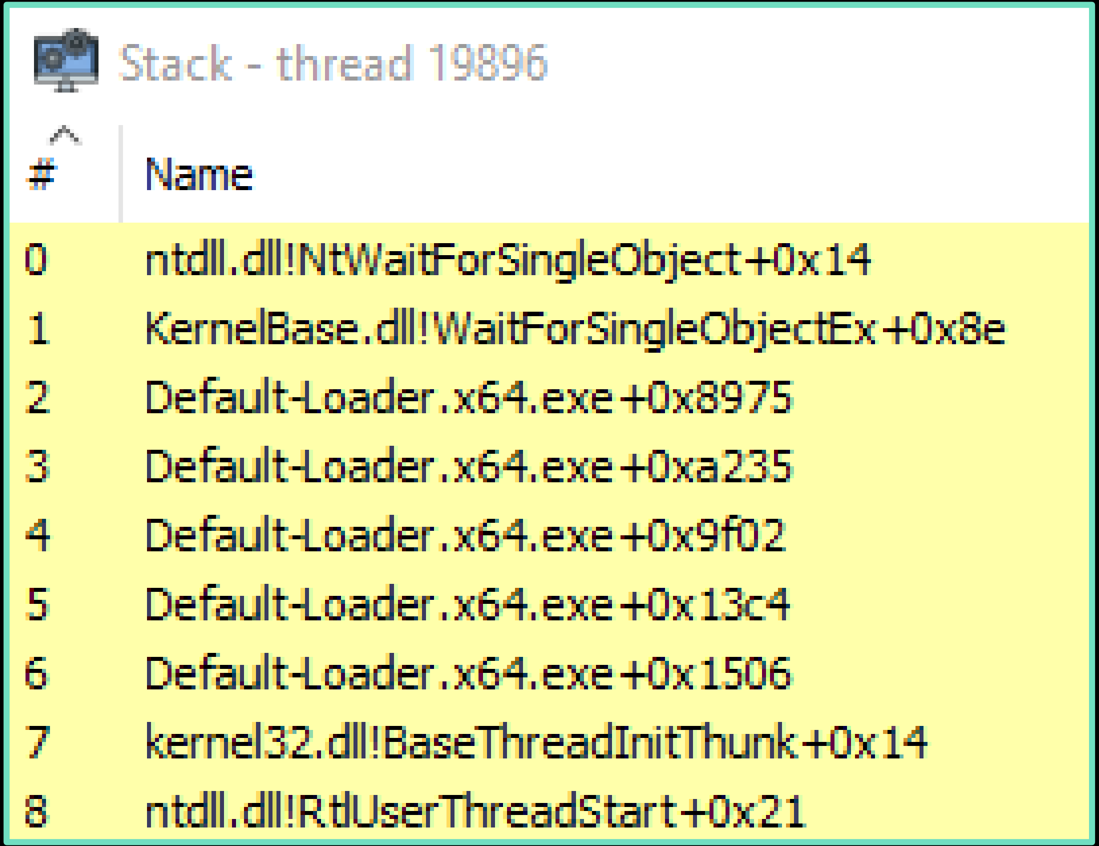
```
  TID    C
 7196          hread
 7264          adStart
 7832          hread
12708          hread
18160          hread
19896          .exe+0x14f0
20500          hread
```



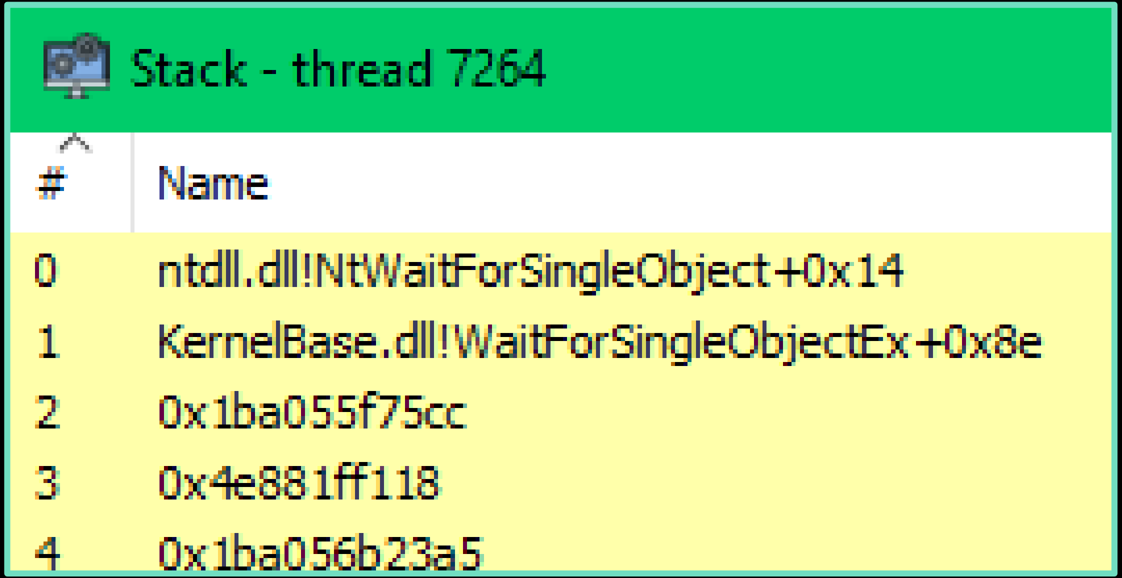
chrome.exe (1680) Properties

```
 TID   Start address
1152   nvwgf2umx_cfg.dll!OpenAdapter12+0x5a0020
1356   libGLESv2.dll!ANGLEResetDisplayPlatform+0x2ee5e0
1992   nvwgf2umx_cfg.dll!OpenAdapter12+0x5a0020
2416   libGLESv2.dll!ANGLEResetDisplayPlatform+0x2ee5e0
2980   libGLESv2.dll!ANGLEResetDisplayPlatform+0x2ee5e0
3308   libGLESv2.dll!ANGLEResetDisplayPlatform+0x2ee5e0
4552   ntdll.dll!TppWorkerThread
4892   chrome.dll+0x413cd0
5280   nvwgf2umx_cfg.dll!OpenAdapter12+0x5a0020
5384   nvwgf2umx_cfg.dll!OpenAdapter12+0x5a0020
5780   nvwgf2umx_cfg.dll!OpenAdapter12+0x5a0020
6256   nvwgf2umx_cfg.dll!OpenAdapter12+0x5a0020
6504   nvwgf2umx_cfg.dll!OpenAdapter12+0x5a0020
6836   nvwgf2umx_cfg.dll!OpenAdapter12+0x5a0020
6888   nvwgf2umx_cfg.dll!OpenAdapter12+0x5a0020
7404   libGLESv2.dll!ANGLEResetDisplayPlatform+0x2ee5e0
7576   libGLESv2.dll!ANGLEResetDisplayPlatform+0x2ee5e0
7628   chrome.dll+0x413cd0
7632   libGLESv2.dll!ANGLEResetDisplayPlatform+0x2ee5e0
```

Callstack Analysis – Example


Stack - thread 19896

#	Name
0	ntdll.dll!NtWaitForSingleObject+0x14
1	KernelBase.dll!WaitForSingleObjectEx+0x8e
2	Default-Loader.x64.exe+0x8975
3	Default-Loader.x64.exe+0xa235
4	Default-Loader.x64.exe+0x9f02
5	Default-Loader.x64.exe+0x13c4
6	Default-Loader.x64.exe+0x1506
7	kernel32.dll!BaseThreadInitThunk+0x14
8	ntdll.dll!RtlUserThreadStart+0x21


Stack - thread 7264

#	Name
0	ntdll.dll!NtWaitForSingleObject+0x14
1	KernelBase.dll!WaitForSingleObjectEx+0x8e
2	0x1ba055f75cc
3	0x4e881ff118
4	0x1ba056b23a5

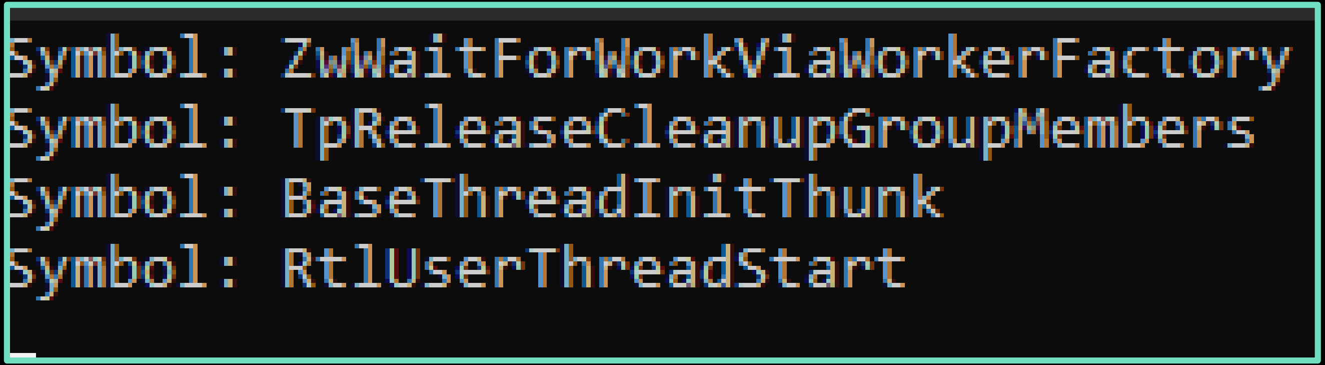
Callstack Analysis – Detect

```c
HANDLE hSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPTHREAD, 0);

THREADENTRY32 te = { 0 };

te.dwSize = sizeof(te);

if (Thread32First(hSnapshot, &te))
{
    do
    {
        if (te.dwSize >= FIELD_OFFSET(THREADENTRY32, th32OwnerProcessID) +
sizeof(te.th32OwnerProcessID))
        {
            if (te.th32OwnerProcessID == 0 && te.th32ThreadID == 0)
            {
                continue;
            }

            // do something

        }
        te.dwSize = sizeof(te);

    } while (Thread32Next(hSnapshot, &te));
}
```

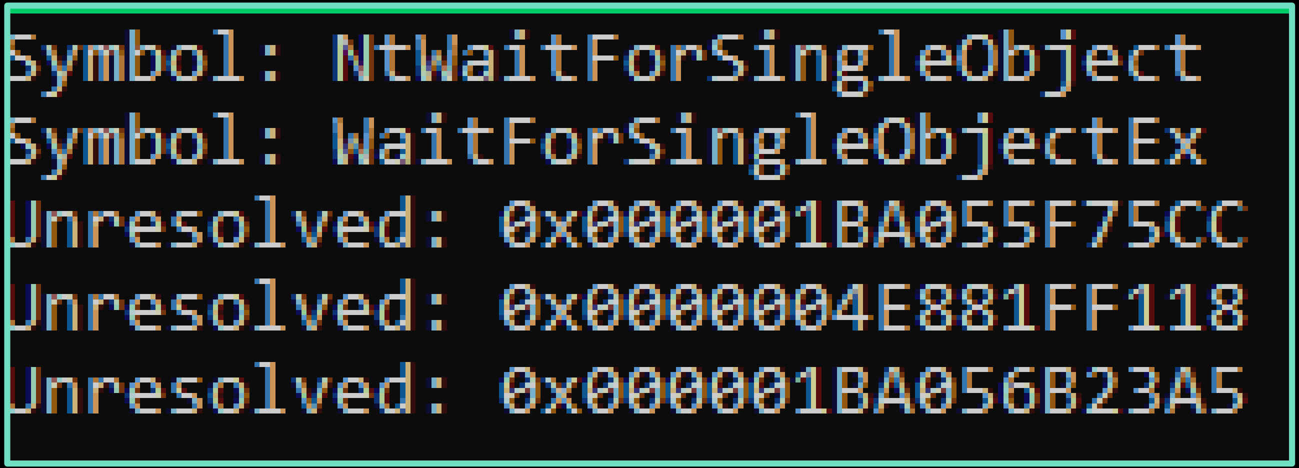
Callstack Analysis - Detect

```
do
{
  if (StackWalk64(IMAGE_FILE_MACHINE_AMD64, GetCurrentProcess(), hThread, &frame, &context, NULL,
SymFunctionTableAccess64, SymGetModuleBase64, NULL) == FALSE)
  {
      break;
  }

  DWORD64 dwOffset = 0;
  char Buffer[sizeof(SYMBOL_INFO) + MAX_SYM_NAME * sizeof(WCHAR)] = { 0 };
  PSYMBOL_INFO Symbol = (PSYMBOL_INFO)Buffer;
  Symbol->SizeOfStruct = sizeof(SYMBOL_INFO);
  Symbol->MaxNameLen = MAX_SYM_NAME;

  if (SymFromAddr(GetCurrentProcess(), frame.AddrPC.Offset, &dwOffset, Symbol))
  {
      printf("Symbol: %s\n", Symbol->Name);
  }
  else
  {
      printf("Unresolved: 0x%p\n", reinterpret_cast<LPVOID>(frame.AddrPC.Offset));
  }

} while (true);
```

Callstack Analysis - Detect

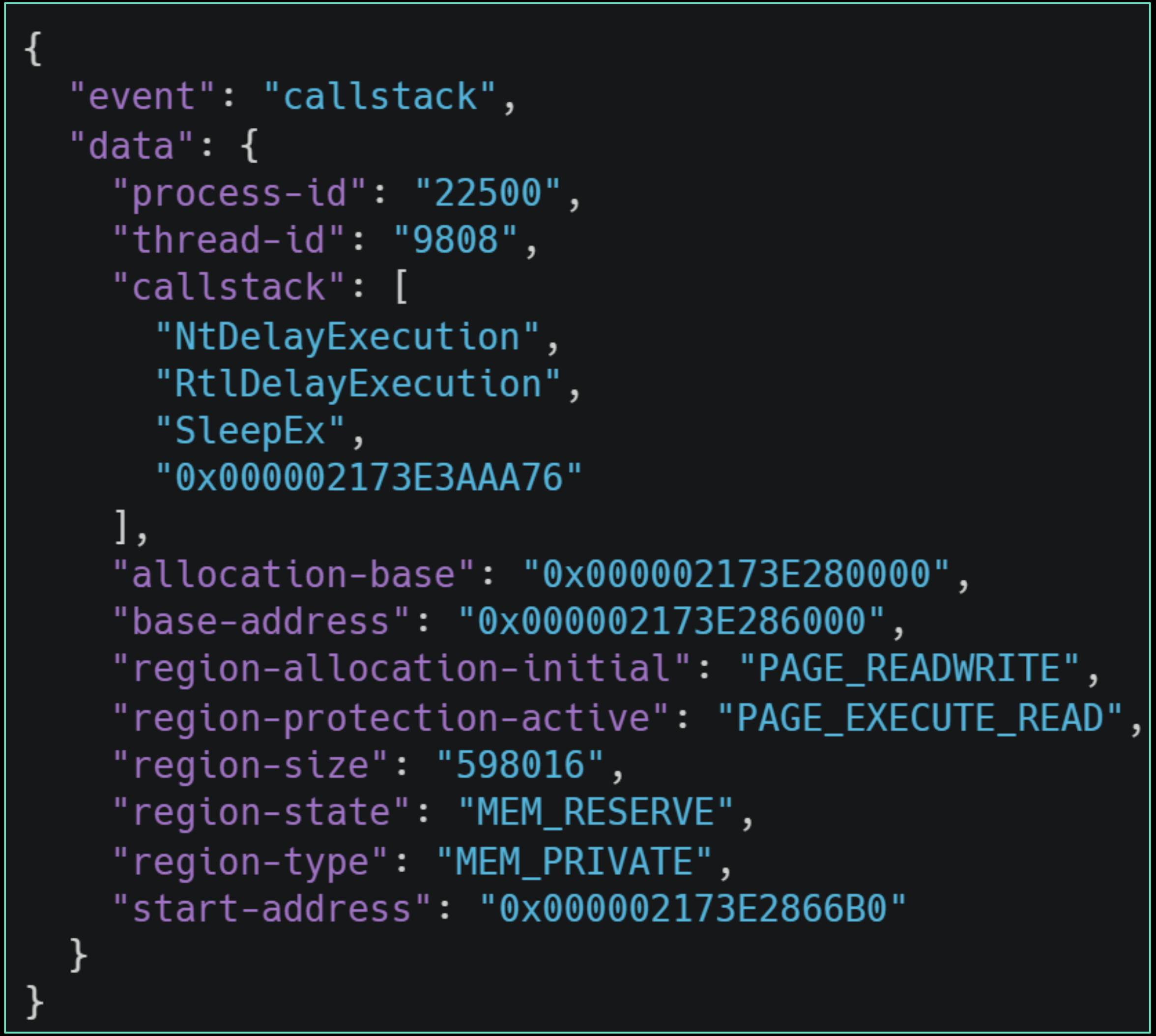


```
Symbol: ZwWaitForWorkViaWorkerFactory
Symbol: TpReleaseCleanupGroupMembers
Symbol: BaseThreadInitThunk
Symbol: RtlUserThreadStart
```



```
Symbol: NtWaitForSingleObject
Symbol: WaitForSingleObjectEx
Unresolved: 0x000001BA055F75CC
Unresolved: 0x0000004E881FF118
Unresolved: 0x000001BA056B23A5
```

Callstack Analysis – Detect



```
{
  "event": "callstack",
  "data": {
    "process-id": "22500",
    "thread-id": "9808",
    "callstack": [
      "NtDelayExecution",
      "RtlDelayExecution",
      "SleepEx",
      "0x000002173E3AAA76"
    ],
    "allocation-base": "0x000002173E280000",
    "base-address": "0x000002173E286000",
    "region-allocation-initial": "PAGE_READWRITE",
    "region-protection-active": "PAGE_EXECUTE_READ",
    "region-size": "598016",
    "region-state": "MEM_RESERVE",
    "region-type": "MEM_PRIVATE",
    "start-address": "0x000002173E2866B0"
  }
}
```

Callstack Analysis – Detect

```
{
  "process-id": "22500",
  "thread-id": "9808",
  "image-file-name": "OutlookHelper.exe",
  "image-file-location": "c:\\windows\\temp",
  "callstack": [
    "NtDelayExecution",
    "RtlDelayExecution",
    "SleepEx",
    "0x000002173E3AAA76"
  ],
  "allocation-base": "0x000002173E280000",
  "base-address": "0x000002173E286000",
  "region-allocation-initial": "PAGE_READWRITE",
  "region-protection-active": "PAGE_EXECUTE_READ",
  "region-size": "598016",
  "region-state": "MEM_RESERVE",
  "region-type": "MEM_PRIVATE",
  "start-address": "0x000002173E2866B0",
  "symbol-info": {
    "symbol-function": "NtCreateThreadEx",
    "symbol-module": ""
  }
}
```

Kernel Callbacks

The Callbacks

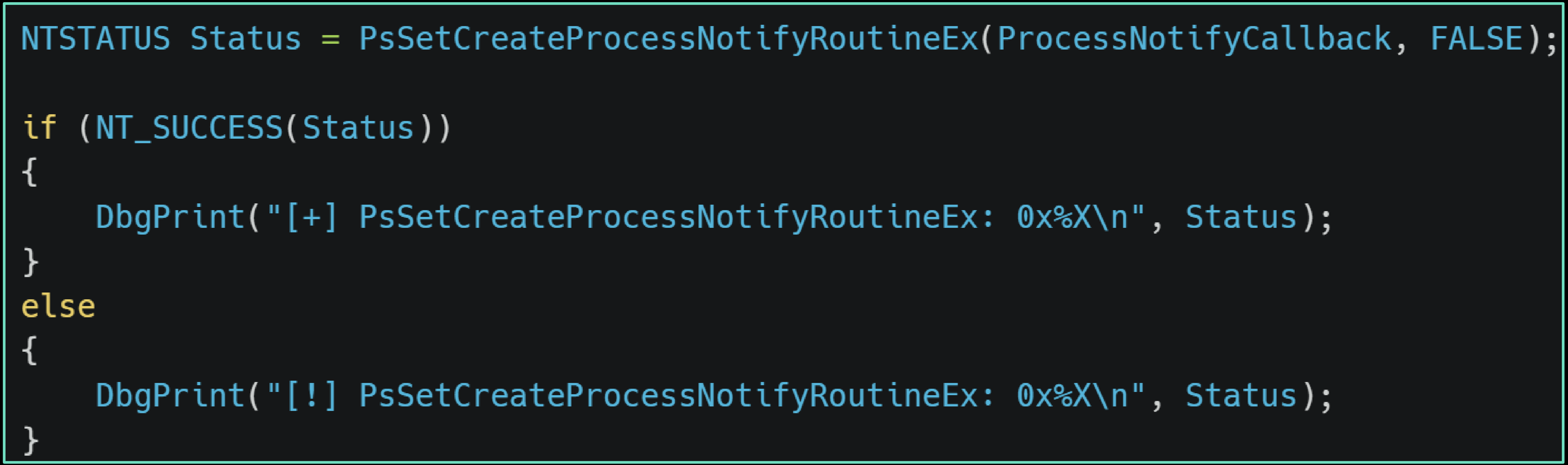
PsSetCreateProcessNotifyRoutineEx

PsSetCreateThreadNotifyRoutine

PsSetCreateProcessNotifyRoutineEx

```
NTSTATUS PsSetCreateProcessNotifyRoutineEx(
  [in] PCREATE_PROCESS_NOTIFY_ROUTINE_EX NotifyRoutine,
  [in] BOOLEAN                           Remove
);
```

PsSetCreateProcessNotifyRoutineEx

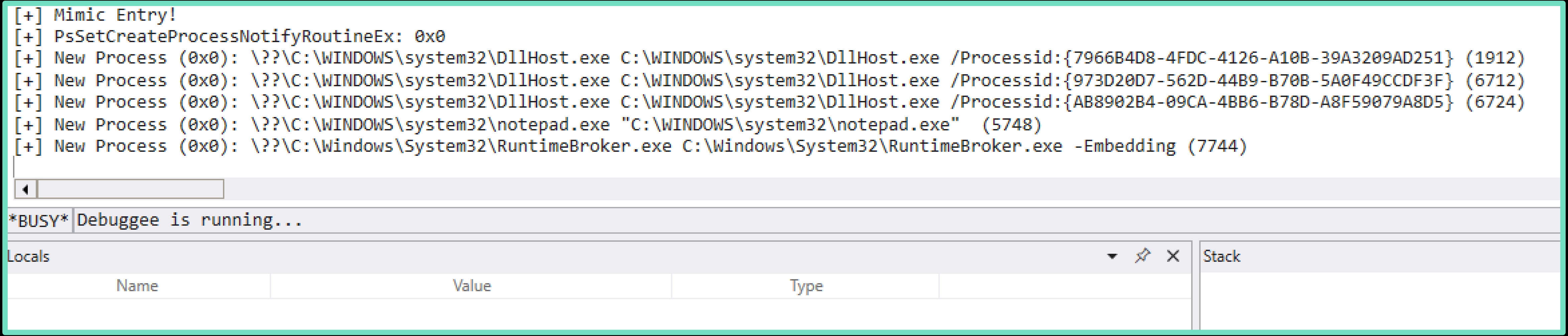


```c
NTSTATUS Status = PsSetCreateProcessNotifyRoutineEx(ProcessNotifyCallback, FALSE);

if (NT_SUCCESS(Status))
{
    DbgPrint("[+] PsSetCreateProcessNotifyRoutineEx: 0x%X\n", Status);
}
else
{
    DbgPrint("[!] PsSetCreateProcessNotifyRoutineEx: 0x%X\n", Status);
}
```

PsSetCreateProcessNotifyRoutineEx

```c
VOID ProcessNotifyCallback(PEPROCESS Process, HANDLE ProcessId, PPS_CREATE_NOTIFY_INFO CreateInfo)
{
    UNREFERENCED_PARAMETER(Process);

    if (CreateInfo)
    {
        if (CreateInfo->FileOpenNameAvailable)
        {
            DbgPrint("[*] New Process (0x%X): %wZ %wZ (%d)\n",
                CreateInfo->CreationStatus,
                CreateInfo->ImageFileName,
                CreateInfo->CommandLine,
                ProcessId
            );
        }
    }
}
```

PsSetCreateProcessNotifyRoutineEx



```
[+] Mimic Entry!
[+] PsSetCreateProcessNotifyRoutineEx: 0x0
[+] New Process (0x0): \??\C:\WINDOWS\system32\DllHost.exe C:\WINDOWS\system32\DllHost.exe /Processid:{7966B4D8-4FDC-4126-A10B-39A3209AD251} (1912)
[+] New Process (0x0): \??\C:\WINDOWS\system32\DllHost.exe C:\WINDOWS\system32\DllHost.exe /Processid:{973D20D7-562D-44B9-B70B-5A0F49CCDF3F} (6712)
[+] New Process (0x0): \??\C:\WINDOWS\system32\DllHost.exe C:\WINDOWS\system32\DllHost.exe /Processid:{AB8902B4-09CA-4BB6-B78D-A8F59079A8D5} (6724)
[+] New Process (0x0): \??\C:\WINDOWS\system32\notepad.exe "C:\WINDOWS\system32\notepad.exe"  (5748)
[+] New Process (0x0): \??\C:\Windows\System32\RuntimeBroker.exe C:\Windows\System32\RuntimeBroker.exe -Embedding (7744)
```

BUSY Debuggee is running...

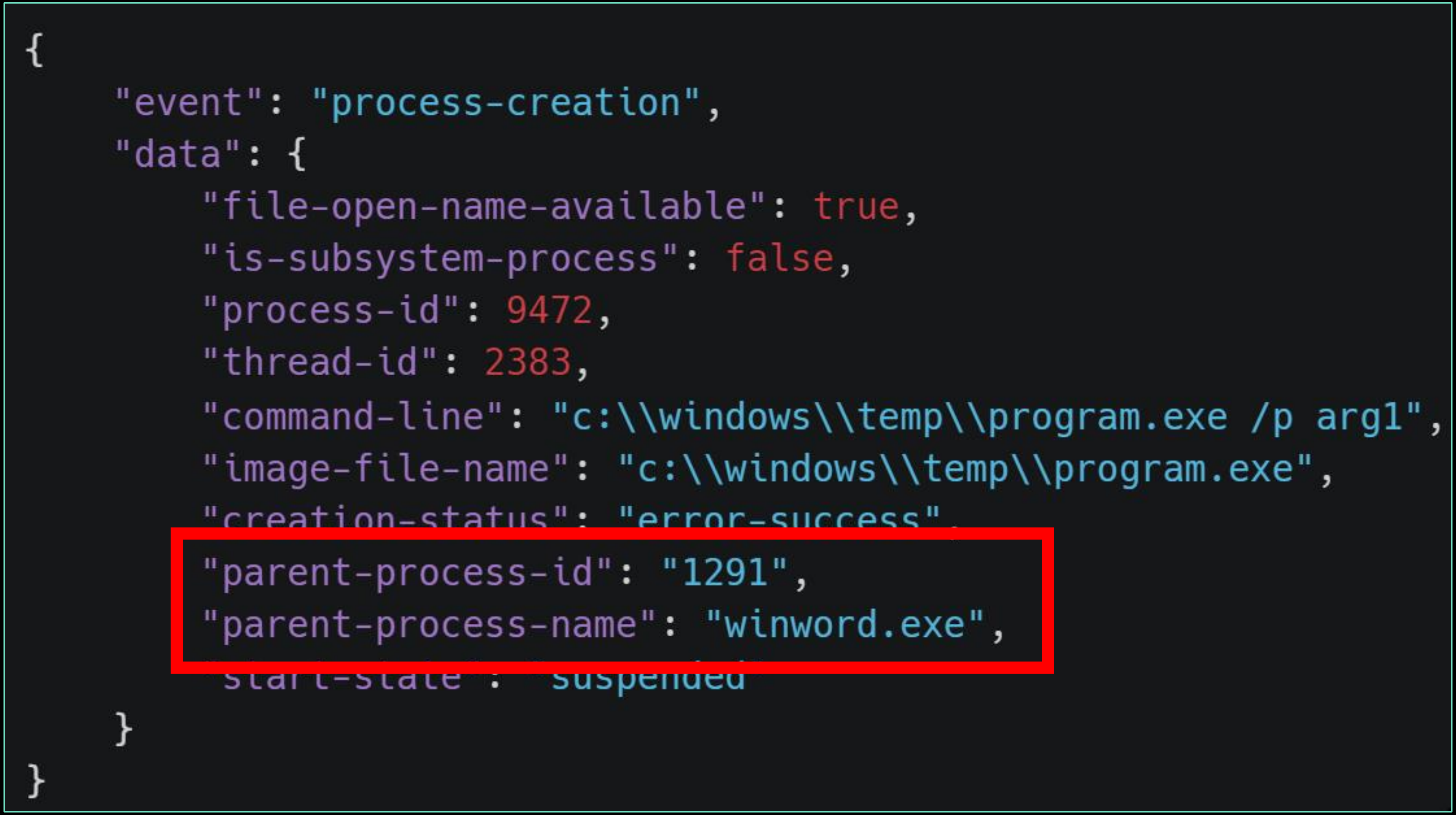
PsSetCreateProcessNotifyRoutineEx

```c
typedef struct _PS_CREATE_NOTIFY_INFO {
  SIZE_T              Size;
  union {
    ULONG Flags;
    struct {
      ULONG FileOpenNameAvailable : 1;
      ULONG IsSubsystemProcess : 1;
      ULONG Reserved : 30;
    };
  };
  HANDLE              ParentProcessId;
  CLIENT_ID           CreatingThreadId;
  struct _FILE_OBJECT *FileObject;
  PCUNICODE_STRING    ImageFileName;
  PCUNICODE_STRING    CommandLine;
  NTSTATUS            CreationStatus;
} PS_CREATE_NOTIFY_INFO,
```

PsSetCreateProcessNotifyRoutineEx

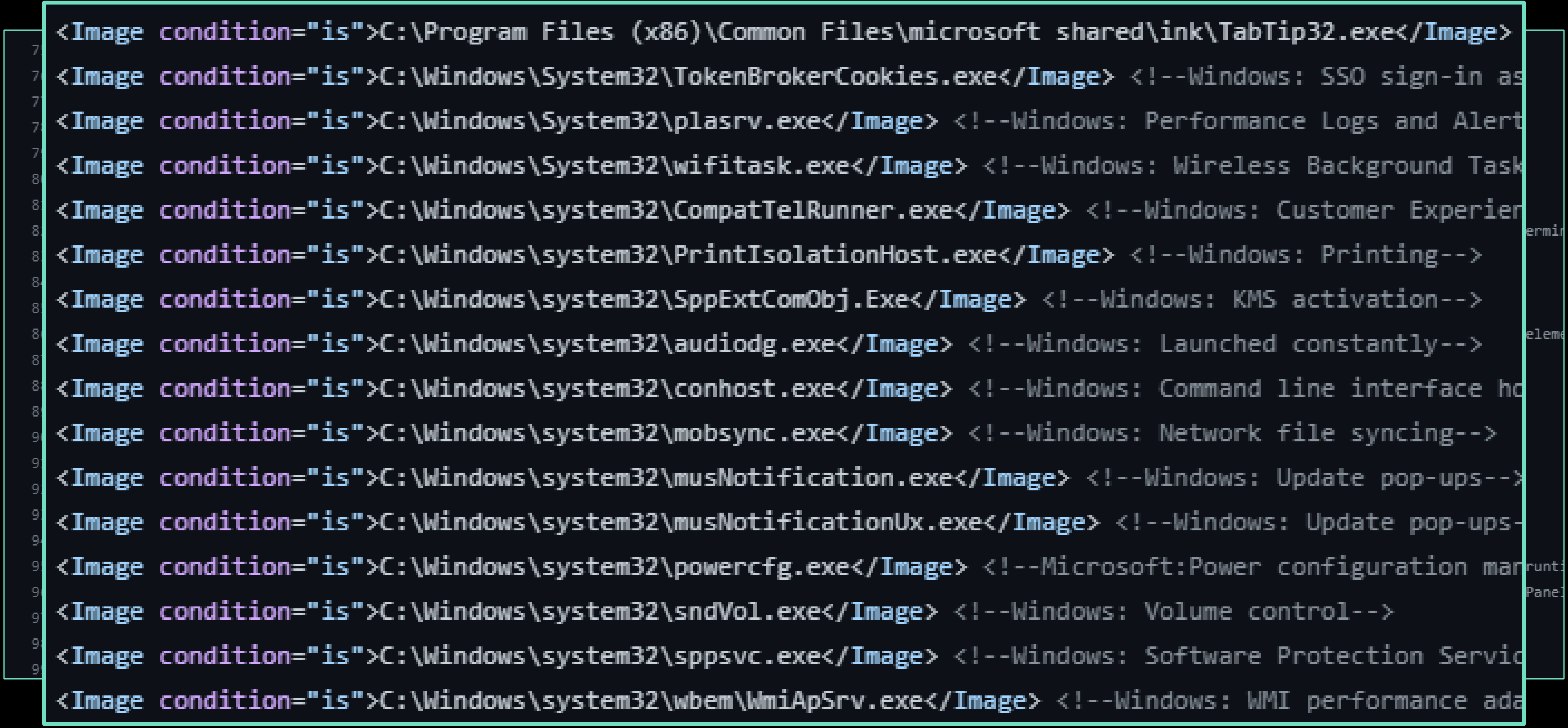
```
{
    "event": "process-creation",
    "data": {
        "file-open-name-available": true,
        "is-subsystem-process": false,
        "process-id": 9472,
        "thread-id": 2383,
        "command-line": "c:\\windows\\temp\\program.exe /p arg1",
        "image-file-name": "c:\\windows\\temp\\program.exe",
        "creation-status": "error-success",
    }
}
```

PsSetCreateProcessNotifyRoutineEx



```
{
    "event": "process-creation",
    "data": {
        "file-open-name-available": true,
        "is-subsystem-process": false,
        "process-id": 9472,
        "thread-id": 2383,
        "command-line": "c:\\windows\\temp\\program.exe /p arg1",
        "image-file-name": "c:\\windows\\temp\\program.exe",
        "creation-status": "error-success",
        "parent-process-id": "1291",
        "parent-process-name": "winword.exe",
        "start-state": "suspended"
    }
}
```

PsSetCreateProcessNotifyRoutineEx



```
<Image condition="is">C:\Program Files (x86)\Common Files\microsoft shared\ink\TabTip32.exe</Image>
<Image condition="is">C:\Windows\System32\TokenBrokerCookies.exe</Image> <!--Windows: SSO sign-in as
<Image condition="is">C:\Windows\System32\plasrv.exe</Image> <!--Windows: Performance Logs and Alert
<Image condition="is">C:\Windows\System32\wifitask.exe</Image> <!--Windows: Wireless Background Task
<Image condition="is">C:\Windows\system32\CompatTelRunner.exe</Image> <!--Windows: Customer Experien
<Image condition="is">C:\Windows\system32\PrintIsolationHost.exe</Image> <!--Windows: Printing-->
<Image condition="is">C:\Windows\system32\SppExtComObj.Exe</Image> <!--Windows: KMS activation-->
<Image condition="is">C:\Windows\system32\audiodg.exe</Image> <!--Windows: Launched constantly-->
<Image condition="is">C:\Windows\system32\conhost.exe</Image> <!--Windows: Command line interface ho
<Image condition="is">C:\Windows\system32\mobsync.exe</Image> <!--Windows: Network file syncing-->
<Image condition="is">C:\Windows\system32\musNotification.exe</Image> <!--Windows: Update pop-ups-->
<Image condition="is">C:\Windows\system32\musNotificationUx.exe</Image> <!--Windows: Update pop-ups-
<Image condition="is">C:\Windows\system32\powercfg.exe</Image> <!--Microsoft:Power configuration man
<Image condition="is">C:\Windows\system32\sndVol.exe</Image> <!--Windows: Volume control-->
<Image condition="is">C:\Windows\system32\sppsvc.exe</Image> <!--Windows: Software Protection Servic
<Image condition="is">C:\Windows\system32\wbem\WmiApSrv.exe</Image> <!--Windows: WMI performance ada
```

PsSetCreateProcessNotifyRoutineEx

```
query = '''
sequence by host.id with maxspan=1m
  [process where host.os.type == "windows" and event.code == "1" and
   /* sysmon process creation */
   process.parent.name : ("winword.exe", "excel.exe", "outlook.exe", "powerpnt.exe", "eqnedt32.exe", "fltldr.exe",
                          "mspub.exe", "msaccess.exe","cscript.exe", "wscript.exe", "rundll32.exe", "regsvr32.exe",
                          "mshta.exe", "wmic.exe", "cmstp.exe", "msxsl.exe") and

   /* noisy FP patterns */
   not (process.parent.name : "EXCEL.EXE" and process.executable : "?:\\Program Files\\Microsoft Office\\root\\Office*\\ADDINS\\*.exe") and
   not (process.executable : "?:\\Windows\\splwow64.exe" and process.args in ("8192", "12288") and process.parent.name : ("winword.exe", "excel.exe", "outlook.exe", "powerpnt.exe")) and
   not (process.parent.name : "rundll32.exe" and process.parent.args : ("?:\\WINDOWS\\Installer\\MSI*.tmp,zzzzInvokeManagedCustomActionOutOfProc", "--no-sandbox")) and
   not (process.executable :
            ("?:\\Program Files (x86)\\Microsoft\\EdgeWebView\\Application\\*\\msedgewebview2.exe",
             "?:\\Program Files\\Adobe\\Acrobat DC\\Acrobat\\Acrobat.exe",
             "?:\\Windows\\SysWOW64\\DWWIN.EXE") and
        process.parent.name : ("winword.exe", "excel.exe", "outlook.exe", "powerpnt.exe")) and
   not (process.parent.name : "regsvr32.exe" and process.parent.args : ("?:\\Program Files\\*", "?:\\Program Files (x86)\\*"))
   ] by process.parent.entity_id, process.entity_id
  [process where host.os.type == "windows" and event.code == "10" and
   /* Sysmon process access event from unknown module */
   winlog.event_data.CallTrace : "*UNKNOWN*"] by process.entity_id, winlog.event_data.TargetProcessGUID
'''
```

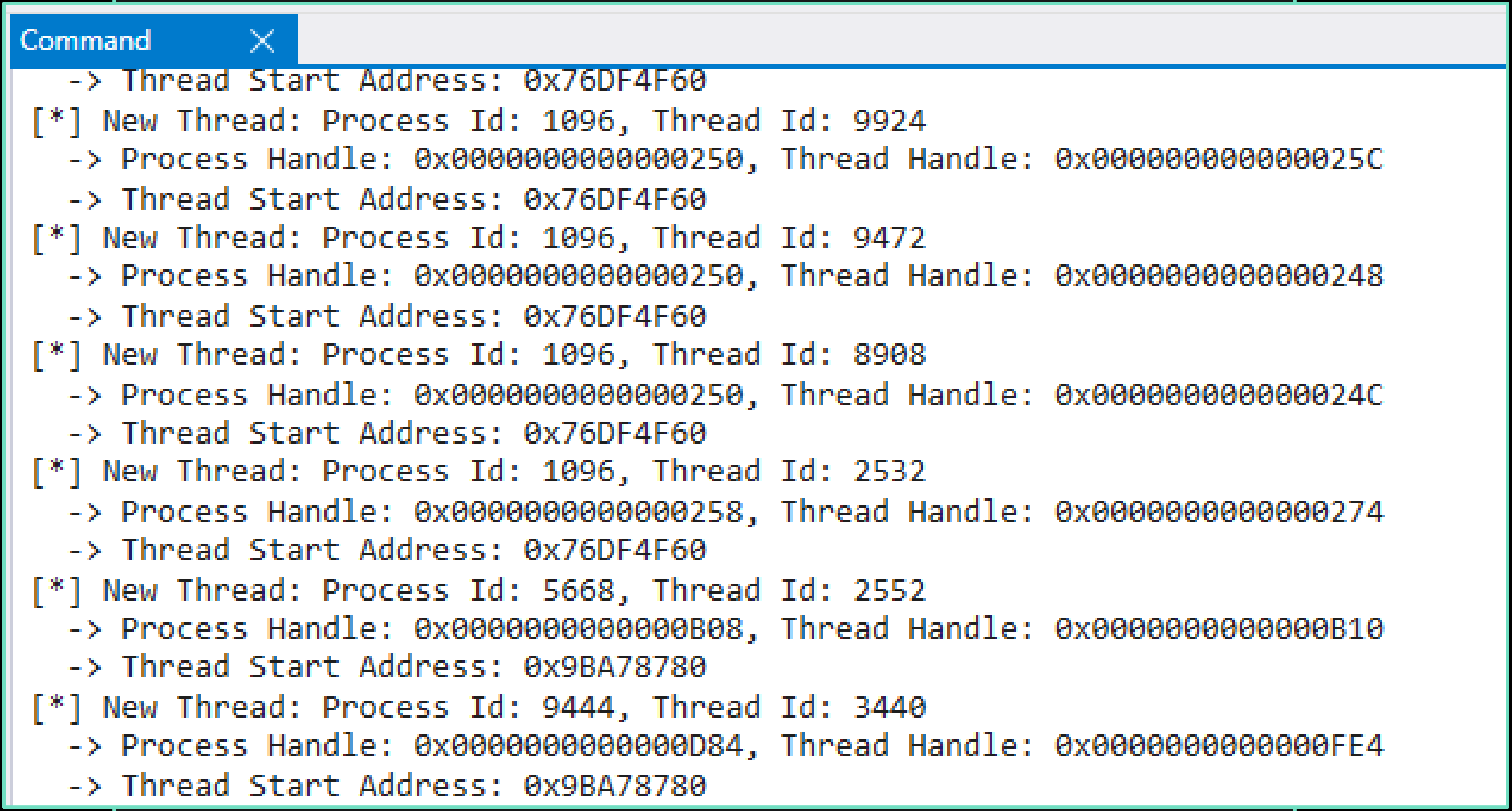
PsSetCreateThreadNotifyRoutine

```c
NTSTATUS Status = PsSetCreateThreadNotifyRoutine(ThreadNotifyCallback);

if (NT_SUCCESS(Status))
{
    DbgPrint("[+] PsSetCreateThreadNotifyRoutineEx: 0x%X\n", Status);
}
else
{
    DbgPrint("[!] PsSetCreateThreadNotifyRoutineEx: 0x%X\n", Status);
}
```

PsSetCreateThreadNotifyRoutine

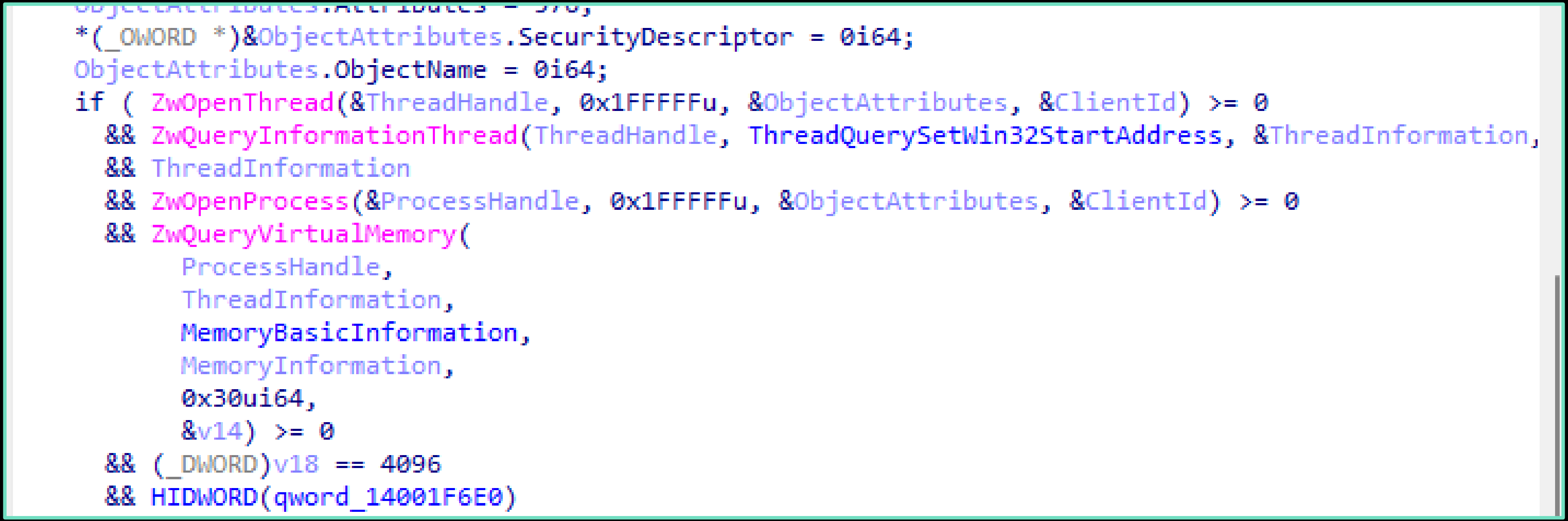
```
NTSTATUS Status = PsSetCreateThreadNotifyRoutine(ThreadNotifyCallback);

if (NT_SUCCESS(Status))
{
    DbgPrint("[+] PsSetCreateThreadNotifyRoutineEx: 0x%X\n", Status);
}
else
{
    DbgPrint("[!] PsSetCreateThreadNotifyRoutineEx: 0x%X\n", Status);
}
```

PsSetCreateThreadNotifyRoutine



```
   -> Thread Start Address: 0x76DF4F60
[*] New Thread: Process Id: 1096, Thread Id: 9924
   -> Process Handle: 0x0000000000000250, Thread Handle: 0x000000000000025C
   -> Thread Start Address: 0x76DF4F60
[*] New Thread: Process Id: 1096, Thread Id: 9472
   -> Process Handle: 0x0000000000000250, Thread Handle: 0x0000000000000248
   -> Thread Start Address: 0x76DF4F60
[*] New Thread: Process Id: 1096, Thread Id: 8908
   -> Process Handle: 0x0000000000000250, Thread Handle: 0x000000000000024C
   -> Thread Start Address: 0x76DF4F60
[*] New Thread: Process Id: 1096, Thread Id: 2532
   -> Process Handle: 0x0000000000000258, Thread Handle: 0x0000000000000274
   -> Thread Start Address: 0x76DF4F60
[*] New Thread: Process Id: 5668, Thread Id: 2552
   -> Process Handle: 0x0000000000000B08, Thread Handle: 0x0000000000000B10
   -> Thread Start Address: 0x9BA78780
[*] New Thread: Process Id: 9444, Thread Id: 3440
   -> Process Handle: 0x0000000000000D84, Thread Handle: 0x0000000000000FE4
   -> Thread Start Address: 0x9BA78780
```

PsSetCreateThreadNotifyRoutine



```
*(_OWORD *)&ObjectAttributes.SecurityDescriptor = 0i64;
ObjectAttributes.ObjectName = 0i64;
if ( ZwOpenThread(&ThreadHandle, 0x1FFFFFu, &ObjectAttributes, &ClientId) >= 0
  && ZwQueryInformationThread(ThreadHandle, ThreadQuerySetWin32StartAddress, &ThreadInformation,
  && ThreadInformation
  && ZwOpenProcess(&ProcessHandle, 0x1FFFFFu, &ObjectAttributes, &ClientId) >= 0
  && ZwQueryVirtualMemory(
       ProcessHandle,
       ThreadInformation,
       MemoryBasicInformation,
       MemoryInformation,
       0x30ui64,
       &v14) >= 0
  && (_DWORD)v18 == 4096
  && HIDWORD(qword_14001F6E0)
```

PsSetCreateThreadNotifyRoutine

```json
{
    "event": "thread-creation",
    "data": {
        "thread-id": 2932,
        "process-id": 9472,
        "disk-backed": false
    }
}
```

All4One

All4One

```json
{
  "events": [
    "process-instrumentation",
    "process-creation",
    "callstack"
  ],
  "process-id": "22500",
  "thread-id": "9808",
  "image-file-name": "OutlookHelper.exe",
  "image-file-location": "c:\\windows\\temp",
  "command-line": "/q 1a56f852-38dd-49f3-b544-ebb8bf0597fd",
  "parent-process-id": 493,
  "parent-process-name": "explorer.exe",
  "process-start-state": "suspended",
  "callstack": [
    "NtDelayExecution",
    "RtlDelayExecution",
    "SleepEx",
    "0x000002173E3AAA76"
  ],
  "allocation-base": "0x000002173E280000",
  "base-address": "0x000002173E286000",
  "region-allocation-initial": "PAGE_READWRITE",
  "region-protection-active": "PAGE_EXECUTE_READ",
  "region-size": "598016",
  "region-state": "MEM_RESERVE",
  "region-type": "MEM_PRIVATE",
  "start-address": "0x000002173E2866B0",
  "symbol-info": {
    "symbol-function": "NtCreateThreadEx",
    "symbol-module": ""
  }
}
```

Final note

Understanding telemetry life cycle > new hotness

As AI/ML usage increases, simple bypasses like SysCalls from a few years ago…

Go/Rust/Nim/stuff work right now because of low data points

Thanks!

// SteelCon 2023