H-CON
HACKPLAYERS
CONFERENCE

V
FIVE

24 y 25 FEBRERO
MADRID 2023

WHAT THE (WMF)uck?!

Nacho Gómez AKA nagomez

About Me

- Offensive Security Researcher
- (Ex) Red Team
- Windows Internals, EDR/AV evasión
- C/C++/**C# <=**
- Informática + Matemáticas
- @_nag0mez

https://pwnedcoffee.com/

TOC

What the (WNF)uck?!

Nacho Gómez aka *nag0mez*

24 y 25 FEBRERO

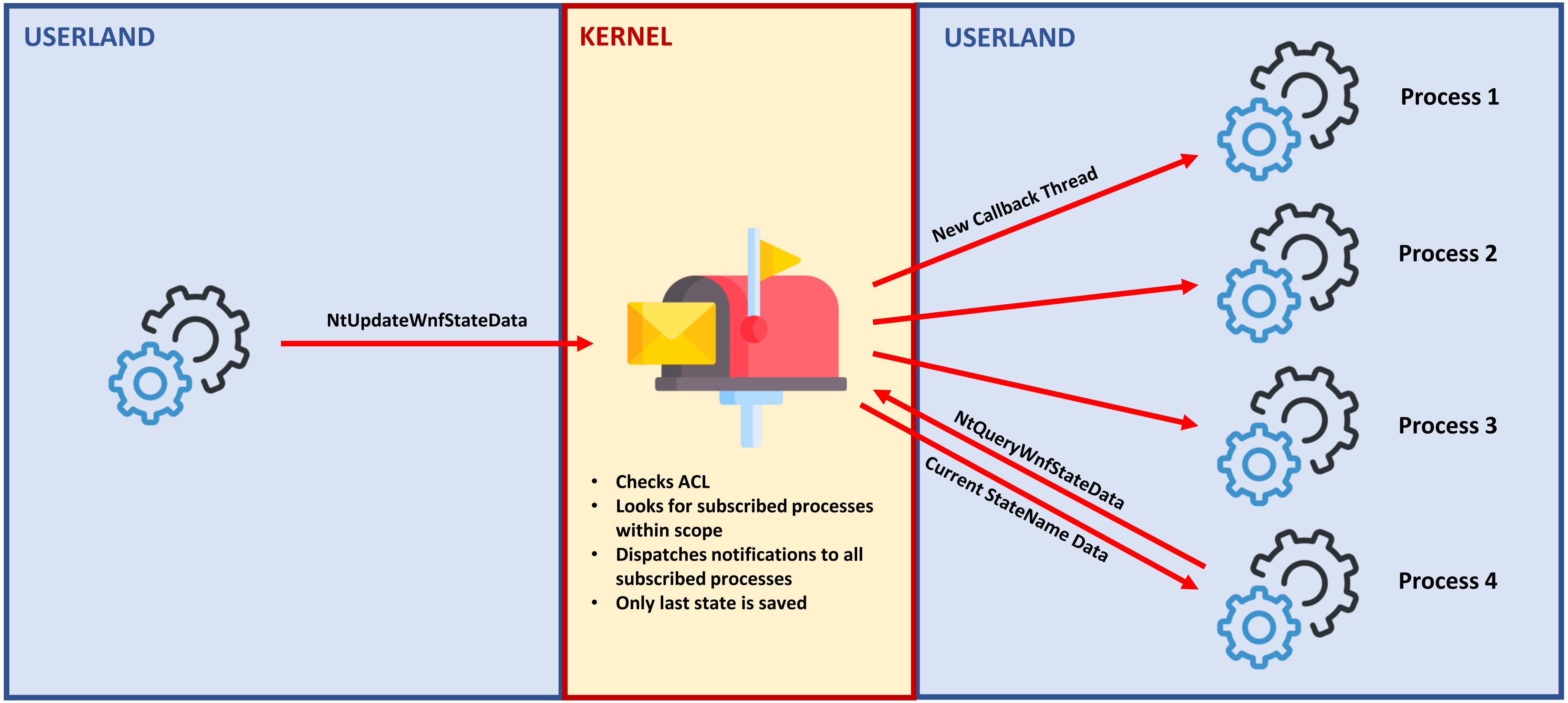
MADRID 2023

INTRODUCCIÓN

~~WTF~~ WNF?

- **Windows Notification Facility**
- Componente kernel (ntoskrnl.exe)
- Sistema de notificaciones entre user y kernel basado en subscripciones introducido en Windows 8.
- Cualquier proceso puede suscribirse a un evento, incluso si este no existe todavía.
- Notificaciones basadas en un **StateName**
- Payloads de hasta 4KB, DACL para controlar RW
- "Windows Notification Facility: Peeling the Onion of the Most Undocumented Kernel Attack Surface Yet", Alex Ionescu + Gabrielle Viala, Black Hat 2018.

INTRODUCCIÓN



- Checks ACL
- Looks for subscribed processes within scope
- Dispatches notifications to all subscribed processes
- Only last state is saved

What the (WNF)uck?!

Nacho Gómez aka *nag0mez*

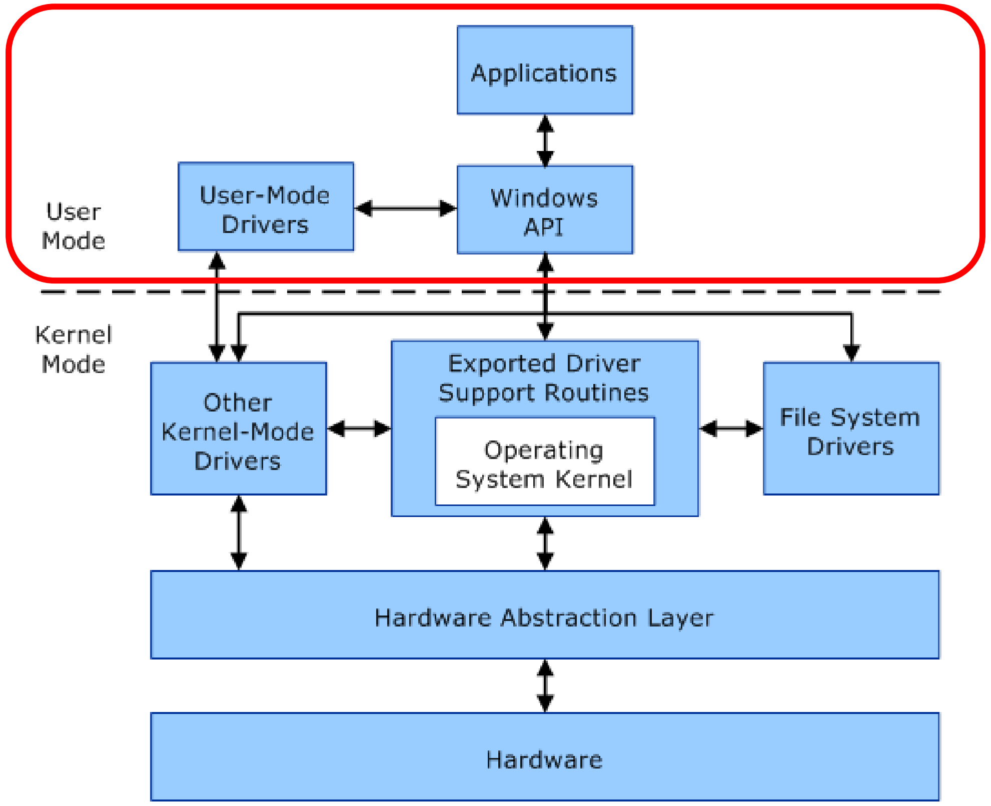
24 y 25 FEBRERO

MADRID 2023

USER/KERNEL MODES

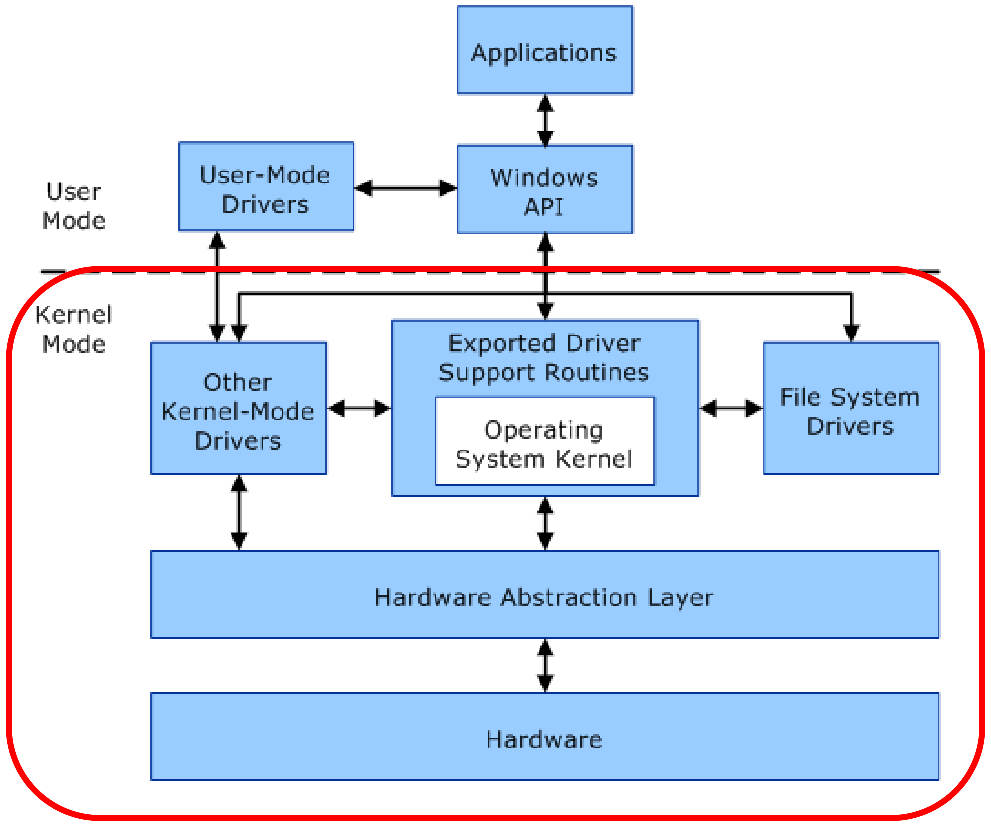
User Mode

- Windows asigna un **proceso** a cada aplicación User Mode.
- Cada proceso tiene su propio **espacio de direcciones virtuales** y una tabla privada de **handles** (identificadores de objetos en kernel)
- Aislamiento entre procesos

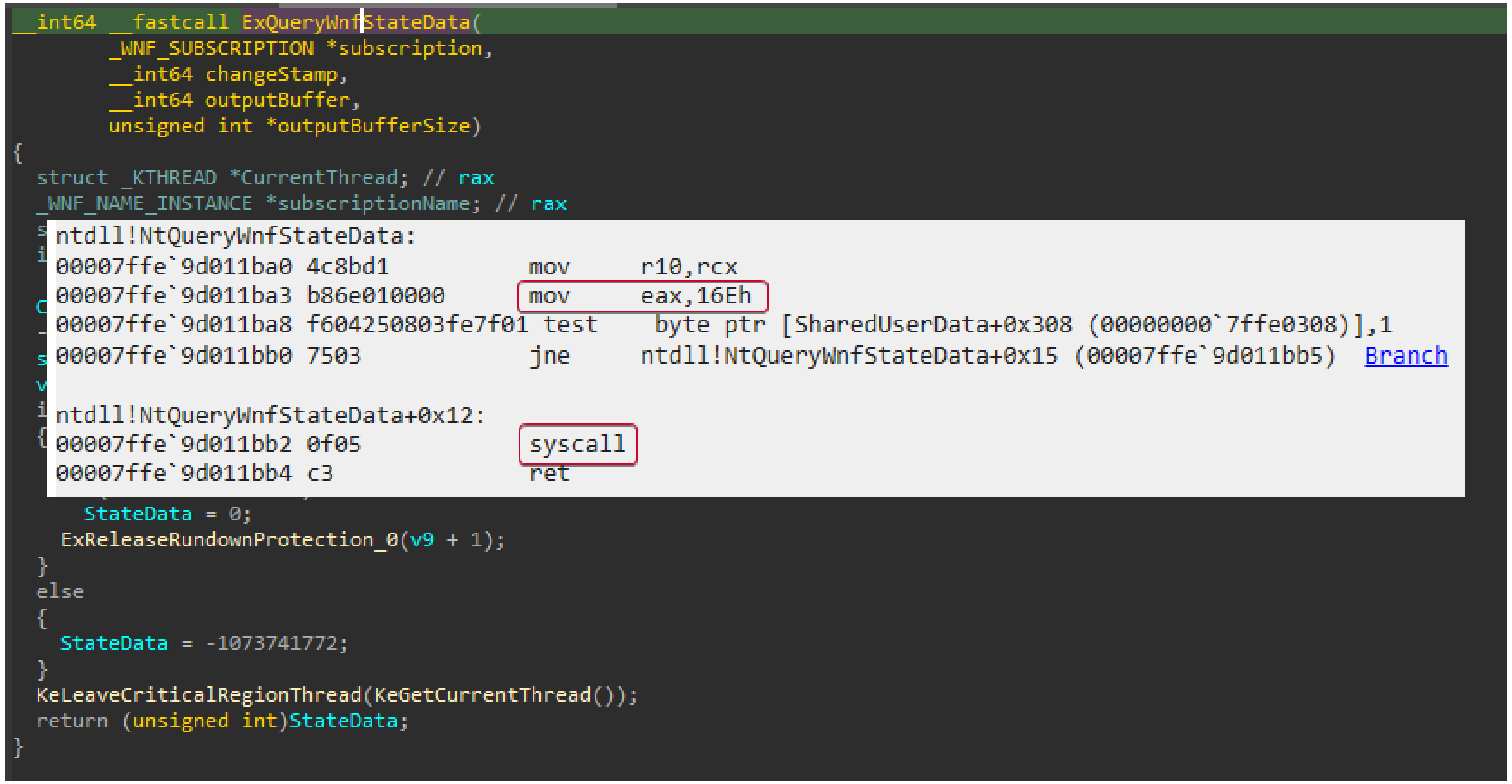


Kernel Mode

- Todo el código ejecutado en Kernel Mode tiene un único espacio de direcciones virtuales
- Objetos representados por **handles**
- **NTDLL** sirve como punto de "salto" entre User y Kernel.



USER/KERNEL MODES



```
__int64 __fastcall ExQueryWnfStateData(
        _WNF_SUBSCRIPTION *subscription,
        __int64 changeStamp,
        __int64 outputBuffer,
        unsigned int *outputBufferSize)
{
  struct _KTHREAD *CurrentThread; // rax
  _WNF_NAME_INSTANCE *subscriptionName; // rax
  s
  i

  C
  _
  s
  v
  i
  {
      StateData = 0;
    ExReleaseRundownProtection_0(v9 + 1);
  }
  else
  {
    StateData = -1073741772;
  }
  KeLeaveCriticalRegionThread(KeGetCurrentThread());
  return (unsigned int)StateData;
}
```

```
ntdll!NtQueryWnfStateData:
00007ffe`9d011ba0 4c8bd1          mov     r10,rcx
00007ffe`9d011ba3 b86e010000      mov     eax,16Eh
00007ffe`9d011ba8 f604250803fe7f01 test    byte ptr [SharedUserData+0x308 (00000000`7ffe0308)],1
00007ffe`9d011bb0 7503            jne     ntdll!NtQueryWnfStateData+0x15 (00007ffe`9d011bb5)  Branch

ntdll!NtQueryWnfStateData+0x12:
00007ffe`9d011bb2 0f05            syscall
00007ffe`9d011bb4 c3              ret
```

WNF DATA

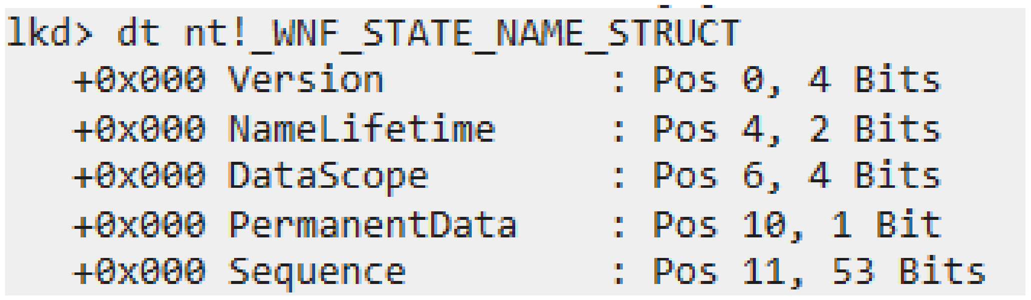
- Kernel Object (almacenado en Kernel Pool) identificado por un StateName
- Se copia la información en la memoria de cada proceso (Query/Publicación)
- ACL/Security Descriptors
- Structs más adelante

WNF STATE NAMES

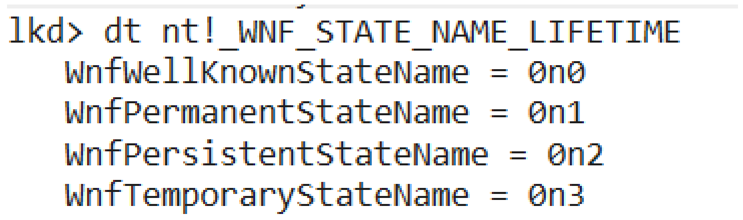
WNF STATE NAMES

State Names

- 64-bit integer que “esconde” un struct
- StateName XOR 0x41C64E6DA3BC0074
- **NameLifetime**
  - Well-known - HKLM\SYSTEM\CurrentControlSet\Control\Notifications
  - Permanent - HKLM\SOFTWARE\Microsoft\Windows NT\ CurrentVersion\Notifications
  - Persistent - HKLM\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\VolatileNotifications
  - Temporary - Misma vida que el proceso que los crea



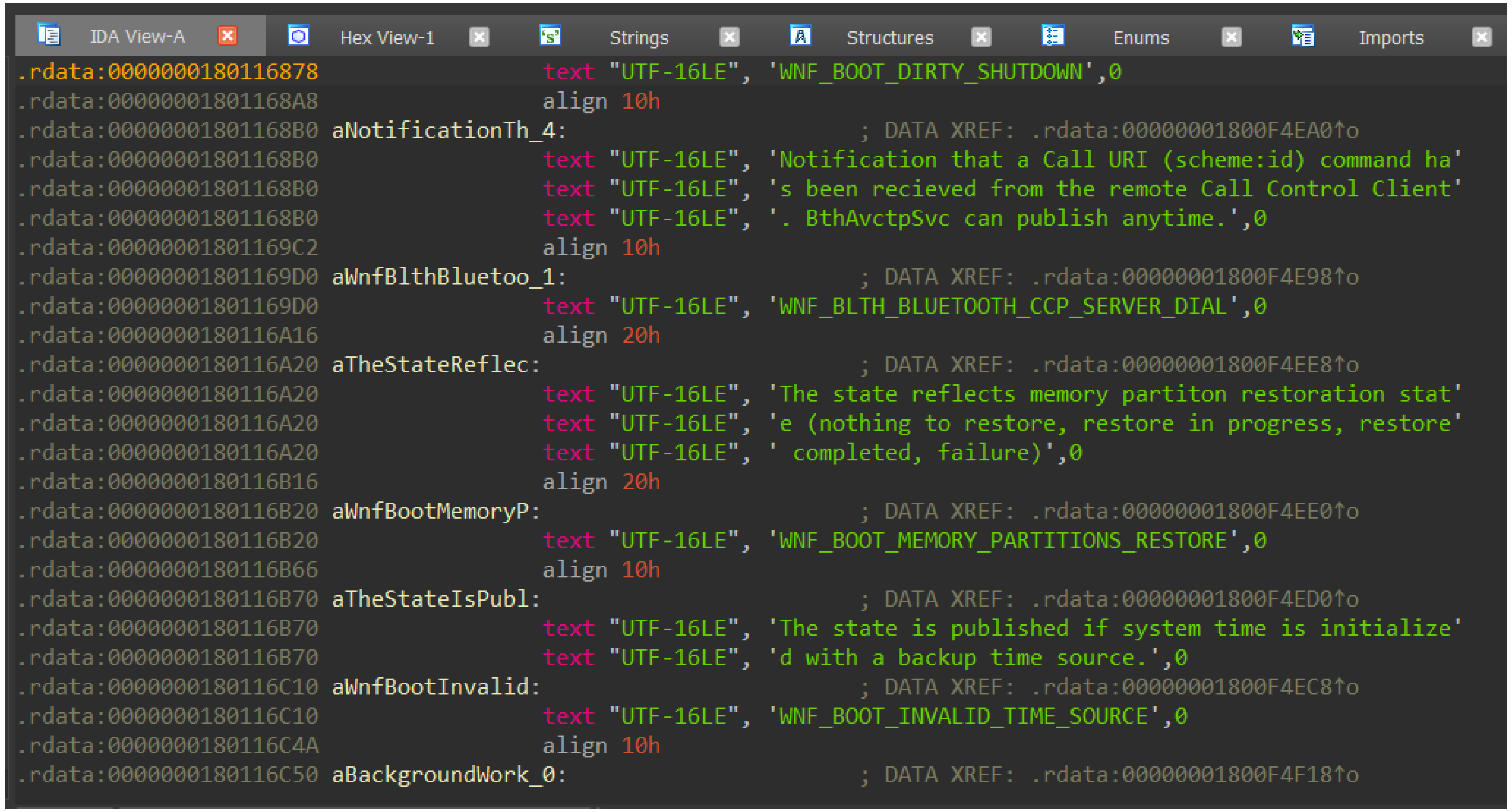
```
lkd> dt nt!_WNF_STATE_NAME_STRUCT
   +0x000 Version          : Pos 0, 4 Bits
   +0x000 NameLifetime     : Pos 4, 2 Bits
   +0x000 DataScope        : Pos 6, 4 Bits
   +0x000 PermanentData    : Pos 10, 1 Bit
   +0x000 Sequence         : Pos 11, 53 Bits
```



```
lkd> dt nt!_WNF_STATE_NAME_LIFETIME
   WnfWellKnownStateName = 0n0
   WnfPermanentStateName = 0n1
   WnfPersistentStateName = 0n2
   WnfTemporaryStateName = 0n3
```

WNF STATE NAMES

C:\Windows\System32\ContentDeliveryManager.Utilities.dll

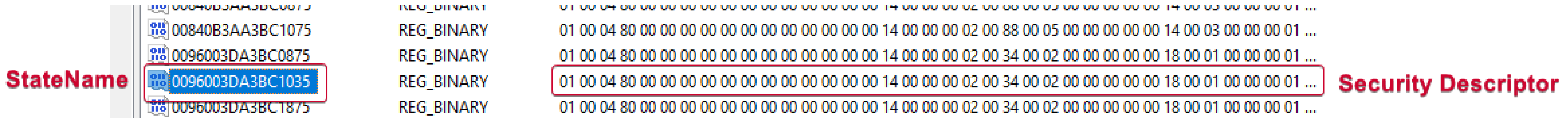


```
.rdata:0000000180116878                     text "UTF-16LE", 'WNF_BOOT_DIRTY_SHUTDOWN',0
.rdata:00000001801168A8                     align 10h
.rdata:00000001801168B0 aNotificationTh_4:                      ; DATA XREF: .rdata:00000001800F4EA0↑o
.rdata:00000001801168B0                     text "UTF-16LE", 'Notification that a Call URI (scheme:id) command ha'
.rdata:00000001801168B0                     text "UTF-16LE", 's been recieved from the remote Call Control Client'
.rdata:00000001801168B0                     text "UTF-16LE", '. BthAvctpSvc can publish anytime.',0
.rdata:00000001801169C2                     align 10h
.rdata:00000001801169D0 aWnfBlthBluetoo_1:                      ; DATA XREF: .rdata:00000001800F4E98↑o
.rdata:00000001801169D0                     text "UTF-16LE", 'WNF_BLTH_BLUETOOTH_CCP_SERVER_DIAL',0
.rdata:0000000180116A16                     align 20h
.rdata:0000000180116A20 aTheStateReflec:                        ; DATA XREF: .rdata:00000001800F4EE8↑o
.rdata:0000000180116A20                     text "UTF-16LE", 'The state reflects memory partiton restoration stat'
.rdata:0000000180116A20                     text "UTF-16LE", 'e (nothing to restore, restore in progress, restore'
.rdata:0000000180116A20                     text "UTF-16LE", ' completed, failure)',0
.rdata:0000000180116B16                     align 20h
.rdata:0000000180116B20 aWnfBootMemoryP:                        ; DATA XREF: .rdata:00000001800F4EE0↑o
.rdata:0000000180116B20                     text "UTF-16LE", 'WNF_BOOT_MEMORY_PARTITIONS_RESTORE',0
.rdata:0000000180116B66                     align 10h
.rdata:0000000180116B70 aTheStateIsPubl:                        ; DATA XREF: .rdata:00000001800F4ED0↑o
.rdata:0000000180116B70                     text "UTF-16LE", 'The state is published if system time is initialize'
.rdata:0000000180116B70                     text "UTF-16LE", 'd with a backup time source.',0
.rdata:0000000180116C10 aWnfBootInvalid:                        ; DATA XREF: .rdata:00000001800F4EC8↑o
.rdata:0000000180116C10                     text "UTF-16LE", 'WNF_BOOT_INVALID_TIME_SOURCE',0
.rdata:0000000180116C4A                     align 10h
.rdata:0000000180116C50 aBackgroundWork_0:                      ; DATA XREF: .rdata:00000001800F4F18↑o
```

Data Scopes & Permissions

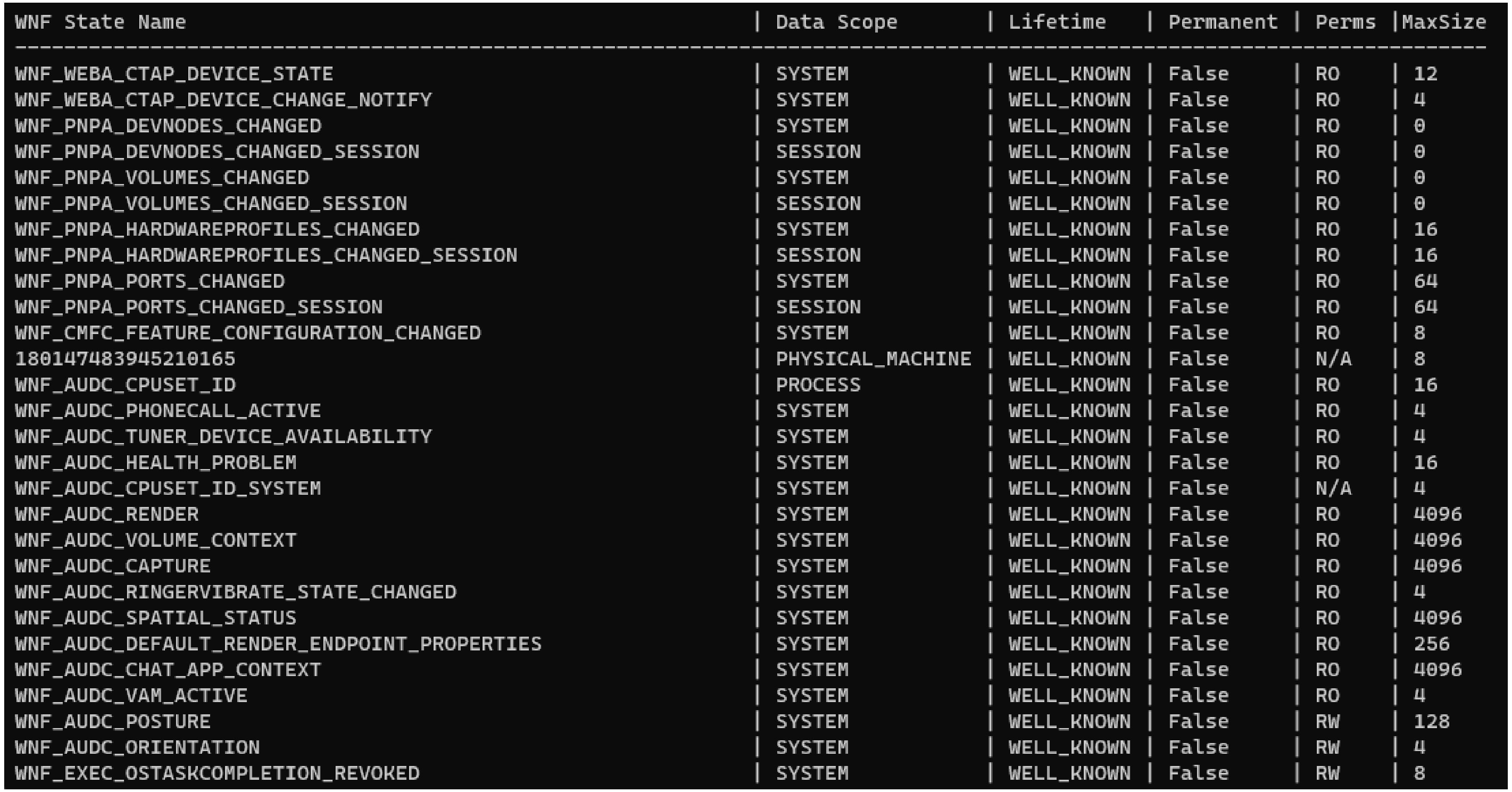
- Quién puede acceder
- *System, session (sess ID), user (SID), process (EPROCESS address) o machine scopes*

```
lkd> dt nt!_WNF_DATA_SCOPE
   WnfDataScopeSystem = 0n0
   WnfDataScopeSession = 0n1
   WnfDataScopeUser = 0n2
   WnfDataScopeProcess = 0n3
   WnfDataScopeMachine = 0n4
   WnfDataScopePhysicalMachine = 0n5
```



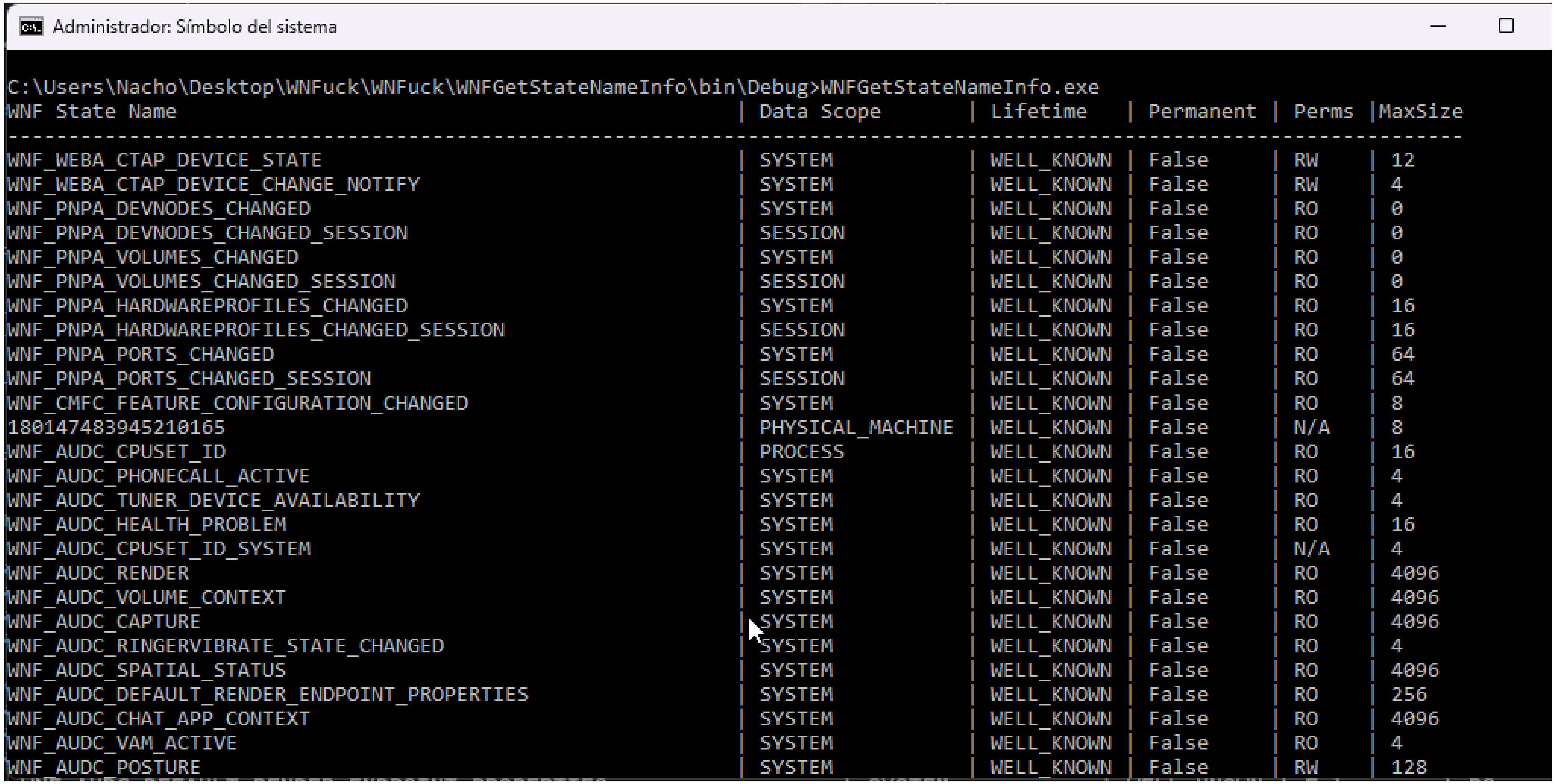
```
00840B3AA3BC0875   REG_BINARY   01 00 04 80 00 00 00 00 00 00 00 00 00 00 00 00 14 00 00 00 02 00 88 00 05 00 00 00 00 00 14 00 03 00 00 00 01 ...
00840B3AA3BC1075   REG_BINARY   01 00 04 80 00 00 00 00 00 00 00 00 00 00 00 00 14 00 00 00 02 00 88 00 05 00 00 00 00 00 14 00 03 00 00 00 01 ...
0096003DA3BC0875   REG_BINARY   01 00 04 80 00 00 00 00 00 00 00 00 00 00 00 00 14 00 00 00 02 00 34 00 02 00 00 00 00 00 18 00 01 00 00 00 01 ...
0096003DA3BC1035   REG_BINARY   01 00 04 80 00 00 00 00 00 00 00 00 00 00 00 00 14 00 00 00 02 00 34 00 02 00 00 00 00 00 18 00 01 00 00 00 01 ...
0096003DA3BC1875   REG_BINARY   01 00 04 80 00 00 00 00 00 00 00 00 00 00 00 00 14 00 00 00 02 00 34 00 02 00 00 00 00 00 18 00 01 00 00 00 01 ...
```

WNF STATE NAMES



```
WNF State Name                                               | Data Scope        | Lifetime   | Permanent  | Perms  |MaxSize
-------------------------------------------------------------------------------------------------------------------------
WNF_WEBA_CTAP_DEVICE_STATE                                   | SYSTEM            | WELL_KNOWN | False      | RO     | 12
WNF_WEBA_CTAP_DEVICE_CHANGE_NOTIFY                           | SYSTEM            | WELL_KNOWN | False      | RO     | 4
WNF_PNPA_DEVNODES_CHANGED                                    | SYSTEM            | WELL_KNOWN | False      | RO     | 0
WNF_PNPA_DEVNODES_CHANGED_SESSION                            | SESSION           | WELL_KNOWN | False      | RO     | 0
WNF_PNPA_VOLUMES_CHANGED                                     | SYSTEM            | WELL_KNOWN | False      | RO     | 0
WNF_PNPA_VOLUMES_CHANGED_SESSION                             | SESSION           | WELL_KNOWN | False      | RO     | 0
WNF_PNPA_HARDWAREPROFILES_CHANGED                            | SYSTEM            | WELL_KNOWN | False      | RO     | 16
WNF_PNPA_HARDWAREPROFILES_CHANGED_SESSION                    | SESSION           | WELL_KNOWN | False      | RO     | 16
WNF_PNPA_PORTS_CHANGED                                       | SYSTEM            | WELL_KNOWN | False      | RO     | 64
WNF_PNPA_PORTS_CHANGED_SESSION                               | SESSION           | WELL_KNOWN | False      | RO     | 64
WNF_CMFC_FEATURE_CONFIGURATION_CHANGED                       | SYSTEM            | WELL_KNOWN | False      | RO     | 8
180147483945210165                                           | PHYSICAL_MACHINE  | WELL_KNOWN | False      | N/A    | 8
WNF_AUDC_CPUSET_ID                                           | PROCESS           | WELL_KNOWN | False      | RO     | 16
WNF_AUDC_PHONECALL_ACTIVE                                    | SYSTEM            | WELL_KNOWN | False      | RO     | 4
WNF_AUDC_TUNER_DEVICE_AVAILABILITY                           | SYSTEM            | WELL_KNOWN | False      | RO     | 4
WNF_AUDC_HEALTH_PROBLEM                                      | SYSTEM            | WELL_KNOWN | False      | RO     | 16
WNF_AUDC_CPUSET_ID_SYSTEM                                    | SYSTEM            | WELL_KNOWN | False      | N/A    | 4
WNF_AUDC_RENDER                                              | SYSTEM            | WELL_KNOWN | False      | RO     | 4096
WNF_AUDC_VOLUME_CONTEXT                                      | SYSTEM            | WELL_KNOWN | False      | RO     | 4096
WNF_AUDC_CAPTURE                                             | SYSTEM            | WELL_KNOWN | False      | RO     | 4096
WNF_AUDC_RINGERVIBRATE_STATE_CHANGED                         | SYSTEM            | WELL_KNOWN | False      | RO     | 4
WNF_AUDC_SPATIAL_STATUS                                      | SYSTEM            | WELL_KNOWN | False      | RO     | 4096
WNF_AUDC_DEFAULT_RENDER_ENDPOINT_PROPERTIES                  | SYSTEM            | WELL_KNOWN | False      | RO     | 256
WNF_AUDC_CHAT_APP_CONTEXT                                    | SYSTEM            | WELL_KNOWN | False      | RO     | 4096
WNF_AUDC_VAM_ACTIVE                                          | SYSTEM            | WELL_KNOWN | False      | RO     | 4
WNF_AUDC_POSTURE                                             | SYSTEM            | WELL_KNOWN | False      | RW     | 128
WNF_AUDC_ORIENTATION                                         | SYSTEM            | WELL_KNOWN | False      | RW     | 4
WNF_EXEC_OSTASKCOMPLETION_REVOKED                            | SYSTEM            | WELL_KNOWN | False      | RW     | 8
```

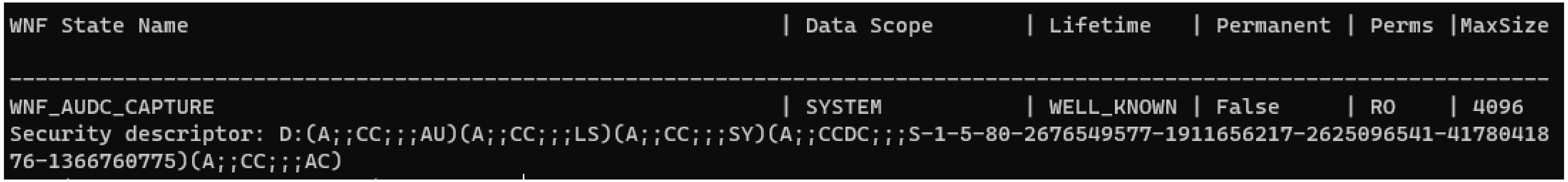
WNF STATE NAMES



Administrador: Símbolo del sistema

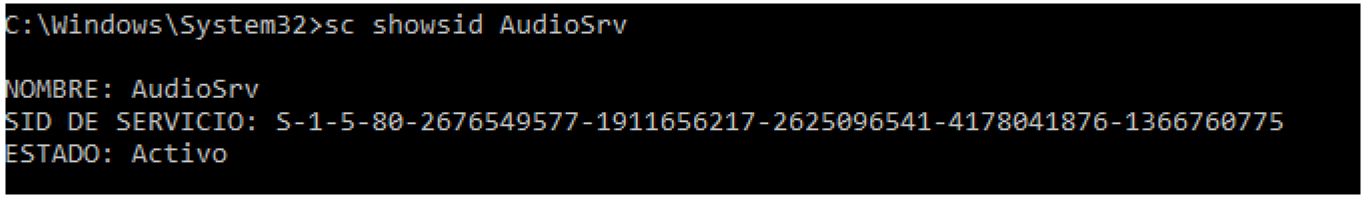
```
C:\Users\Nacho\Desktop\WNFuck\WNFuck\WNFGetStateNameInfo\bin\Debug>WNFGetStateNameInfo.exe
WNF State Name                                                | Data Scope        | Lifetime    | Permanent  | Perms  |MaxSize
--------------------------------------------------------------------------------------------------------------------------------
WNF_WEBA_CTAP_DEVICE_STATE                                    | SYSTEM            | WELL_KNOWN  | False      | RW     | 12
WNF_WEBA_CTAP_DEVICE_CHANGE_NOTIFY                            | SYSTEM            | WELL_KNOWN  | False      | RW     | 4
WNF_PNPA_DEVNODES_CHANGED                                     | SYSTEM            | WELL_KNOWN  | False      | RO     | 0
WNF_PNPA_DEVNODES_CHANGED_SESSION                             | SESSION           | WELL_KNOWN  | False      | RO     | 0
WNF_PNPA_VOLUMES_CHANGED                                      | SYSTEM            | WELL_KNOWN  | False      | RO     | 0
WNF_PNPA_VOLUMES_CHANGED_SESSION                              | SESSION           | WELL_KNOWN  | False      | RO     | 0
WNF_PNPA_HARDWAREPROFILES_CHANGED                             | SYSTEM            | WELL_KNOWN  | False      | RO     | 16
WNF_PNPA_HARDWAREPROFILES_CHANGED_SESSION                     | SESSION           | WELL_KNOWN  | False      | RO     | 16
WNF_PNPA_PORTS_CHANGED                                        | SYSTEM            | WELL_KNOWN  | False      | RO     | 64
WNF_PNPA_PORTS_CHANGED_SESSION                                | SESSION           | WELL_KNOWN  | False      | RO     | 64
WNF_CMFC_FEATURE_CONFIGURATION_CHANGED                        | SYSTEM            | WELL_KNOWN  | False      | RO     | 8
1801474839452l0165                                            | PHYSICAL_MACHINE  | WELL_KNOWN  | False      | N/A    | 8
WNF_AUDC_CPUSET_ID                                            | PROCESS           | WELL_KNOWN  | False      | RO     | 16
WNF_AUDC_PHONECALL_ACTIVE                                     | SYSTEM            | WELL_KNOWN  | False      | RO     | 4
WNF_AUDC_TUNER_DEVICE_AVAILABILITY                            | SYSTEM            | WELL_KNOWN  | False      | RO     | 4
WNF_AUDC_HEALTH_PROBLEM                                       | SYSTEM            | WELL_KNOWN  | False      | RO     | 16
WNF_AUDC_CPUSET_ID_SYSTEM                                     | SYSTEM            | WELL_KNOWN  | False      | N/A    | 4
WNF_AUDC_RENDER                                               | SYSTEM            | WELL_KNOWN  | False      | RO     | 4096
WNF_AUDC_VOLUME_CONTEXT                                       | SYSTEM            | WELL_KNOWN  | False      | RO     | 4096
WNF_AUDC_CAPTURE                                              | SYSTEM            | WELL_KNOWN  | False      | RO     | 4096
WNF_AUDC_RINGERVIBRATE_STATE_CHANGED                          | SYSTEM            | WELL_KNOWN  | False      | RO     | 4
WNF_AUDC_SPATIAL_STATUS                                       | SYSTEM            | WELL_KNOWN  | False      | RO     | 4096
WNF_AUDC_DEFAULT_RENDER_ENDPOINT_PROPERTIES                   | SYSTEM            | WELL_KNOWN  | False      | RO     | 256
WNF_AUDC_CHAT_APP_CONTEXT                                     | SYSTEM            | WELL_KNOWN  | False      | RO     | 4096
WNF_AUDC_VAM_ACTIVE                                           | SYSTEM            | WELL_KNOWN  | False      | RO     | 4
WNF_AUDC_POSTURE                                              | SYSTEM            | WELL_KNOWN  | False      | RW     | 128
```

WNF STATE NAMES



```
WNF State Name                                           | Data Scope        | Lifetime    | Permanent | Perms  |MaxSize

--------------------------------------------------------------------------------------------------------------------------
WNF_AUDC_CAPTURE                                         | SYSTEM            | WELL_KNOWN  | False     | RO     | 4096
Security descriptor: D:(A;;CC;;;AU)(A;;CC;;;LS)(A;;CC;;;SY)(A;;CCDC;;;S-1-5-80-2676549577-1911656217-2625096541-41780418
76-1366760775)(A;;CC;;;AC)
```

WNF_AUDC_CAPTURE únicamente tiene RW para el servicio AudioSrv



```
C:\Windows\System32>sc showsid AudioSrv

NOMBRE: AudioSrv
SID DE SERVICIO: S-1-5-80-2676549577-1911656217-2625096541-4178041876-1366760775
ESTADO: Activo
```

What the (WNF)uck?!

Nacho Gómez aka *nag0mez*

24 y 25 FEBRERO

MADRID 2023

WINDOWS APIs

Publishing Data

```
[DllImport("ntdll.dll", SetLastError = true)]
public static extern NTSTATUS NtUpdateWnfStateData(
    in ulong StateName,
    byte[] Buffer,
    int Length, // Less than MaximumSize, always less than 4096
    IntPtr TypeId,
    IntPtr ExplicitScope,
    int MatchingChangeScope, // Last TimeStamp
    int CheckStamp); // If 1, publish only if MatchingChangeScope is valid
```

Check Client Demo. Also NtDeleteWnfStateData

- WNF se usa activamente en Windows 10/11.
- Al publicar una notificación, sobreescribes su estado previo (puede que aún no haya sido consumido).
- También aumenta el stamp, por lo que si algún proceso intenta publicar comprobando el stamp previo, fallará.
- Es posible dejar el SO en un estado difícil de recuperar! (Especialmente EXPLORER.EXE)

Querying Data

```
[DllImport("ntdll.dll")]
public static extern NTSTATUS NtQueryWnfStateData(
    in ulong StateName,
    IntPtr TypeId,
    IntPtr ExplicitScope,
    out int ChangeStamp,
    byte[] Buffer, // Output
    out int BufferSize);
```

Check Consumer Demo

Creating Names

```
[DllImport("ntdll.dll")]
public static extern NTSTATUS NtCreateWnfStateName(
    out ulong StateName,
    WNF_STATE_NAME_LIFETIME NameLifetime,
    WNF_DATA_SCOPE DataScope,
    bool PersistData,
    IntPtr TypeId,
    int MaximumStateSize,
    SafeMemoryHandle SecurityDescriptor);
```

Check CreateTemporaryName Demo. Also NtDeleteWnfStateName

Subscribing To State Changes

```
[DllImport("ntdll.dll")]
public static extern NTSTATUS RtlSubscribeWnfStateChangeNotification(
    out IntPtr Subscription,
    ulong StateName,
    int ChangeStamp,
    IntPtr Callback,
    IntPtr CallbackContext,
    IntPtr TypeId,
    int SerializationGroup,
    int Unknown);
```

Check Server Demo

Callback Prototype

```
[UnmanagedFunctionPointer(CallingConvention.StdCall)]
private delegate NTSTATUS CallbackDelegate(
    ulong StateName,
    int ChangeStamp,
    IntPtr TypeId,
    IntPtr CallbackContext,
    IntPtr Buffer,
    int BufferSize);
```

Unsubscribing

```
[DllImport("ntdll.dll")]
public static extern NTSTATUS RtlUnsubscribeWnfStateChangeNotification(
    IntPtr Subscription);
```

What the (WNF)uck?!

Nacho Gómez aka *nag0mez*

24 y 25 FEBRERO

MADRID 2023

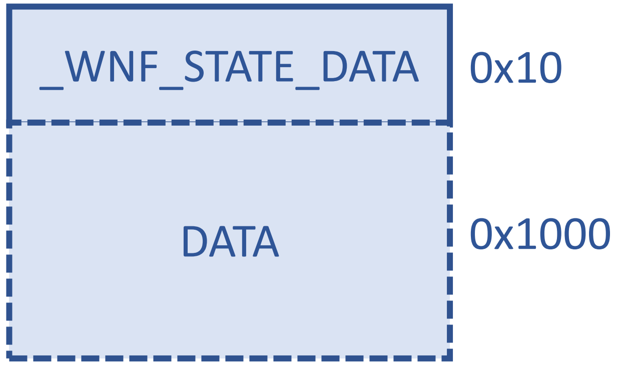
WNF STRUCTURES

WNF STRUCTURES

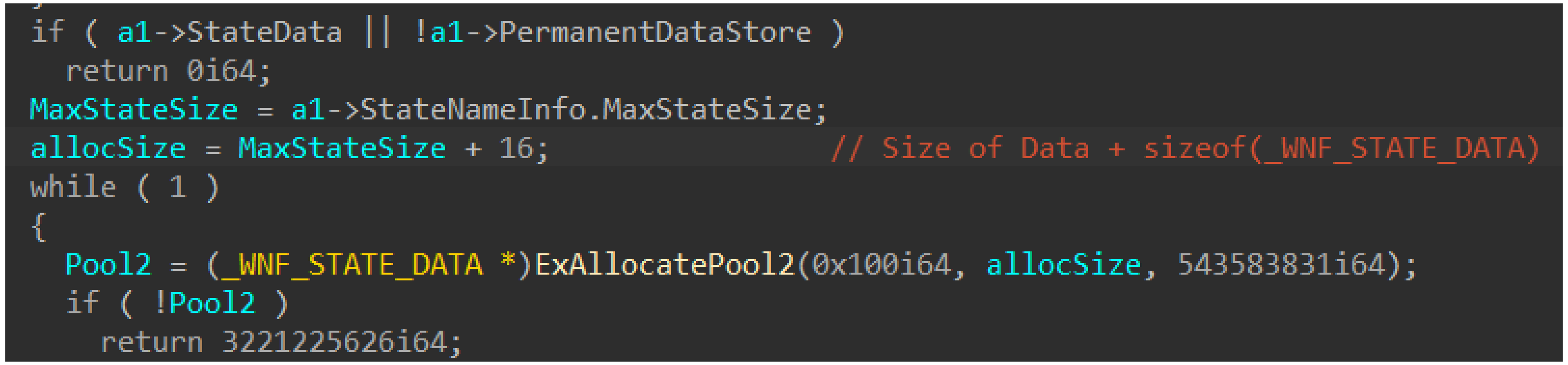
```
//0xa8 bytes (sizeof)
struct _WNF_NAME_INSTANCE
{
    struct _WNF_NODE_HEADER Header;
    struct _EX_RUNDOWN_REF RunRef;
    struct _RTL_BALANCED_NODE TreeLinks;
    struct _WNF_STATE_NAME_STRUCT StateName;
    struct _WNF_SCOPE_INSTANCE* ScopeInstance;
    struct _WNF_STATE_NAME_REGISTRATION StateNameInfo;
    struct _WNF_LOCK StateDataLock;
    struct _WNF_STATE_DATA* StateData;
    ULONG CurrentChangeStamp;
    VOID* PermanentDataStore;
    struct _WNF_LOCK StateSubscriptionListLock;
    struct _LIST_ENTRY StateSubscriptionListHead;
    struct _LIST_ENTRY TemporaryNameListEntry;
    struct _EPROCESS* CreatorProcess;
    LONG DataSubscribersCount;
    LONG CurrentDeliveryCount;
};
```

```
lkd> dt nt!_WNF_STATE_NAME_STRUCT
   +0x000 Version          : Pos 0, 4 Bits
   +0x000 NameLifetime     : Pos 4, 2 Bits
   +0x000 DataScope        : Pos 6, 4 Bits
   +0x000 PermanentData    : Pos 10, 1 Bit
   +0x000 Sequence         : Pos 11, 53 Bits
```

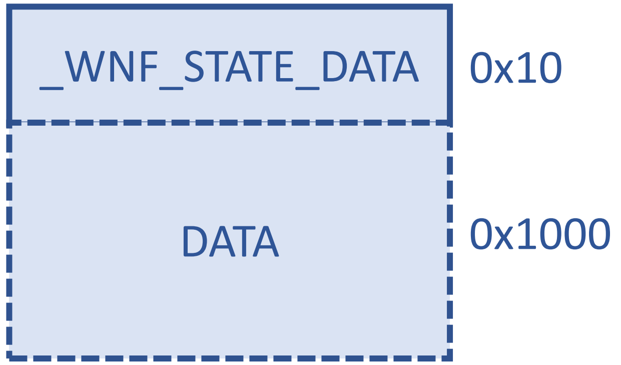
```
//0x10 bytes (sizeof)
struct _WNF_STATE_DATA
{
    struct _WNF_NODE_HEADER Header;
    ULONG AllocatedSize;
    ULONG DataSize;
    ULONG ChangeStamp;
};
```



WNF STRUCTURES



```
if ( a1->StateData || !a1->PermanentDataStore )
  return 0i64;
MaxStateSize = a1->StateNameInfo.MaxStateSize;
allocSize = MaxStateSize + 16;              // Size of Data + sizeof(_WNF_STATE_DATA)
while ( 1 )
{
  Pool2 = (_WNF_STATE_DATA *)ExAllocatePool2(0x100i64, allocSize, 543583831i64);
  if ( !Pool2 )
    return 3221225626i64;
```



_WNF_SUBSCRIPTION_TABLE

```
public struct WNF_SUBSCRIPTION_TABLE64_WIN11
{
    public WNF_CONTEXT_HEADER Header;
    public long NamesTableLock;
    public RTL_RB_TREE64 NamesTableEntry;
    public LIST_ENTRY64 SerializationGroupListHead;
    public long SerializationGroupLock;
    public int[] Unknown1;
    public int SubscribedEventSet;
    public int[] Unknown2;
    public long Timer;
    public ulong TimerDueTime;
}
```

```
[StructLayout(LayoutKind.Sequential)]
internal struct WNF_CONTEXT_HEADER
{
    public short NodeTypeCode; // 0x911
    public short NodeByteSize;
}
```

_WNF_NAME_SUBSCRIPTION[] (Win10)
RTL_RB_TREE, Red-Black Tree (Win11)

Para encontrar la tabla podemos recorrer el Heap de NTDLL leyendo estructuras sizeof(WNF_SUBSCRIPTION_TABLE) y comparando el valor del NodeTypeCode con 0x911

_WNF_NAME_SUBSCRIPTION

```
public struct WNF_NAME_SUBSCRIPTION64_WIN11
{
    public WNF_CONTEXT_HEADER Header;
    public ulong SubscriptionId;
    public ulong StateName;
    public uint CurrentChangeStamp;
    public RTL_BALANCED_NODE64 NamesTableEntry;
    public long TypeId;
    public long SubscriptionLock;
    public LIST_ENTRY64 SubscriptionsListHead;
    public uint NormalDeliverySubscriptions;
    public uint[] NotificationTypeCount;
    public long RetryDescriptor;
    public uint DeliveryState;
    public ulong ReliableRetryTime;
}
```

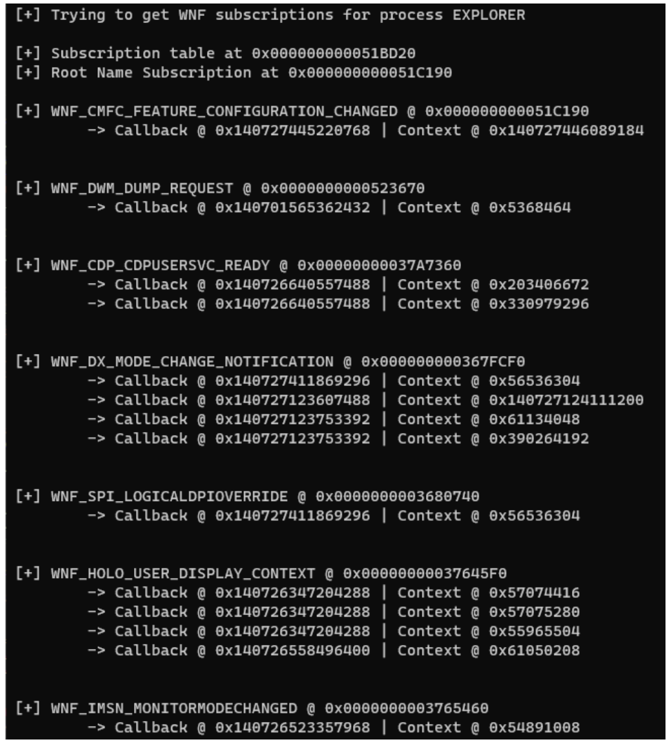
NamesTableEntry ← _WNF_SUBSCRIPTION_TABLE -> NamesTableEntry -> Root

SubscriptionsListHead → _WNF_USER_SUBSCRIPTION List

La propiedad NamesTableEntry es un nodo balanceado que apunta a las siguientes _WNF_NAME_SUBSCRIPTIONS, por lo que podemos recorrer el árbol. Todas las referencias apuntan a esta propiedad, así que es necesario restar el offset.

_WNF_USER_SUBSCRIPTION

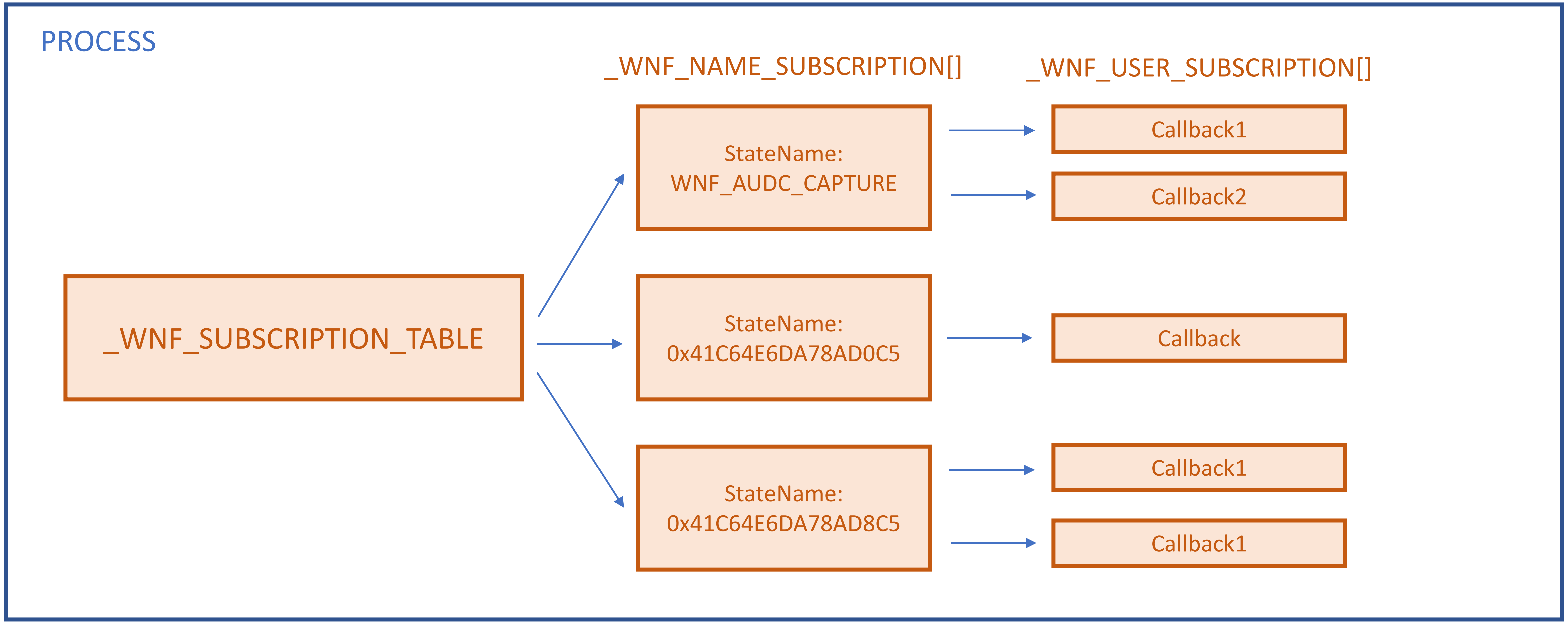
```
internal struct WNF_USER_SUBSCRIPTION64
{
    public WNF_CONTEXT_HEADER Header;
    public LIST_ENTRY64 SubscriptionsListEntry;
    public long NameSubscription;
    public long Callback;
    public long CallbackContext;
    public ulong SubProcessTag;
    public uint CurrentChangeStamp;
    public uint DeliveryOptions;
    public uint SubscribedEventSet;
    public long SerializationGroup;
    public uint UserSubscriptionCount;
    public ulong[] Unknown;
}
```



```
[+] Trying to get WNF subscriptions for process EXPLORER

[+] Subscription table at 0x000000000051BD20
[+] Root Name Subscription at 0x000000000051C190

[+] WNF_CMFC_FEATURE_CONFIGURATION_CHANGED @ 0x000000000051C190
      -> Callback @ 0x140727445220768 | Context @ 0x140727446089184


[+] WNF_DWM_DUMP_REQUEST @ 0x0000000000523670
      -> Callback @ 0x140701565362432 | Context @ 0x5368464


[+] WNF_CDP_CDPUSERSVC_READY @ 0x00000000037A7360
      -> Callback @ 0x140726640557488 | Context @ 0x203406672
      -> Callback @ 0x140726640557488 | Context @ 0x330979296


[+] WNF_DX_MODE_CHANGE_NOTIFICATION @ 0x000000000367FCF0
      -> Callback @ 0x140727411869296 | Context @ 0x56536304
      -> Callback @ 0x140727123607488 | Context @ 0x140727124111200
      -> Callback @ 0x140727123753392 | Context @ 0x61134048
      -> Callback @ 0x140727123753392 | Context @ 0x390264192


[+] WNF_SPI_LOGICALDPIOVERRIDE @ 0x0000000003680740
      -> Callback @ 0x140727411869296 | Context @ 0x56536304


[+] WNF_HOLO_USER_DISPLAY_CONTEXT @ 0x00000000037645F0
      -> Callback @ 0x140726347204288 | Context @ 0x57074416
      -> Callback @ 0x140726347204288 | Context @ 0x57075280
      -> Callback @ 0x140726347204288 | Context @ 0x55965504
      -> Callback @ 0x140726558496400 | Context @ 0x61050208


[+] WNF_IMSN_MONITORMODECHANGED @ 0x0000000003765460
      -> Callback @ 0x140726523357968 | Context @ 0x54891008
```

Estructura que almacena los callbacks y sus contextos. SubscriptionsListEntry contiene un flink apuntando a la siguiente _WNF_USER_SUBSCRIPTION. El proyecto BasicDemo-CallbackParser puede enumerar las callbacks para un proceso.

WNF STRUCTURES



H-CON HACKPLAYERS CONFERENCE
FIVE

WNF POTENTIAL

What?

- DoS: por ejemplo, EXPLORER.EXE
- Activar/Desactivar Windows Insider Features (https://github.com/riverar/mach2)
- Forzar comportamientos inesperados (ocultar icono del micrófono, entre otros)
- Obtención de información por vías alternativas (WNF_EDGE_LAST_NAVIGATED_HOST)
- Inter-Process Communication
- Process Injection evitando la secuencia Alloc-Protect-CreateThread (Demo)
- Persistencia de datos fuera de memoria (Demo)
- Fuzzing para encontrar DoS, LPEs o incluso Kernel bugs

WNF INJECTION

WNF INJECTION

>h-c0n

How?

1. Encuentra Use[illegible]Subscriptions para el StateN[illegible] que uses como trigger
2. VirtualAllocEx[illegible]RX) + [illegible]ite[illegible]cessMemory para copiar shellcode [illegible]proceso objetivo
3. Modifica los [illegible]llback[illegible]ara e[illegible]ateN[illegible]ne q[illegible]e qu[illegible]s usar para que apunten a tu[illegible]hellcode
4. NtUpdateW[illegible]StateD[illegible]a par[illegible]nza[illegible]ger
5. Cleanup

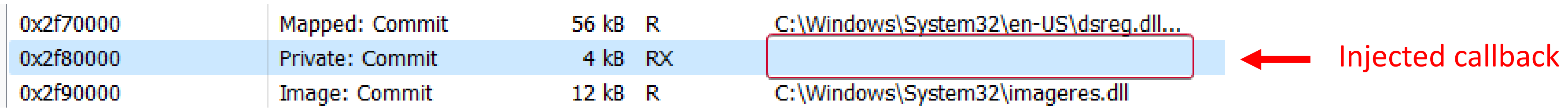
DEMO

Características

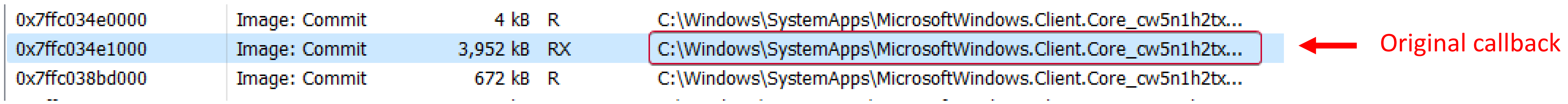
- Evita el uso de CreateThread u otras técnicas de ejecución comunes
- Se ejecuta en threads creados legítimamente
- No ETW!
- Difícil de “tracear”
- Es posible “publicar” el shellcode para que lo reciba un callback del proceso objetivo, por lo que podría llegar a ser posible evitar el VirtualAllocEx + WriteProcessMemory (requiere más research)

Detección

- Shellcode no respaldado en disco



```
0x2f70000    Mapped: Commit    56 kB   R    C:\Windows\System32\en-US\dsreg.dll...
0x2f80000    Private: Commit    4 kB   RX
0x2f90000    Image: Commit     12 kB   R    C:\Windows\System32\imageres.dll
```



```
0x7ffc034e0000    Image: Commit        4 kB   R    C:\Windows\SystemApps\MicrosoftWindows.Client.Core_cw5n1h2tx...
0x7ffc034e1000    Image: Commit    3,952 kB   RX   C:\Windows\SystemApps\MicrosoftWindows.Client.Core_cw5n1h2tx...
0x7ffc038bd000    Image: Commit      672 kB   R    C:\Windows\SystemApps\MicrosoftWindows.Client.Core_cw5n1h2tx...
```

- VirtualAllocEx + WriteProcessMemory sigue siendo algo habitual

What the (WNF)uck?!

Nacho Gómez aka *nag0mez*

24 y 25 FEBRERO

MADRID 2023

WNF DATA PERSISTENCE

WNF DATA PERSISTENCE

>h-c0n

How?

1. Crea tantos temporary State Names como necesites para abarcar todo el conjunto de datos
2. Elige un WellKnown StateName para almacenar la lista de temporary StateNames generados
3. Divide el conjunto de datos en chunks de 4096 bytes y usa NtUpdateWnfStateData para publicarlos
4. Cuando quieras acceder a los datos lee la lista de temporary StateNames desde el WellKnown que hayas elegido y usa NtQueryWnfStateData para obtener los chunks uno a uno
5. Cleanup

DEMO

WNF_XBOX_ACHIEVEMENTS_RAW_NOTIFICATION_RECEIVED
0x41C64E6DA78A10C5
0x41C64E6DA78AD0C5
0x41C64E6DA78AD8C5
0x41C64E6DA78AE8C5

0x41C64E6DA78A10C5

data_chunk_1

0x41C64E6DA78AD0C5

data_chunk_2

0x41C64E6DA78AD8C5

data_chunk_3

Características

- Almacenamiento de datos en "kernel" (no toca disco, no ocupa memoria de proceso)
- "Difícil" de detectar
- Es posible Comprimir/Cifrar
- Puedes usar distintos WellKnown StateNames
- Inter-Processes

What the (WNF)uck?!

Nacho Gómez aka *nag0mez*

24 y 25 FEBRERO

MADRID 2023

HIDING THE MICROPHONE

HIDING THE MICROPHONE

How?

1. WNF_AUDC APTURE / WNF_CAM MICROPHONE_USAGE_CHANGED
2. Explorer est suscr stos dos eventos. Los callbacks los encargados de mostrar el icono del micrófono.
3. Si "cruzas los cabl y apu as e s callback tros no hagan nada", el ono nunca aparecerá
4. Siempre h y que vertir

DEMO

What the (WNF)uck?!

Nacho Gómez aka *nag0mez*

24 y 25 FEBRERO

MADRID 2023

REFERENCIAS

Referencias

- "Windows Notification Facility: Peeling the Onion of the Most Undocumented Kernel Attack Surface Yet", Alex Ionescu + Gabrielle Viala, Black Hat 2018.
- https://blog.quarkslab.com/playing-with-the-windows-notification-facility-wnf.html
- https://modexp.wordpress.com/2019/06/15/4083/
- http://redplait.blogspot.com/2019/01/wnf-ids-from-w10-build-18312.html
- https://github.com/riverar/mach2

H-CON HACKPLAYERS CONFERENCE

THANKS!

Hackplayers c0nference

```
Sending SIGKILL to all processes.
Please stand by while rebooting the system.
[64857.521348] sd 0:0:0:0: [sda] Synchronizing SCSI cache
[64857.522838] Restarting system.

_
```

www.h-c0n.com