ATTACKING WINDOWS BY WINDOWS

Yin Liang & Zhou Li
Tencent PC Manager

Outline

- About us
- How to exploit old Windows OS
- Windows 10`s limit
- New exploit method

About us

Team member:

- xin, godz, ki, michael, kelvin, willj

Achieve:

- Attacking Adobe Flash in Pwn2Own 2016

- Attacking Surface Pro 4 in GeekPwn Macao

- 46 acknowledgments！

How to exploit?

Q1: Where to write?

Q2: What to write?

Q3: What can we do now?

Old days..

Q1: Where to write?

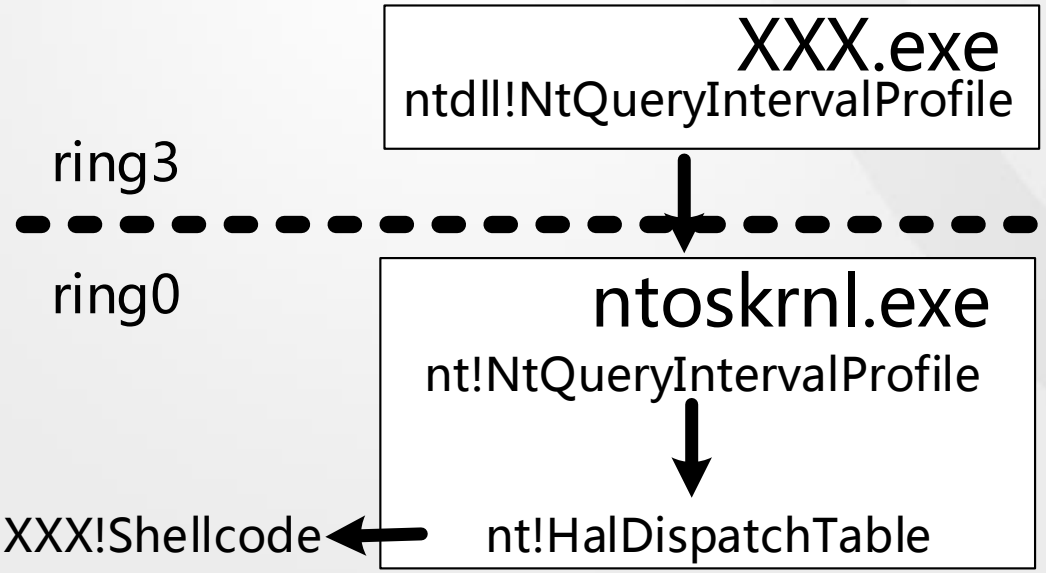
- nt!HalDispatchTable

```
NTSTATUS WINAPI NtQuerySystemInformation(
    _In_      SYSTEM_INFORMATION_CLASS SystemInformationClass,
    _Inout_   PVOID                    SystemInformation,
    _In_      ULONG                    SystemInformationLength,
    _Out_opt_ PULONG                   ReturnLength
    );
```

Q2: What to write?

- Userland shellcode address

Q3: What can we do now?

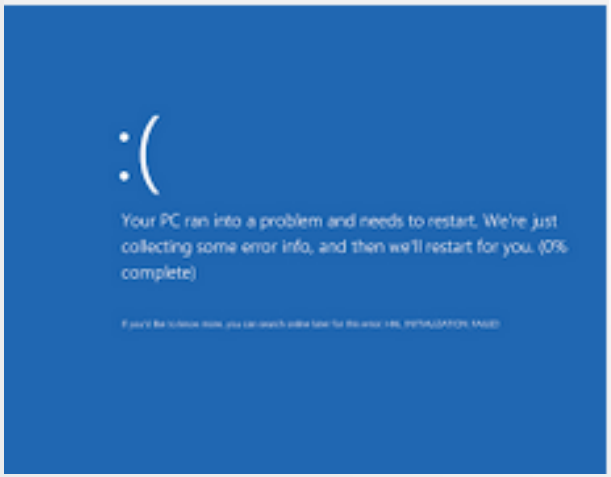


Now..

- Forbid low integrity user

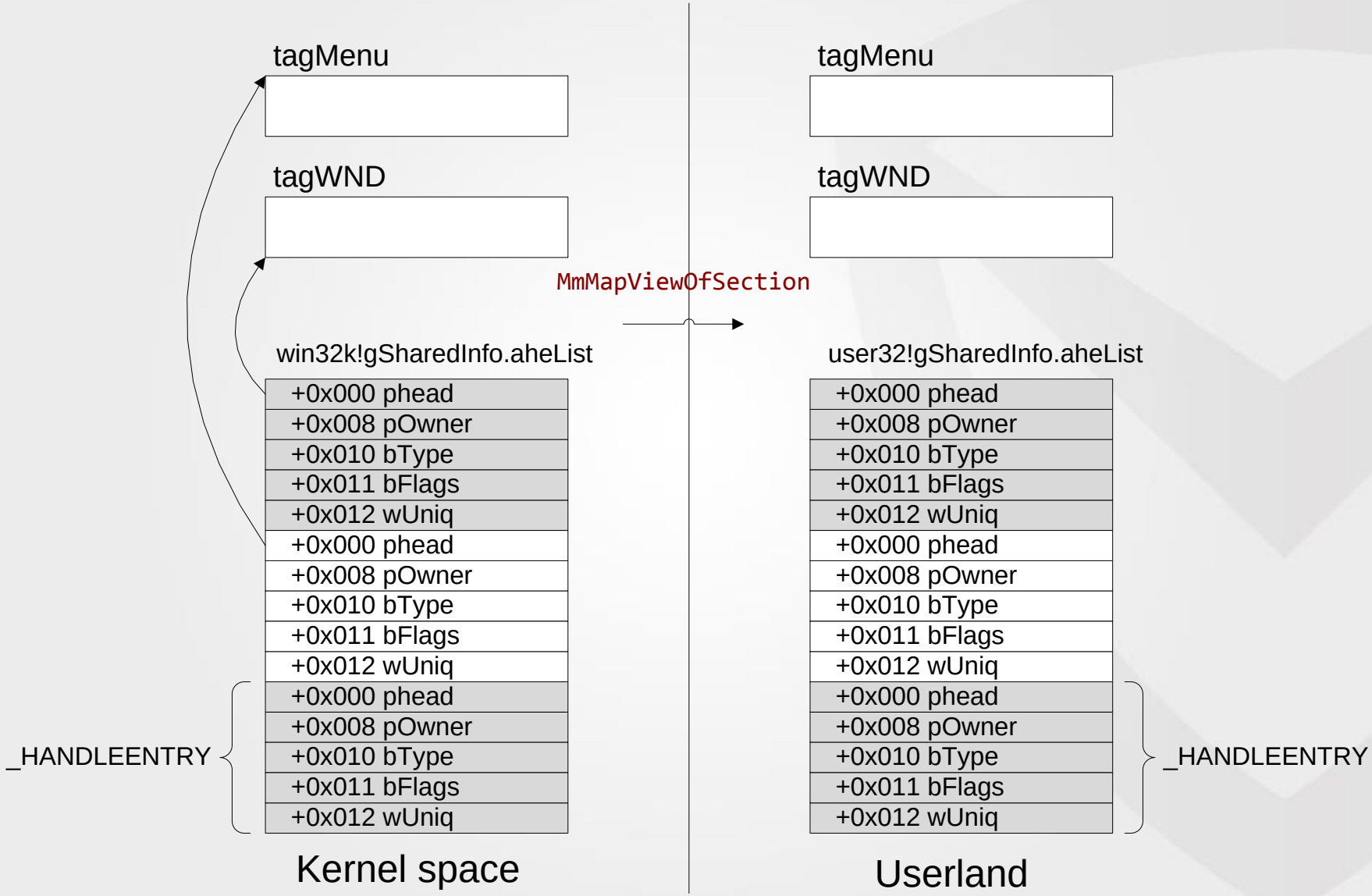
```
loc_14043EF36:          ; jumptable 000000014043EADA case 11
movzx   ecx, r10b
call    ExIsRestrictedCaller
test    eax, eax
jnz     loc_1405D4CBB
```

- SMEP



Where to write?

gSharedInfo



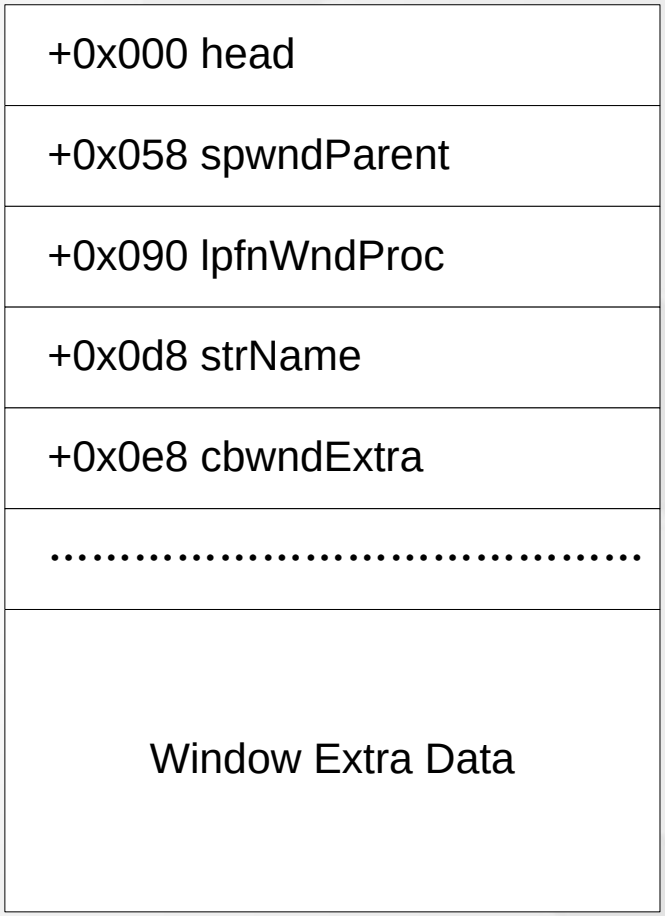
What to write?

Basic object: Window

```
typedef struct tagWNDCLASSW {
    UINT        style;
    WNDPROC     lpfnWndProc;
    int         cbClsExtra;
    int         cbWndExtra;
    HINSTANCE   hInstance;
    HICON       hIcon;
    HCURSOR     hCursor;
    HBRUSH      hbrBackground;
    LPCWSTR     lpszMenuName;
    LPCWSTR     lpszClassName;
} WNDCLASSW, *PWNDCLASSW;
```

```
HWND WINAPI CreateWindowExW(
    __in DWORD dwExStyle,
    __in_opt LPCWSTR lpClassName,
    __in_opt LPCWSTR lpWindowName,
    __in DWORD dwStyle,
    __in int X,
    __in int Y,
    __in int nWidth,
    __in int nHeight,
    __in_opt HWND hWndParent,
    __in_opt HMENU hMenu,
    __in_opt HINSTANCE hInstance,
    __in_opt LPVOID lpParam);
```

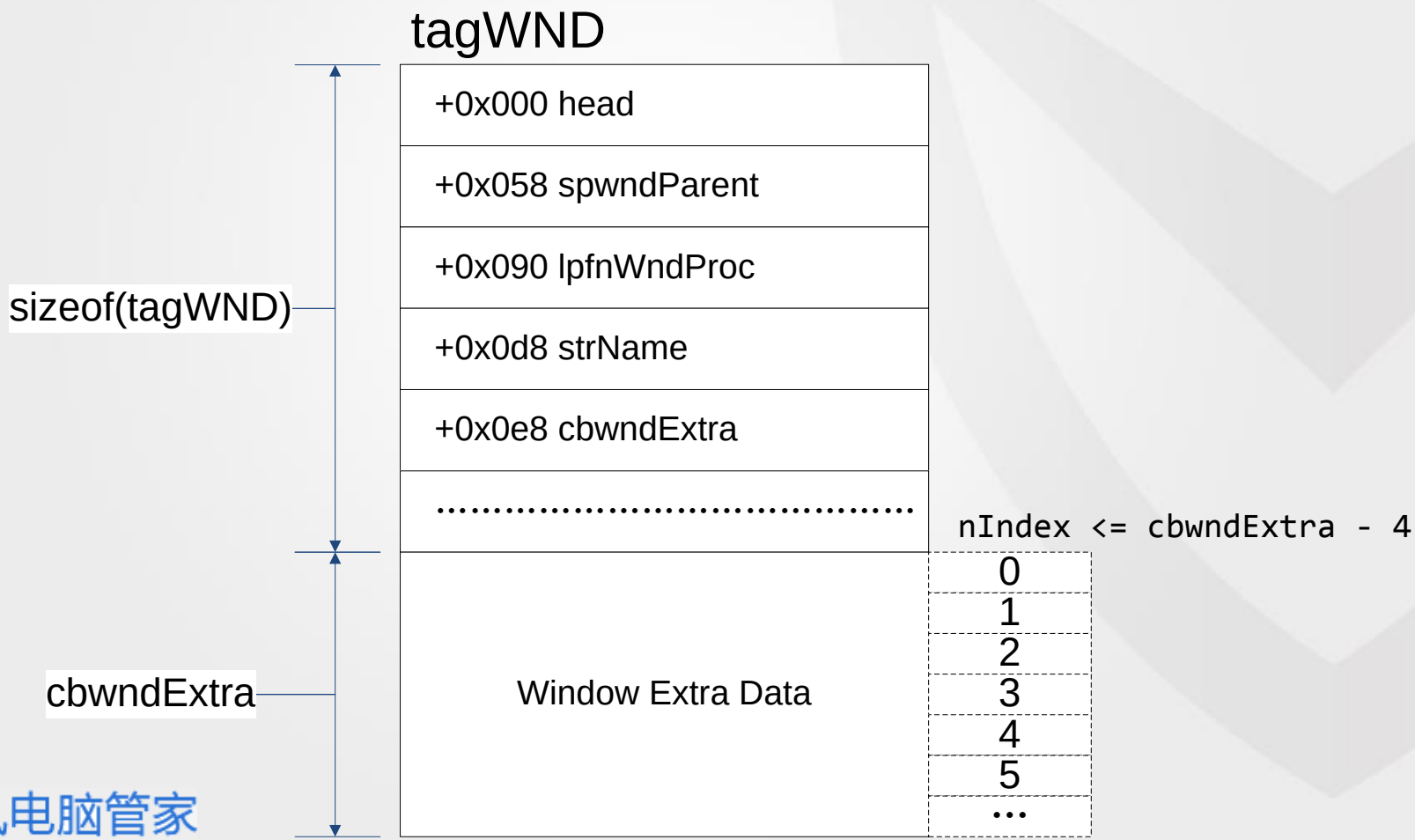
tagWND



Window Extra Data

- Two APIs:

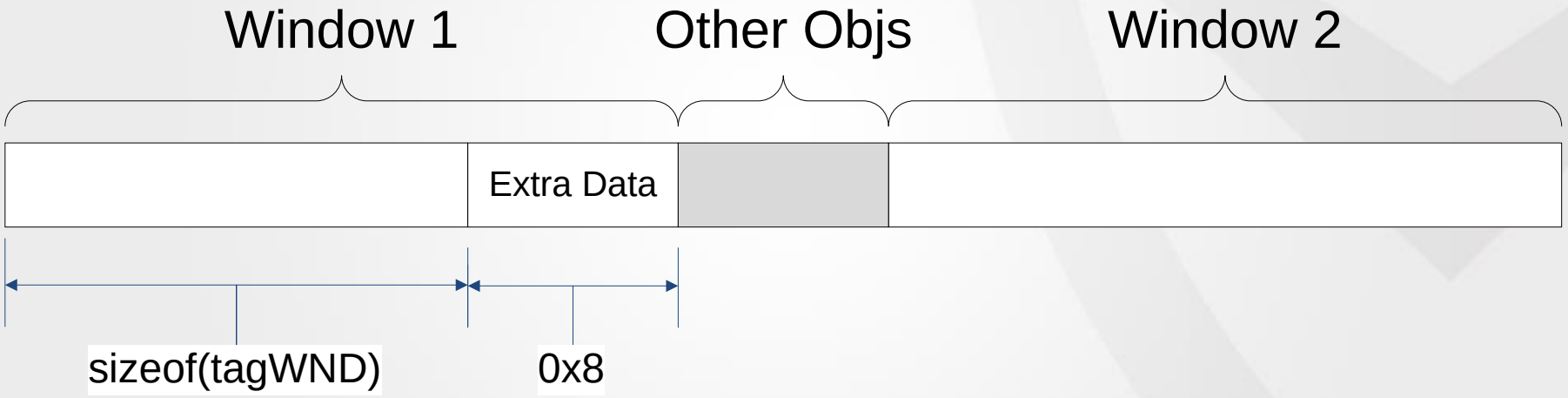
```
LONG WINAPI SetWindowLongW( HWND hWnd, int nIndex, LONG dwNewLong);
LONG WINAPI GetWindowLongW( HWND hWnd, int nIndex);
```



Normal Case

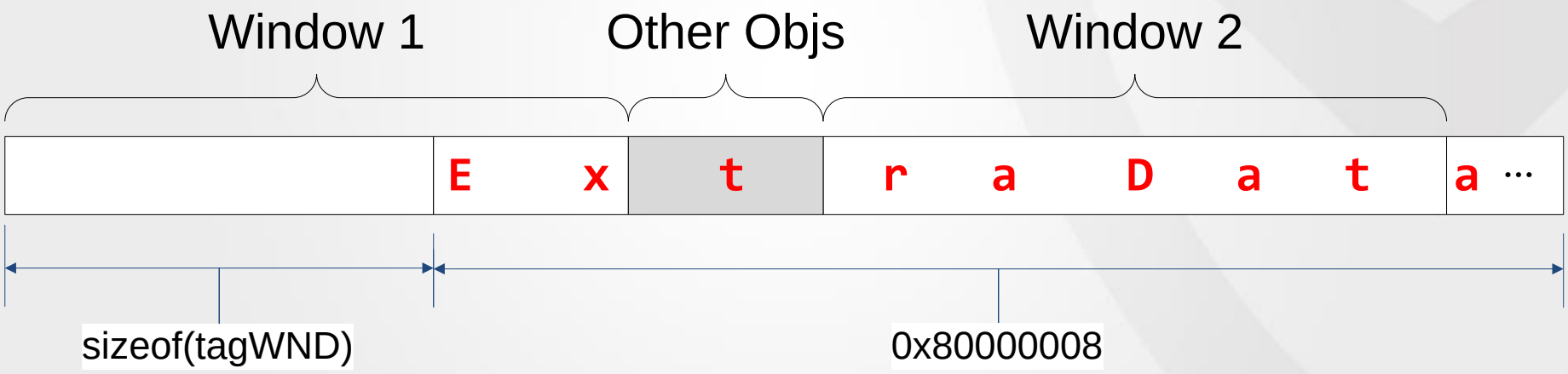
- `cbwndExtra = 0x8`
- `Hex: 0x8`
- `Bin: 0000 0000 0000 0000 0000 0000 0000 1000`

```
      pos  31                              15                              0
```



If we change a bit…

- Bin: 1000 0000 0000 0000 0000 0000 0000 1000
  pos 31 15 0
- Hex: 0x80000008
- cbwndExtra = 0x80000008

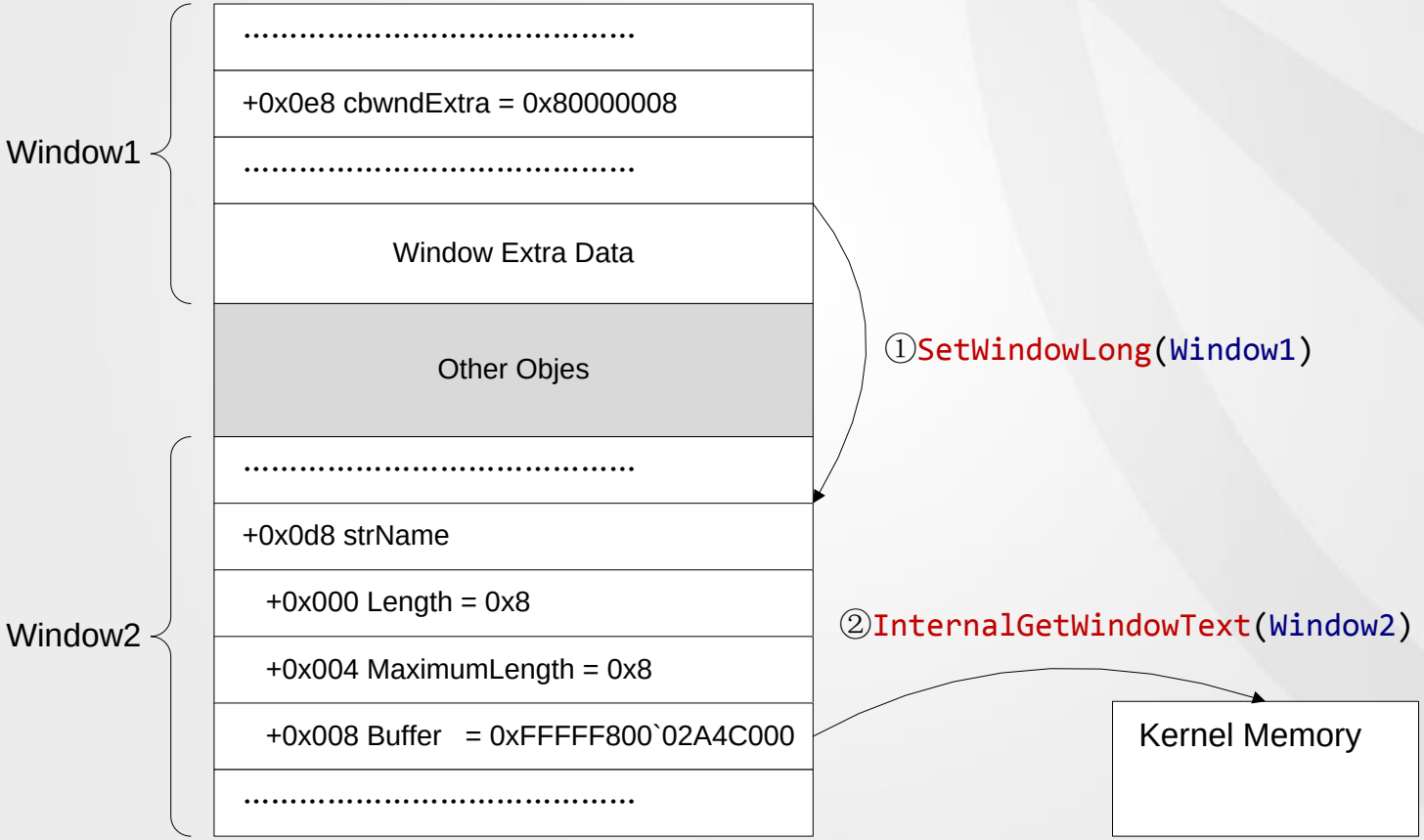


What can we do now?

Read from anywhere

- Two APIs:

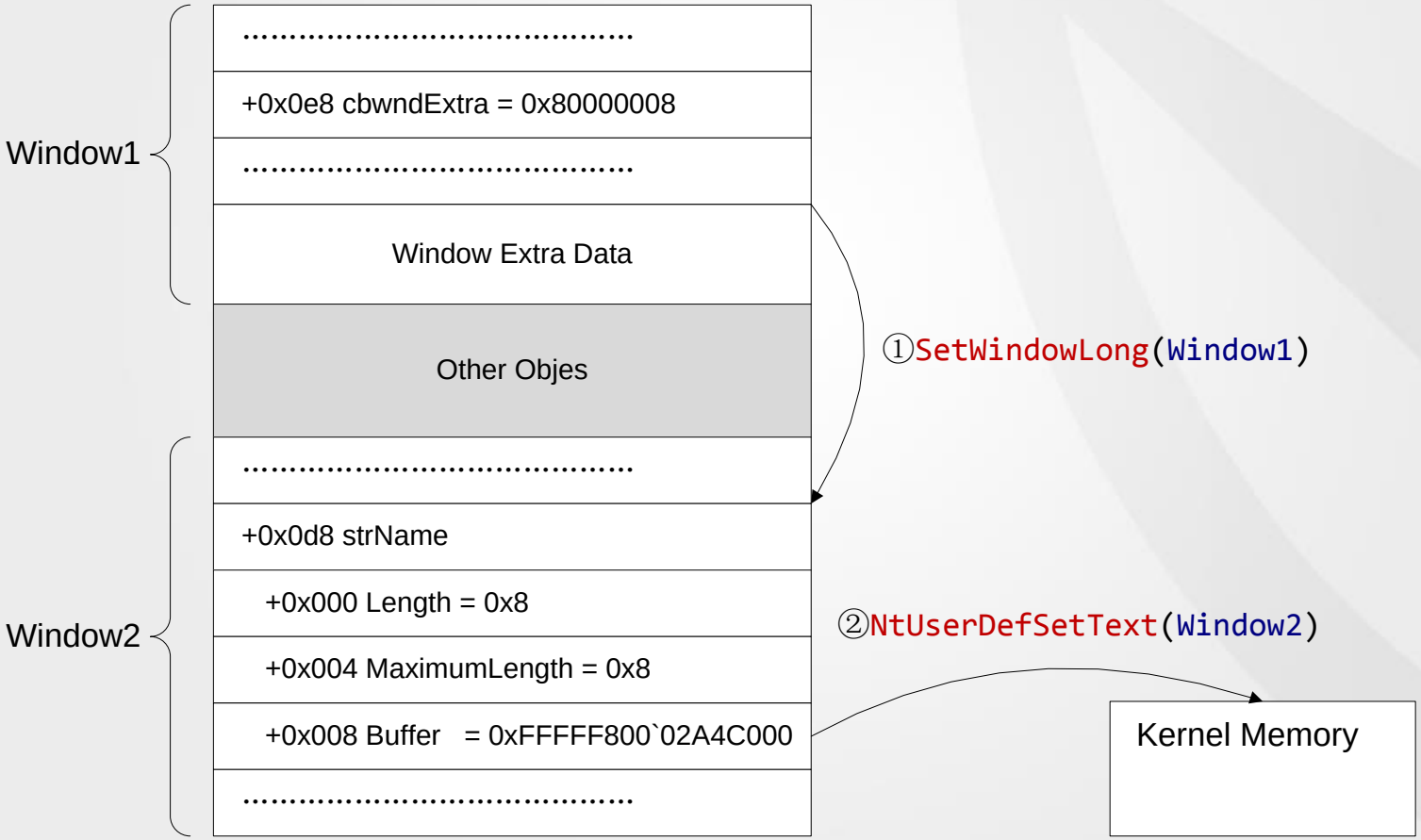
```
LONG WINAPI SetWindowLongW( HWND hWnd, int nIndex, LONG dwNewLong);
int  WINAPI InternalGetWindowText( HWND hWnd, LPWSTR lpString, int nMaxCount);
```



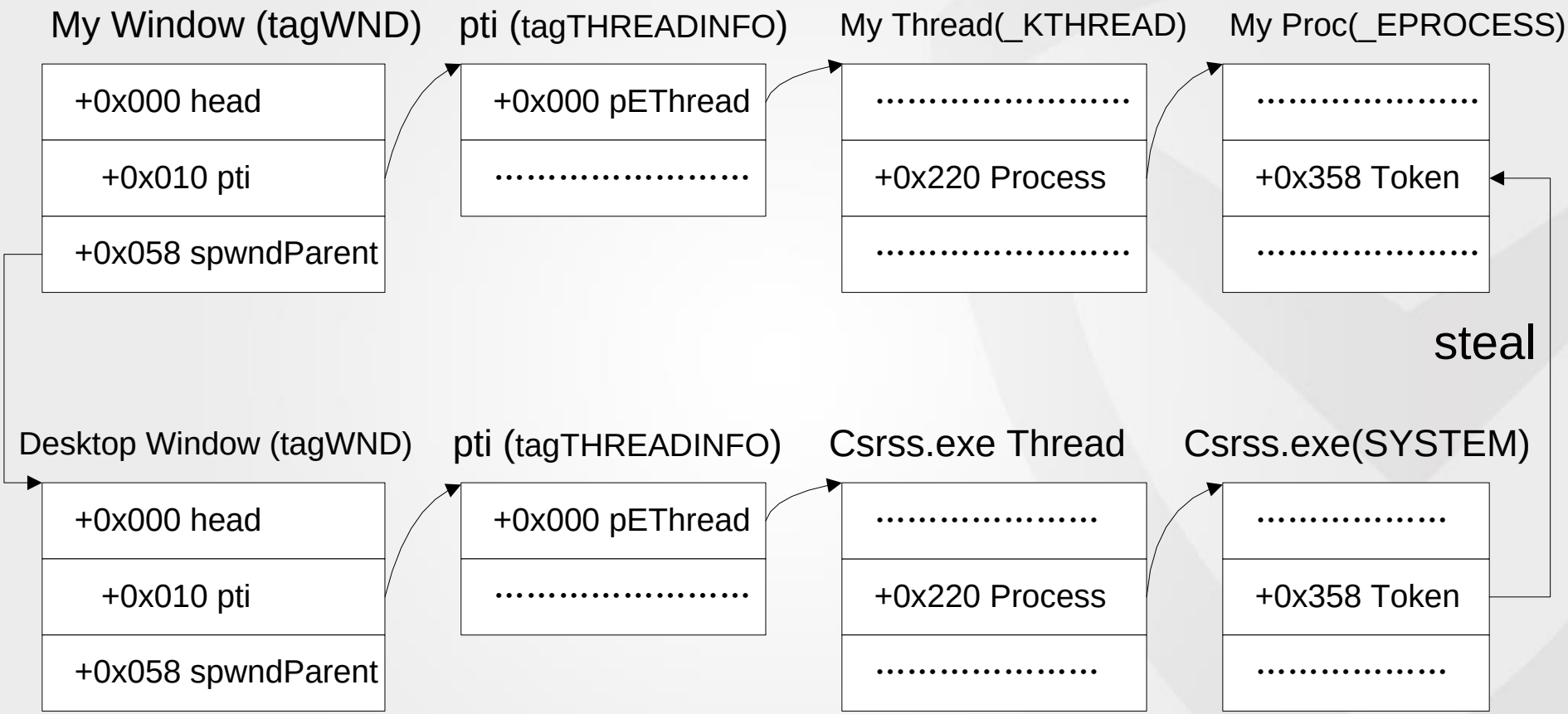
Write to anywhere

- Two APIs:

```
LONG WINAPI SetWindowLongW( HWND hWnd, int nIndex, LONG dwNewLong);
BOOL NtUserDefSetText( HWND hWnd, PLARGE_STRING pstrText );
```



Steal SYSTEM Token



Real Case: CVE-2016-0174

①Load a font to create a PFFOBJ

②Free a PFFOBJ

③Replace the freed Obj

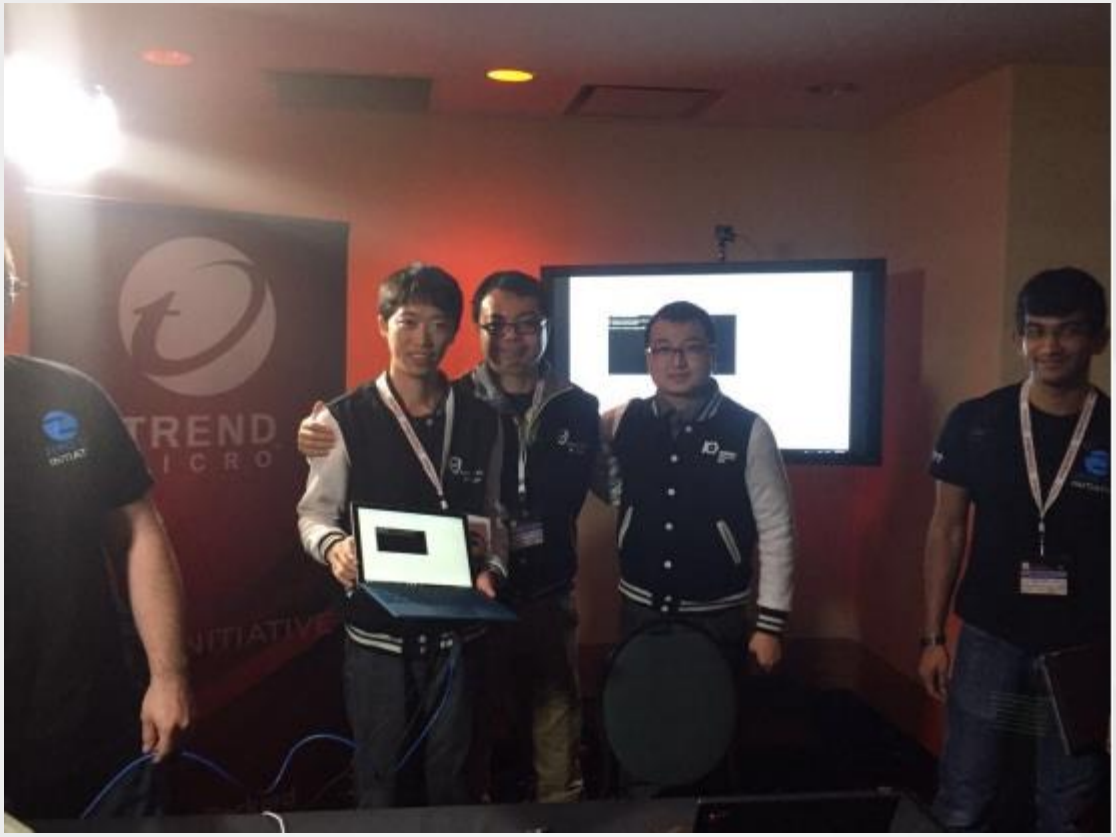
④Free the font

⑤or dword ptr [rax+0Ch], 2

offset	0xEB	0xEA	0xE9	0xE8
Bin:	0000 0000	0000 0000	0000 0000	0000 0000
Bin:	0000 0010	0000 0000	0000 0000	0000 0000

- cbwndExtra: 0 → **0x2000000**

Pwn2Own 2016 Flash



Other Case

dec dword ptr [rax]

offset	0xEB	0xEA	0xE9	0xE8
Bin:	0000 0000	0000 0000	0000 0000	0000 0000
Bin:	FFFF FFFF	FFFF FFFF	FFFF FFFF	FFFF FFFF

- cbwndExtra: 0-1=**0xFFFFFFFF**

inc dword ptr [r10+8]

offset	0xEB	0xEA	0xE9	0xE8
Bin:	0000 0000	0000 0000	0000 0000	0000 0000
Bin:	0000 **1**000	0000 0000	0000 0000	0000 0000

- cbwndExtra: 0 → **0x8000000**

How to exploit 0 → 1

Q1: Where to write?

- tagWND.cbwndExtra

Q2: What to write?

- A big value

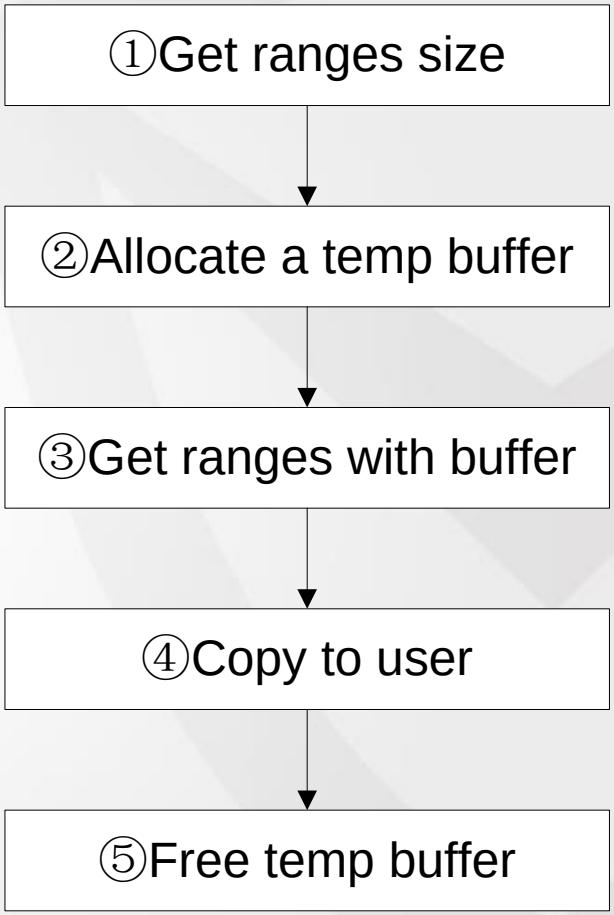
Q3: What can we do now?

- Read from anywhere
- Write any value to anywhere
- Steal csrss.exe`s token

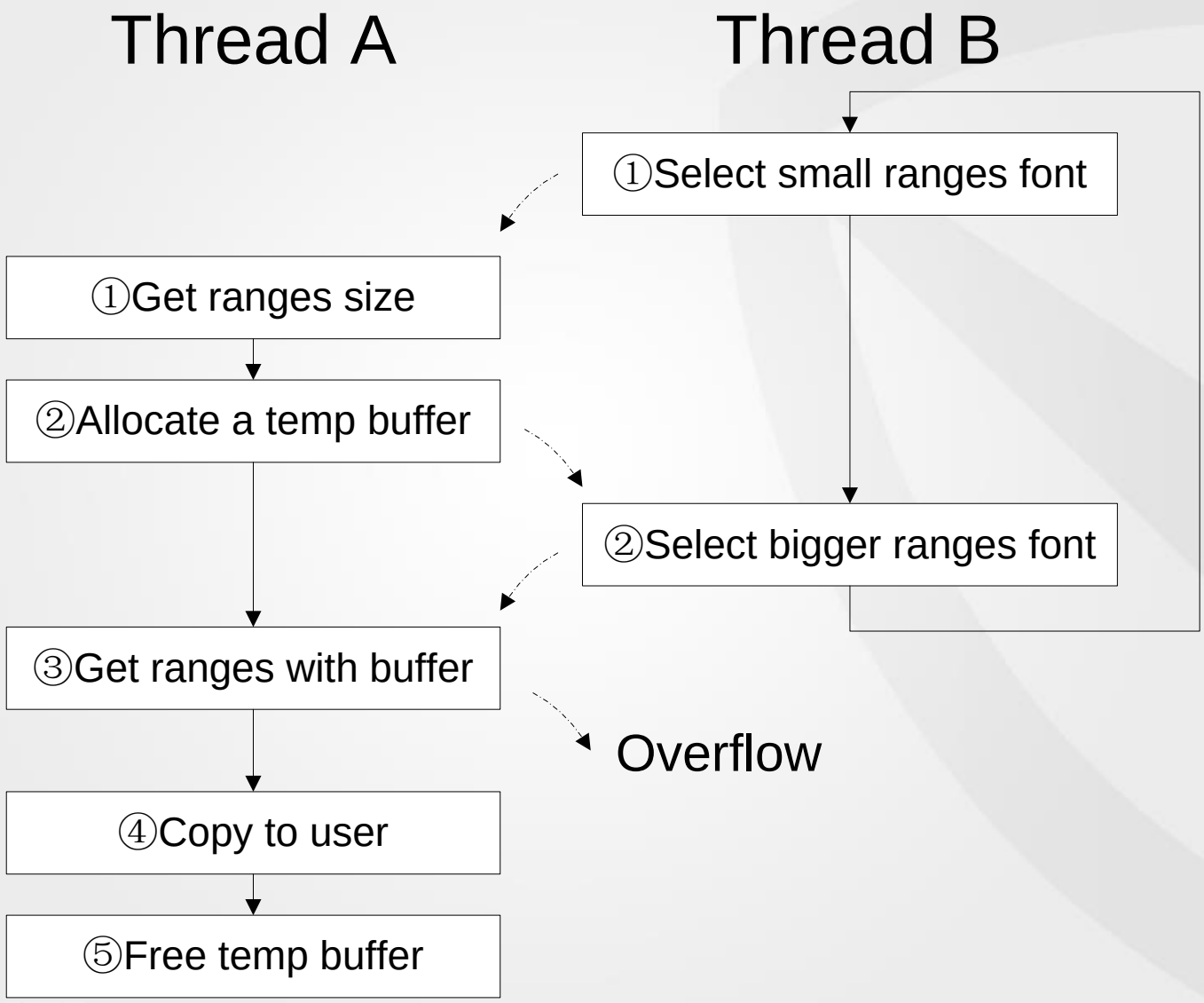
$$1 \rightarrow 0$$

The Function - CVE-2016-3355

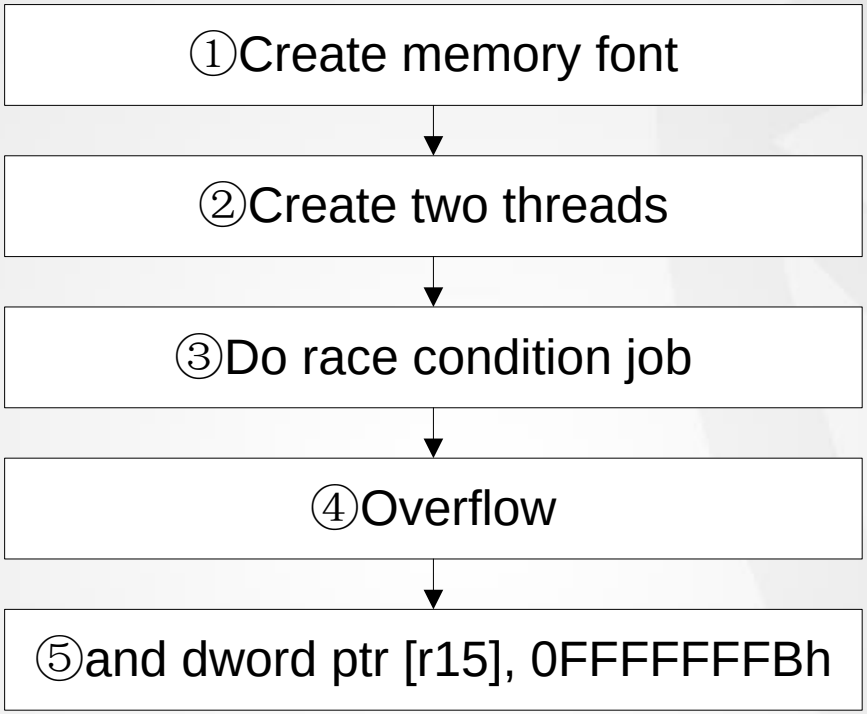
```
DWORD  NtGdiGetFontUnicodeRanges(HDC hdc, LPGLYPHSET lpgs)
{
    DWORD PreSize;
    DWORD PosSize;
    DWORD ReturnSize;
    PVOID pTmpBuf;

    ReturnSize =   0;
    PreSize =   GreGetFontUnicodeRanges(hdc, 0);
    if ( PreSize && lpgs )
    {
        pTmpBuf = AllocFreeTmpBuffer(PreSize);
        if ( pTmpBuf )
        {
            PosSize = GreGetFontUnicodeRanges(hdc, pTmpBuf);
            if ( PosSize && PreSize == PosSize )
            {
                ProbeAndWriteBuffer(lpgs, pTmpBuf, PreSize);
                ReturnSize =   PreSize;
            }
            FreeTmpBuffer(pTmpBuf);
        }
    }
    return ReturnSize;
}
```



The Problem



My Exploit



- Hex: 0xFFFFFFFB
- Bin: 1111 …… 1111 1011

Pos 31 7 3 2 1 0

My Exploit

- Length? 0x4 → 0x0, 0x14 → 0x10

- Flag?

```
1: kd> dt win32k!tagWND
.....................................
   +0x014 bHasHorizontalScrollbar : Pos 2, 1 Bit
.....................................
   +0x018 bStartPaint      : Pos 2, 1 Bit
.....................................
   +0x01c bWS_EX_NOPARENTNOTIFY : Pos 2, 1 Bit
.....................................
   +0x0ac bRedirectedForPrint : Pos 2, 1 Bit
```

- Type?

TYPE_CLIPDATA (6) → TYPE_MENU (2)

TYPE_MONITOR (0xC) → TYPE_ACCELTABLE (0x8)

TYPE_KBDFILE (0xE) → TYPE_DDECONV (0xA)

The Refcount!

- Win32k object:

```
1: kd> dt win32k!tagWND -b
   +0x000 head             : _THRDESKHEAD
      +0x000 h                : Ptr32
      +0x004 cLockObj         : Uint4B
      +0x008 pti              : Ptr32
      +0x00c rpdesk           : Ptr32
      +0x010 pSelf            : Ptr32

1: kd> dt win32k!tagMONITOR -b
   +0x000 head             : _HEAD
      +0x000 h                : Ptr32
      +0x004 cLockObj         : Uint4B
```

```
1: kd> dt win32k!tagMENU -b
   +0x000 head             : _PROCDESKHEAD
      +0x000 h                : Ptr32
      +0x004 cLockObj         : Uint4B
      +0x008 hTaskWow         : Uint4B
      +0x00c rpdesk           : Ptr32
      +0x010 pSelf            : Ptr32
```

- Modify cLockObj:

```
PVOID FASTCALL HMAssignmentLock(PVOID *ppobj, PVOID pobj);

PVOID FASTCALL HMAssignmentUnlock( PVOID *ppobj );
```

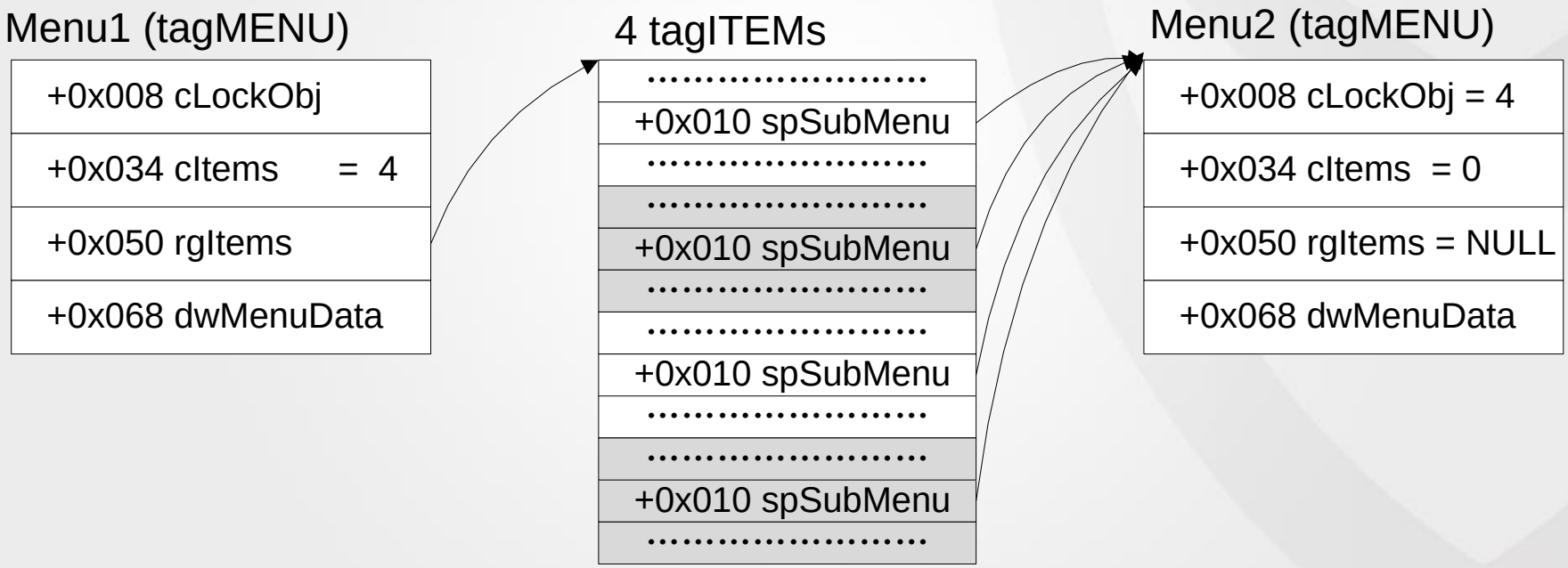
- Check cLockObj:

```
BOOL HMMarkObjectDestroy( PVOID pobj);
```

Basic object: Menu

- Two APIs:

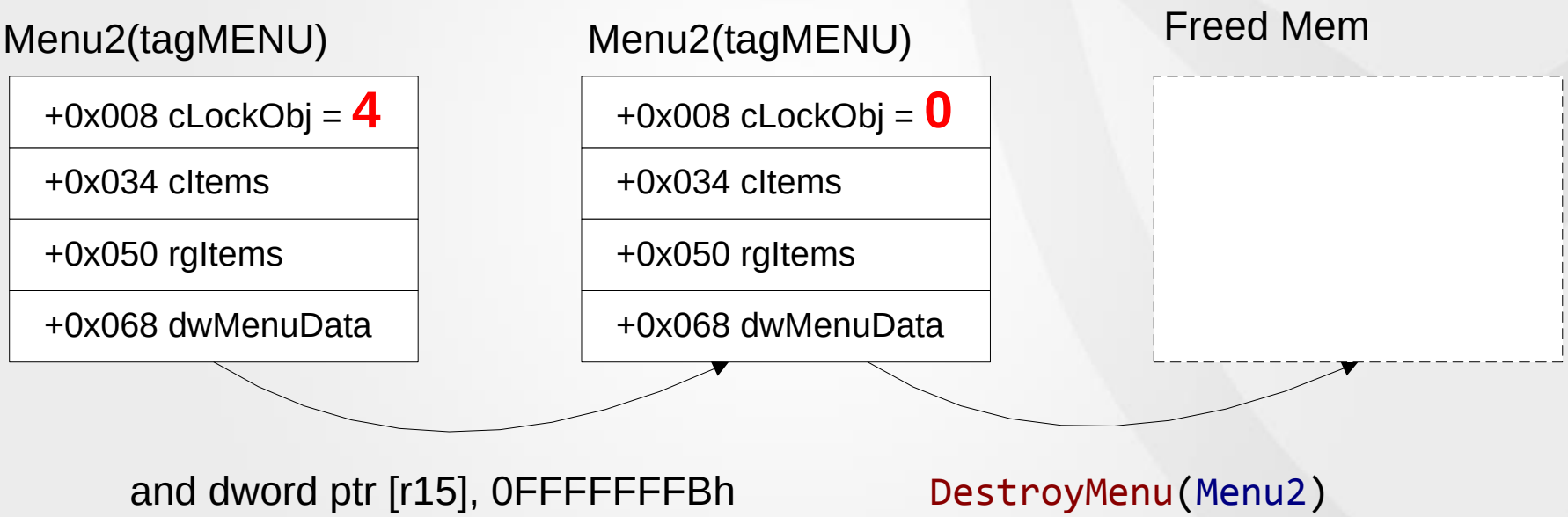
```
HMENU CreateMenu ();

BOOL AppendMenu(HMENU hMenu,UINT uFlags,UINT_PRT uIDNewItem,LPCWSTR lpNewItem);
```



Make a Use-After-Free

- API:

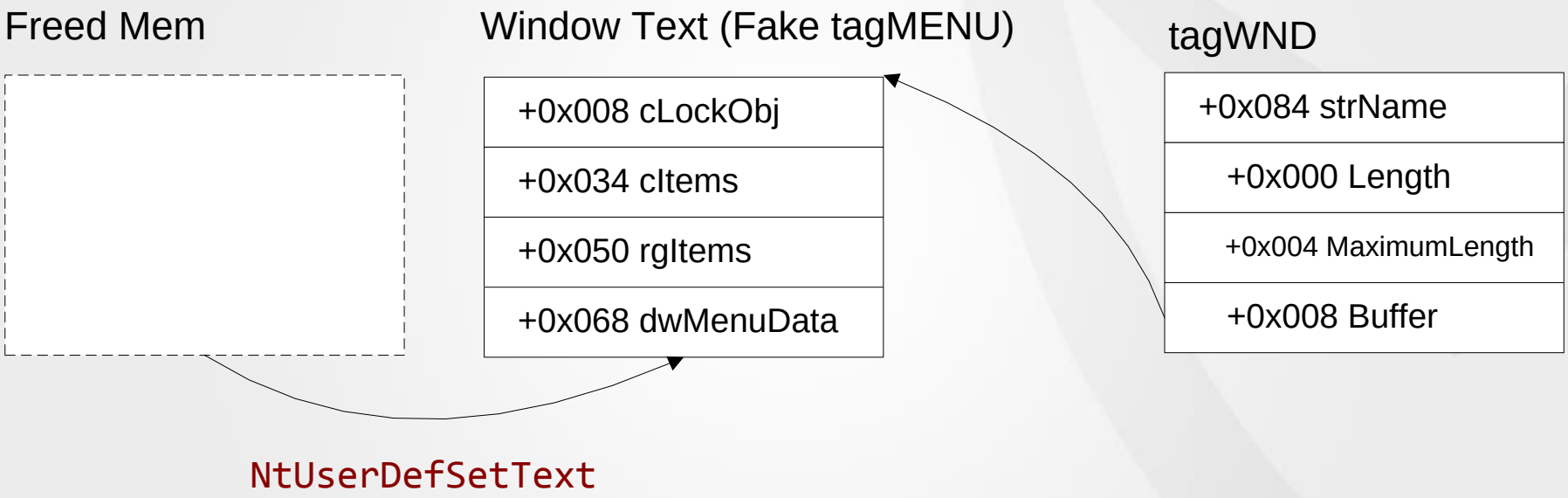
```
BOOL WINAPI DestroyMenu ();
```



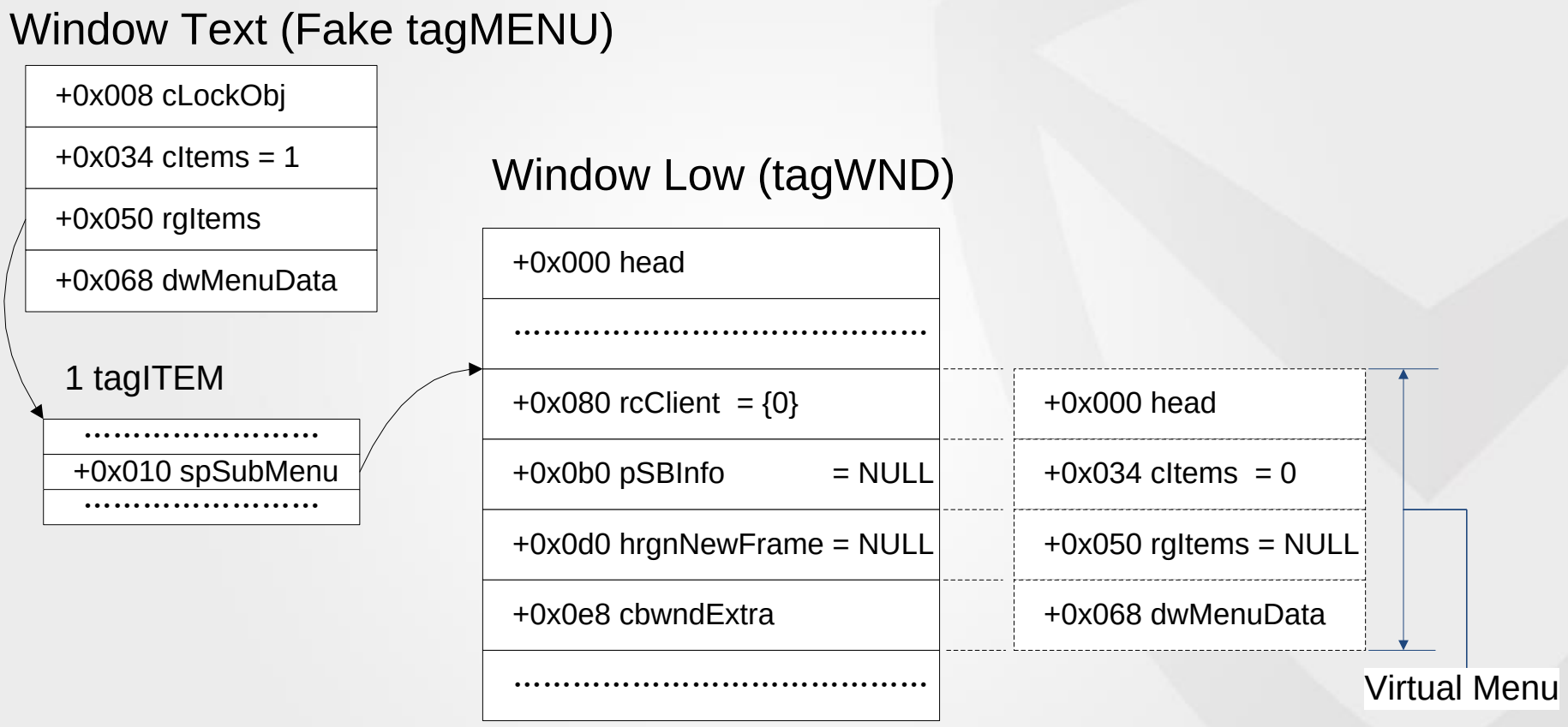
Take Pos

- Native API:

```
BOOL NtUserDefSetText( HWND hWnd, PLARGE_STRING pstrText );
```



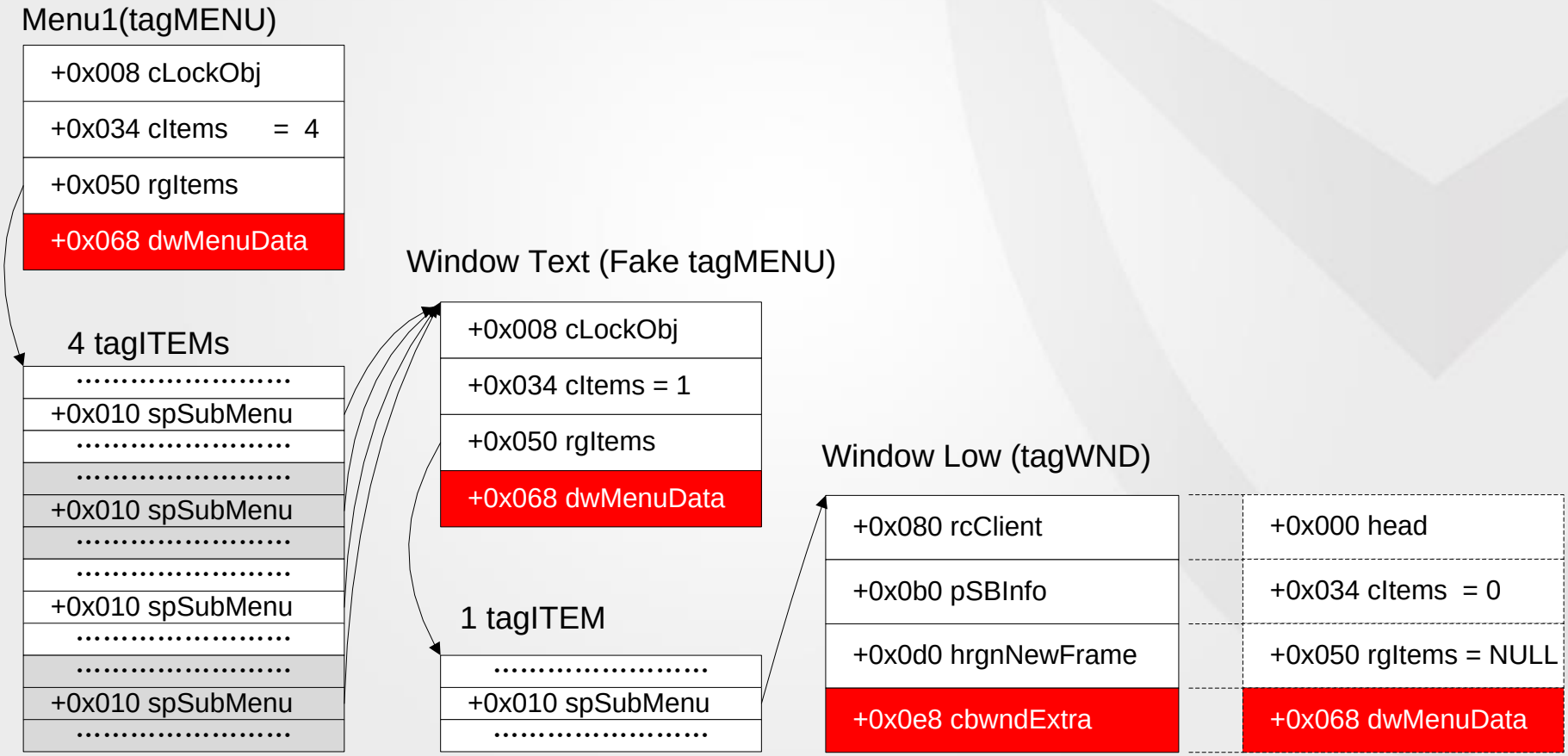
Fake tagMENU



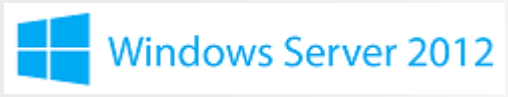
Change the cbwndExtra

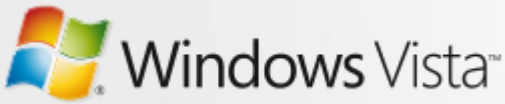
- API:

```
MenuInfo.fMask      =   MIM_MENUDATA | MIM_APPLYTOSUBMENUS;
MenuInfo.dwMenuData =   0xFFFFFFFF;
SetMenuInfo(Menu1,&MenuInfo);
```



Let`s rule them all!





How to exploit 1 → 0

Q1: Where to write?

- tagMENU.cLockObj

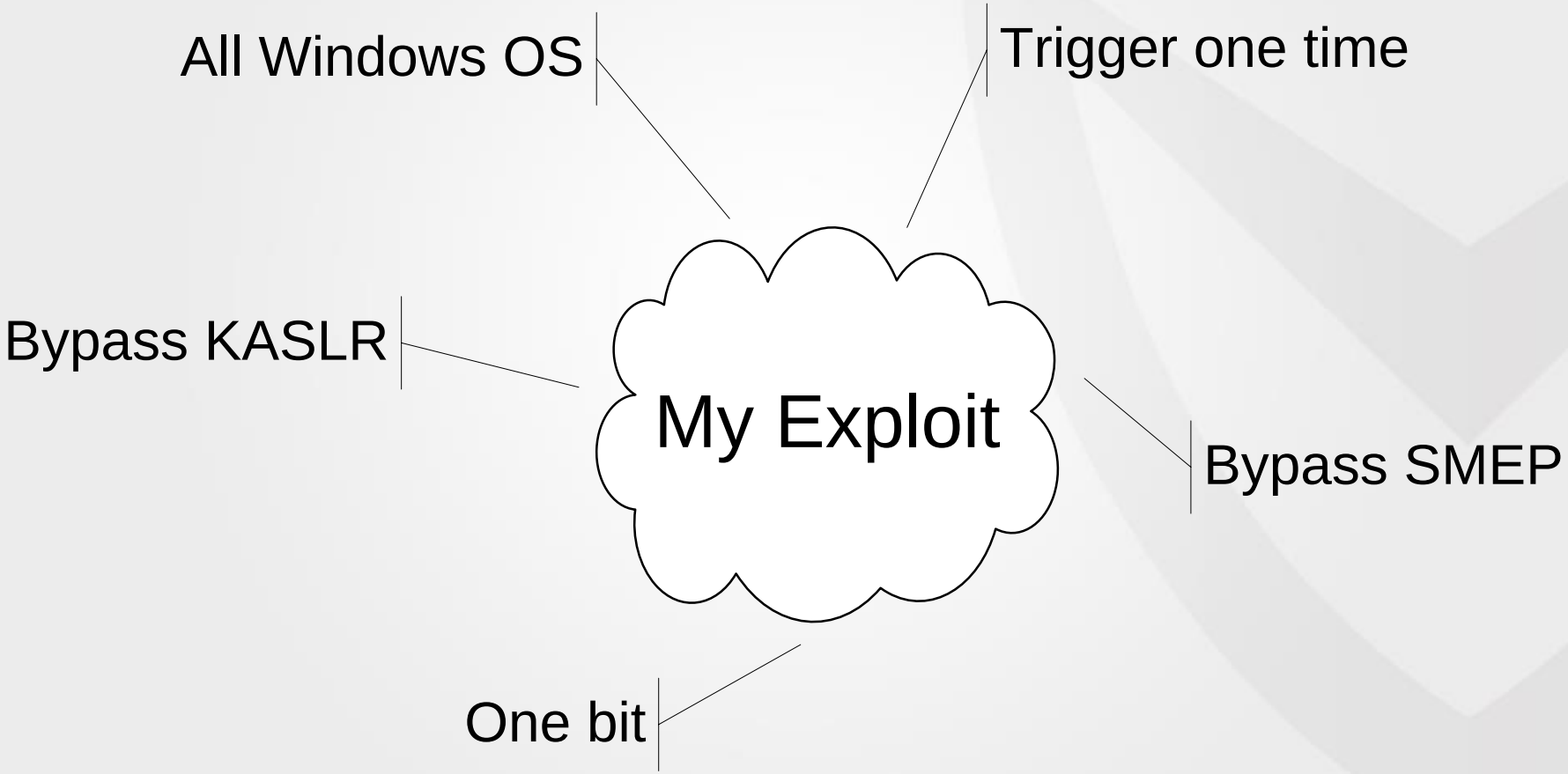
Q2: What to write?

- 0

Q3: What can we do now?

- Make a UAF
- Control the pointer
- Write a big value to tagWND. cbwndExtra

Summary



Q & A